Path Planning Algorithms for Agricultural Machines

T. Oksanen and A. Visala

Helsinki University of Technology, Automation Technology Laboratory, 02150 Espoo, Finland.

E-mail: timo.oksanen@tkk.fi

ABSTRACT

If the field plot shape is not rectangular and if it contains obstacles, the coverage path planning problem is hard to solve for a non-omnidirectional machine. Scientists have developed several algorithms to solve this coverage path planning problem, but all of them have pros and cons. If the machines were omnidirectional and turning times were decreased to insignificant, the problem would be quite easy to solve using known robotic path planning methods. Traditional agricultural machines, like tractors, tractor-trailer combinations, self-propelled harvesters and other man-driven machines are slow to turn at headlands. This is the most differentiating property of the problem formulation compared to traditional robotic coverage path planning, which has dealt mainly with omnidirectional kinematics. In this article two different algorithms are presented to solve the coverage path planning problem for agricultural machines.

The first algorithm is a higher level algorithm to split a complex shaped field plot to smaller parts is presented. The higher level splitting algorithm is presented in detail in this article. The algorithm can handle any field, including obstacles. The algorithm is based on trapezoidal split, merge and search. The algorithm is suited to any kind of vehicle, which is described with a few parameters, like working width and turning time function. In the latest version, the required headlands are generated automatically and there is also a possibility to define regional restrictions as forbidden driving directions. With this formulation it is possible to take into consideration the previous operations, under drains and steep gradients.

The second algorithm utilizes bottom-to-top approach. It is designed for real-time usage and it solves the problem recursively: the operated area is removed from the field and the algorithm is repeated until the whole field plot is completed. In the development phase of algorithm, a simulator has been utilized. The underlying idea is to calculate the efficiency for all possibilities to make one trip around the field and to select the best one. It is assumed that every new swath is side-by-side to the some previous one or to the boundary of the field plot. However, even if the underlying idea is simple, the search space explodes when the number of corners of the field plot raises and heuristics is needed in order to restrict the number of possibilities without losing optimality. The algorithm is suited to any kind of vehicle, which is described with a few parameters like working width and minimum turning radius. Preliminary results are very promising and are presented in the article.

Keywords: Agricultural robotics, coverage, field operations, field plots, field robots, mission planning, path planning

1. INTRODUCTION

Tractors and self propelled farming machines moving on the fields are traditionally driven by a human driver. The human driver has designed the driving strategy of a single field by himself, without any assistance. He/she has chosen the strategy on the basis of type of task, working machine and especially on experience. In family size farms the strategy is based mostly on experience and the driving strategy remains the same over the years. If the field shape is not rectangular or if there are obstacles, the generation of the strategy is not so simple. Usually the most optimal solution is not even the goal, a nearly optimal feasible solution is sufficient.

Autonomous field machines or robots will come, sooner or later. The new issues for autonomous operation are safety, detection of failures, recovering after failures, and automatic refilling or emptying. As a human driver no longer operates the machine, automatic path planning is also needed, whereas 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.

In order to be autonomous, a mobile robot has to know or solve four things: what is the task to do, what is the way to complete it, what is already known and what is the position related to known (Murphy, 2000). In agricultural applications the task is usually given by a human operator. Also the last two are more or less solved, because fields are mapped environment and accurate positioning devices are on the market. So the most difficult part in agricultural robot applications to be solved by artificial intelligence is mission planning. Path planning is one of the key tasks in mission planning (Reid, 2004).

Roboticists understand path planning as an algorithm that has to find a path from place A to place B so that no collisions with obstacles occur and the path is optimal with respect to a certain measure, for example traveling in minimum time or using minimum energy. In robotics, path planning has been divided into two classes, to 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 viewpoint (Murphy, 2000).

In agricultural robotics, the task is usually to cover the whole field, not only going from point A to point B. This kind of path planning is so different from traditional robot path planning that the algorithms are not directly suitable. Similar applications are demining, painting, mowing, mapping unknown environments etc. These kinds of autonomous applications are so new (or coming) that need for this kind of path planning has appeared lately.

In Gray (2001), the orchard tractor navigation development was reported. Orchards are not open fields, trees form blocks in which the navigation is one problem to be solved and the whole mission is another. In Sørensen et al. (2004) a method for optimizing the vehicle route by defining the field nodes as a graph and formulating it 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. 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. Choset (2001) makes a survey of coverage path planning algorithms and classifies the algorithms to three classes: approximate, semi-approximate and exact. As a conclusion it may be said that the path planning of coverage type task is still under research and a general usable optimal and provable algorithm has not been developed yet, so there is space and need for further research of path planning.

In this article, two different algorithms to solve the coverage path planning problem are presented. First of them uses top-to-bottom approach, the field is split into subfields with a certain shape. The second algorithm uses bottom-to-top approach, after each swath all possible routes are simulated over certain limited horizon and the best of them is selected in each step. At the end of this article, a quick comparison of algorithms is presented.

2. SPLIT AND MERGE BASED ALGORITHM

The shape and size of fields varies a lot, especially in Finland fields are usually bounded by other terrain types, like forests, lakes, rocky terrain etc., and shapes are far from orthogonal and convex. If the field is convex and it does not contain any obstacles, path planning for agricultural tasks is quite simple, and only the main driving direction has to be found. The whole field is driven in that direction except headlands if needed. The selection of the main driving direction on the basis the longest edge of field has been a rule of thumb for farmers. Here this rule of thumb based on common sense has been dismissed and it will be checked if the result is still the same.

If the field is nonconvex which means that it has "bays", finding the optimal solution is hard. One possibility to solve the problem is to use split and merge approach for segmentation used in computer vision. The field is split into simple shaped subfields which are convex or near convex, an optimal solution is found for driving in the subfields and finally the solutions are combined. If the shape of a subfield is for example rectangular, finding the optimal driving strategy is pretty simple, even if not trivial. The drawback of this method is that the output, the driving route, is not necessarily a globally optimal solution, but suboptimal.

For some environment and some operations there are limitations for driving direction. For example the underdrainage system made based on height variation limits the ploughing directions, for certain soil types. Also the driving direction in previous operation may limit the driving direction, or more generally the chain of field operations. For example in tilling it may be suggested not to drive in the same direction as the field is ploughed. Another case when driving directions may be wanted to limit is a series of small permanent obstacles and wide working machine, like electric poles and sprayer. Then in the surroundings of electric line it may not be suitable to drive in directions that differ from the direction of electric line only a little bit.

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 is made, at least in Finland. The requirements for a good coverage path planning algorithm are: suitability for all kind of fields, for all kind of machines, and efficient enough in order to be solved in reasonable time.

This chapter concentrates on the higher level algorithm to divide a complex shaped field into simple subfields in which the route planning is easy to do. The algorithm is suitable for all kind of crop farming machines where the task is to do some action in all places in the field exactly once.

2.1 Definitions

Certain type definitions have been set. The field is considered as a uniform 2D region which may contain obstacles. An exterior polygon describes the field outer boundaries and interior polygons describe the obstacles. Vertices are corner points of the polygon. Edges are line segments that connect vertices.

A trapezoid is a quadrangle which has two opposite parallel sides. A triangle is a special case of a trapezoid. A block is a polygon which is constructed by merging two or more trapezoids in their parallel and equal sides – in block two edges are parallel. Headland is a region in which the machine is to be turned. Prohibited region is a region which is a part of field where certain driving directions are prohibited.

2.2 Objective

The objective is to divide a complicated field into subfields. The algorithm searches first largest or most efficiently driven subfields, removes them from the original field and keeps finding subfields until the whole field is computed. In 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 certain criterion which takes into consideration the area and the route length of block and the efficiency of driving.

2.3 Splitting

Crop farming machines have certain working width, which usually remains constant. The requirements for best efficiency and quality are: the driving lines are exactly side by side, no gaps, no overlapping and the turning in headlands is made in minimum time. Parallel swathing assistants or light bars or autopilots help human driver to keep the machine in lane.

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

In this algorithm the field is split into trapezoids, this belongs to the set of exact cellular decompositions (Latombe, 1991). All vertices of the exterior polygon are projected at given direction to all sides and trapezoids are detected. If the field contains obstacles, the interior polygon nodes are also projected to all sides of the polygons. An example of triangulation is presented in figure 1. In the field on the left the number of trapezoids is 11 and in the field on the right the number is 18.

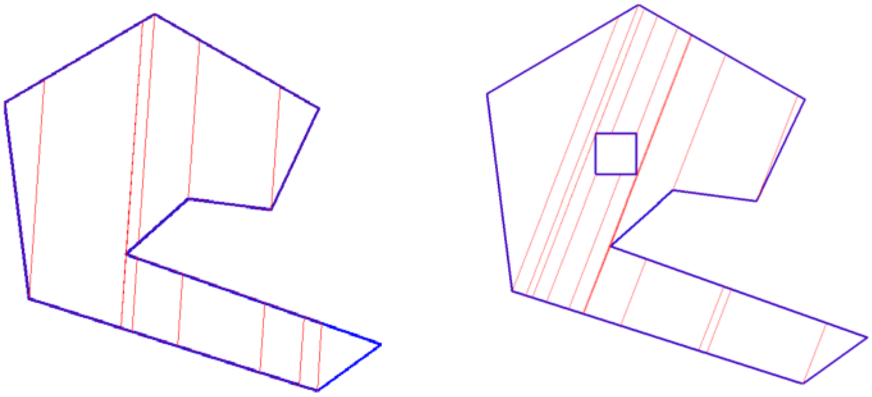


Figure 1. Two examples of triangulation.

2.4 Merging

After splitting the field into trapezoids, the next step is to combine 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 which are far from rectangular shape and should be handled in later phases separately. The minimum angle between matching side and ending side is set to 20° (90° means right angle). The example of merging trapezoids is presented in figure 2.

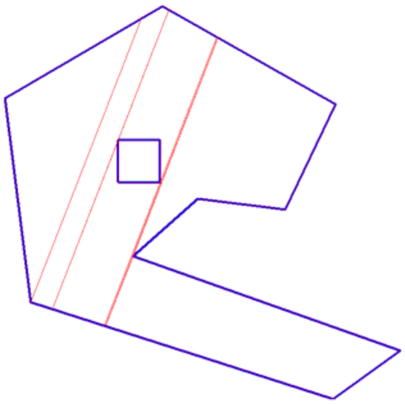


Figure 2. Merged trapezoids.

2.5 Selection Criterion

The idea is that the regions which are most efficient to handle are driven first and the same algorithm is applied iteratively for the rest until the whole field is handled. The region to be selected in each step is a block, the best one of them has to be selected.

The area of the block, the distance of route fitting inside the block and the efficiency of driving are variables in selection criterion. The area is simply the area of the block. The distance is calculated using the working width information and the headland width is subtracted from that. The distance corresponds to the distance that can be driven at normal driving speed with operational part of machine working. The time consumed in the block is estimated from the distance calculated previously and the time spent in headlands is added. The estimate of turning time in certain headland angle can be calculated for example using optimal control techniques (Oksanen and Visala, 2004) or by splines (Noguchi et al., 2001). In perpendicular headlands (compared to driving direction) the quality is best (minimum overlapping in headlands).

In practice efficiency is the primary variable which should be maximized, but this leads easily to a situation where narrow and long blocks are selected first. That leads to an unwanted combined solution. Therefore the other two measures are needed too. All the measures (area, distance, efficiency) are normalized and the cost is a weighted sum of these. Currently the tuned weights are: efficiency 65%, area 15% and distance 20% and these are used in the results below.

If some subfields are already selected, a bonus 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 only. With most cropping machines, a small correction in direction leads to inefficiency and to quality loss.

2.6 Search of the Driving Direction

Splitting into trapezoids and merging them to blocks is made in certain direction. However, the direction is not known and it has to be solved. The characteristics of the blocks are not changing smoothly when the direction is changed in infinitesimal steps, so the cost function of search is not smooth. This means that all possible directions should be gone through (between 0 and 180°) and it takes a lot of calculation time. The following 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 divided by two.
4. New search directions are added to the both sides of the three best directions.
5. Cost is calculated in directions which are not yet calculated.
6. If the goal resolution is reached, exit, otherwise go to step 2.

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

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

2.7 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 directions of blocks after first iteration vary from each other, it is evident that one end of 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 a block which has very steep headland angle e.g. below 15° (90° means again right angle), then the headland can be driven by bending the driving line. The number of swaths needed in headland is input variable for algorithm.

2.8 Prohibited Driving Directions

As mentioned above, for some environments and some operations there are limitations for the driving direction. This can be formulated to this path planning algorithm by defining a prohibited region, in which 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 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 removing phase, the prohibited regions are cropped if needed. It is required that prohibited regions are inside the field region.

2.9 Test Results

Previously (Oksanen et al., 2005) the test results with 1500 real fields were presented. The conclusion from those tests is that this algorithm works nicely for fields with straight edges. The solutions for fields with curved edges are valid, but not so efficient. Here is presented latest results.

Automatic determination of headlands was developed. In the figures below, the headlands are drawn with blue color, and the main swaths are drawn with green. In figure 3, a H-shaped field is presented with the solution. At first, the algorithm has found 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.

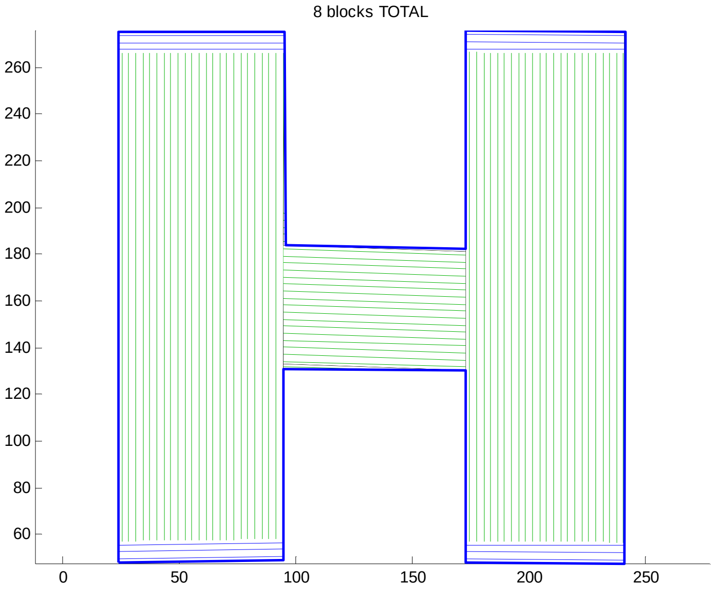


Figure 3. H-shaped field with headlands.

In figure 4, a field with many bays is shown.

Figure 4. Field with many bays.

As it can be seen, the main driving direction was determined on the largest block in the middle. For three of four bays the same driving direction is found to be top-rated (NB: a small bonus is given to 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.

2.10 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 6, a C-shaped field is shown. On the left is a solution without any prohibited regions. The final solution consists of 5 blocks, saving two headlands. On the right, a fictional escarpment (steep slope) is inserted on one corner, this is marked with dashed line and a small red triangles, "bow", represent 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 the tractor does not have enough horsepower to drive 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.

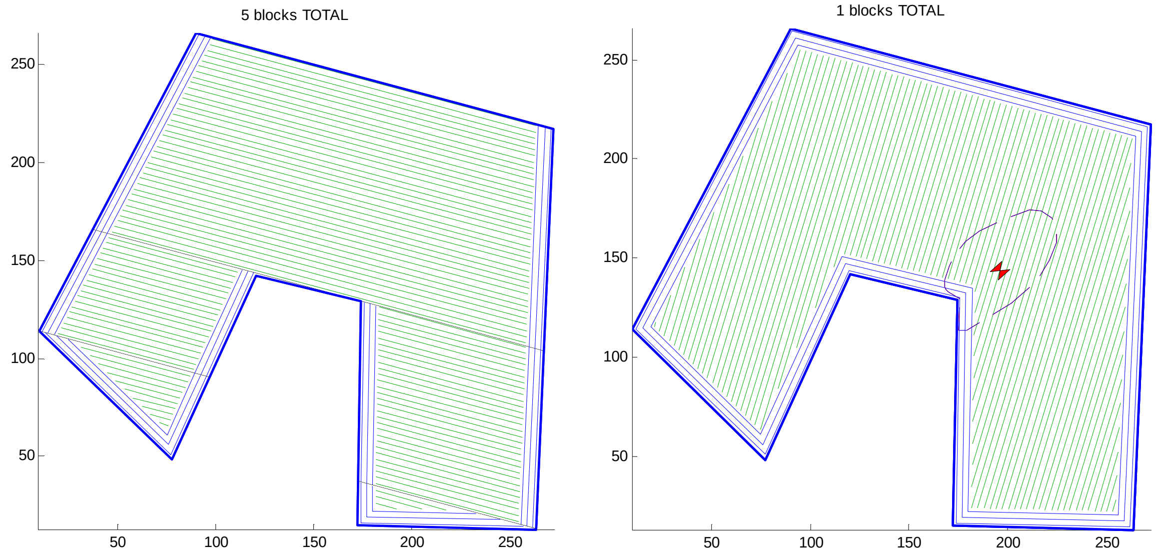


Figure 5. C-shaped field without and with prohibited regions.

In Northern Europe most of the field plots are underdrained. Underdrainage is important especially in soil types which are not transmitting water easily. In certain field operations, like in ploughing it is not recommended to drive in the same direction as the pipes are laid; ploughed furrow is also kind of "pipe". When the furrows and pipes cross, the effect of drainage is at its best. In figure 7, a field with an underdrainage system is presented. A bold blue dashed line represent 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.

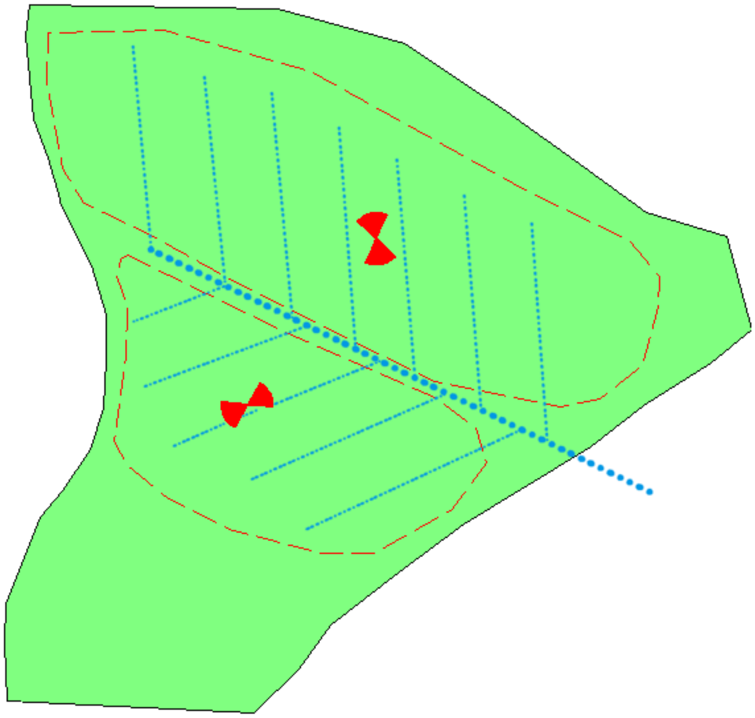


Figure 6. Field with underdrainage system.

In figure 8, on the left the solution of 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 requirement of prohibited region. Actually 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.

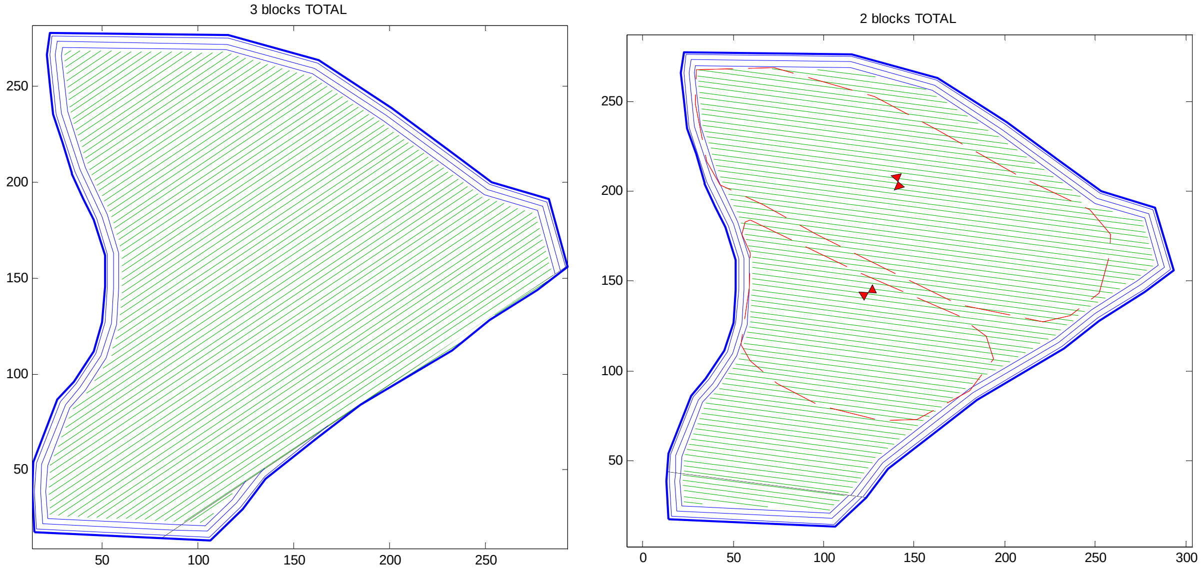


Figure 7. Solution without and with underdrainage.

3. RECURSIVE ONLINE ALGORITHM

Online path planning algorithm means that it may be used online in the vehicle, so that the planning calculation runs in the computer so fast that the vehicle does not need to limit its speed in order to wait for the solution. *Offline path planning algorithm* is a precalculated path. The online algorithm has better adaptively, but offline algorithm may lead to better total solution.

Polyline is a continuous line composed of one or more line segments, the line segments. A c*losed polyline* is a polyline where the starting point and ending point is common. *Vertex* is the point where polyline segments end. *Edge* is a line segment in 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.

Here *Field* is a uniform 2D-region, made by exactly one exterior closed polyline and free number of interior closed polylines representing obstacles.

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 know 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, a precalculation and sampling may be used, see e.g. Oksanen and Visala (2004) or Noguchi et al. (2001). This precalculation applies until the machine properties remain the same. The algorithm has to solve the turning time so many times, that look-up table is needed in order to keep the computing time reasonable.

At this phase of development the refilling/emptying the machine is not considered, but it may and will be added to the algorithm later.

3.2 Required Sub Algorithms

The most important sub algorithm is so called *polygon offsetting*. The aim is to move each edge of region inwards (or in some cases outwards) so that the perpendicular distance is between subtracted region and original is wanted. This problem is analog to the field operation where one round is driven around the field once, doing some operation, and the inner boundary of operated area is to be identified. This is a well known problem in computational geometry. There are several methods to solve this, see e.g. Yang and Huang (1993), Choi et al. (2001a) and Choi et al. (2001b). Here it is used a *straight skeleton* method, (Aichholzer et al., 1995) and (Felkel and Obdrzalek, 1998). The algorithm can handle convex regions as well as holes in the region.

3.3 Basic Idea

A simple field with a shape of a rectangle should be considered (fig. 8). Let the initial state of machine be I, both the location and the direction. The A-B-C-D polygon represents a region which interior needs to be operated and these points may be considered also as critical vertices. As the limitation 1 was set, a restricted set of movements exists. The problem is to search the best route.

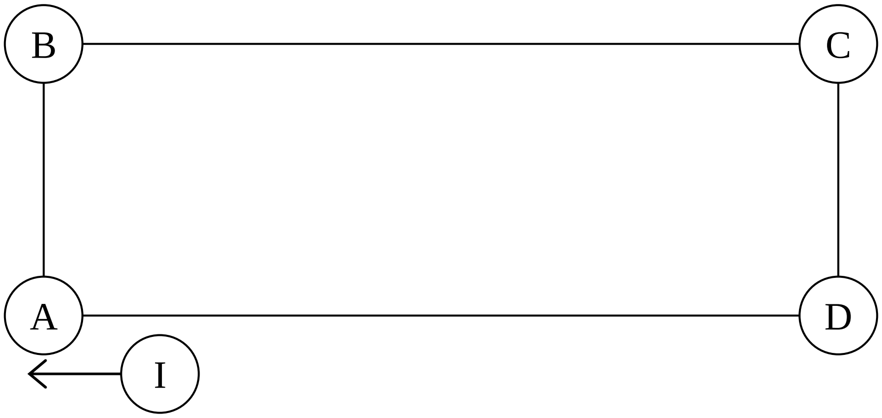


Figure 8. Simple field with a shape of a rectangle.

In order to find the global optimum, according to 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 when the number of corner points increase and/or area/working width ratio increases.

In control engineering, a similar problem arises in optimal control. Search from all possible control functions is impossible, generally. Optimal control methods that minimize certain criterion exist, but in real time operation and for nonlinear systems those are not practical. Model predictive control is a way to improve feedback control: in certain time step and control, 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 in each step, and in the next time step all the computations are repeated. (Maciejowski, 2002).

Here the same analogy is applied to path action search. The "control horizon" and "prediction horizon" are actually equal in tests later, but "prediction horizon" could be longer, roughly it should be multiple of "control horizon". Let the "control horizon" be defined as *search horizon* later in this chapter.

The search horizon is defined as:

- Start point is the nearest vertex to current position (or initial point at the startup).
- Starting direction is free, clockwise (CW) or counter-clockwise (CCW).
- Stop when near start point (A', offset A).
- Zero or one reversion in direction.
- Some segments may be skipped.
- If the segment is skipped in one direction, it must be skipped also in the other direction.

The possible routes in the simple field of figure 8 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 direction CW would be I-A-B-C-D-C'-B'-A' ; I-A-B-C-B'-A' ; I-A-B-A', plus the subsets and the same in CCW direction.

This leads to

$$2\cdot\left(2^{N}+\sum_{i=1}^{N-1}2^{N-1}\right) \tag{1}$$

upper limit of choices, where N is the number of critical vertices. The multiplier 2 is the number of directions, the first part is for circular driving and the second part for reversing in different points. There is some redundancy, because some sub solutions (with skipping) in circular driving versus reversive driving are congruent; therefore this equation gives only the upper limit of choices.

For N=4 (in our case), the upper limit of choices will be 96. For N=7, it is 1152 and for N=10 it is 12,288 and for N=20 it is over 23,000,000. So the reasonable number of critical vertices is around 10, naturally depending on computing resources and efficiency function.

3.4 Generating Routes

In order to generate all possible routes in 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 for three times. The first offset gives the first center line for machine, the second is a boundary of first operation swath, and the third is for the reversive center line.

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 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 is

$$\frac{\sum_i s_W^i}{\sum_i t_W^i+\sum_i t_T^i} \tag{2}$$

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

3.6 Selection of Best Route in Search Horizon

The efficiency function is calculated for all possible routes and the route having maximum efficiency is selected. This phase of 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, more accurate trajectory planning algorithm may be used. As stressed above, only the first segment of selected route is applied at each step of the algorithm.

3.7 Simulation of Driving

In simulation, the route to be applied in each step must be subtracted from the original region, representing field still not operated. This phase turns out to be difficult, because the shape and characteristics of 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 operational area from the original region. Treatment of special cases is needed if the offset method is not working.

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 algorithms. One is to reduce the number of vertices, by approximating the polygon with other having less vertices, and the other is to use the original polygon but merging line segments to polylines. Here the latter is applied, the former may be added in the future if needed.

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 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 wrong results, if the line segments are very short and direction change is very small in all cases.

Several different conditions were tested and the following seems to work best. Let *r* be the turning radius of machine (note: in operational state). For every vertex in polygon, a circle with radius *2r* is drawn, see figure 9. The nearest line segments in both directions of polygon are selected (bold) and the curvature estimate (direction change in certain distance) is calculated. The limit for this value is a tuning variable, in tests thirty degrees was used with *r*=10 m.

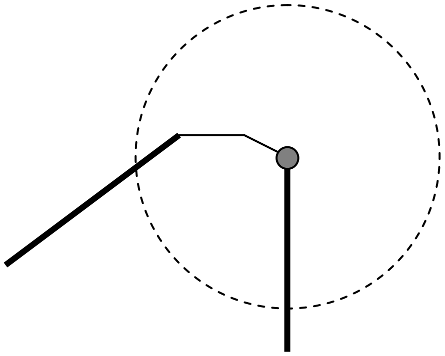


Figure 9. Merging condition.

3.9 Non-convex Fields

If the field is not convex (concave) or it contain obstacles, the region representing non-operated area will easily split into separate regions, or the region is not uniform any more. In these cases the route segment generation is similar: the region offsetting is made for all separate 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 the other must be free. This may be limited by limiting the allowed "jump points" in each region, preferably to one. The easiest, but not optimal way to overcome this problem is to restrict the path planning into a single region and squeezes them one at a time. The other, prima facie better, way is to stop the search horizon also after one jump, see the definition. This part of research is still under work, and no general conclusion is available at this phase.

3.10 Results

The algorithm has been tested with some hand-drawn fields. The algorithm works reliably for all convex fields, and for some non-convex fields that do not split into separate regions during simulation. The property to support jumps is still under development and is not functioning reliably.

The machine specific parameters in the results below have been: working width 3 m, driving speed 10 km/h in working phase, 6 km/h in minimum radius turnings, minimum turning radius 6 m at headlands and 10 m at operation. These are the only parameters needed in simulation.

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

Figure 10 presents a convex field with one long curved edge. Turning routes or driving orders are not shown. The algorithm has first driven the left side (longest) back-and-forth until it is sensible to change to squeeze swathing technique. The result is reasonable.

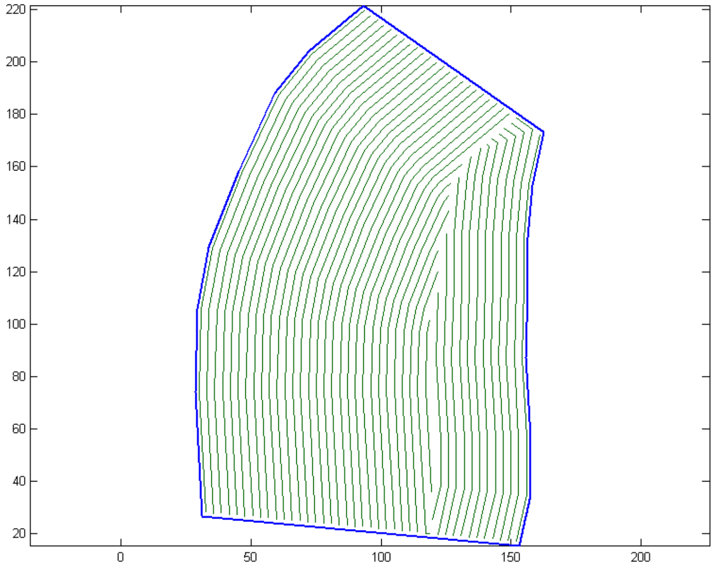


Figure 10. Field with one curve edge.

A field with no straight edges is shown in figure 11. The algorithm first drives the field around but some turnings are made back-and-forth. Also this solution is reasonable.

In figure 12, a L-shaped field is presented. For this field the algorithm first behaves so that field is driven around counter-clockwise with skipping to short edges and after nine rounds is starts to drive "wings" separately, because the turning curve length from L-outcorner to L-incorner becomes short, the solution is plausible.

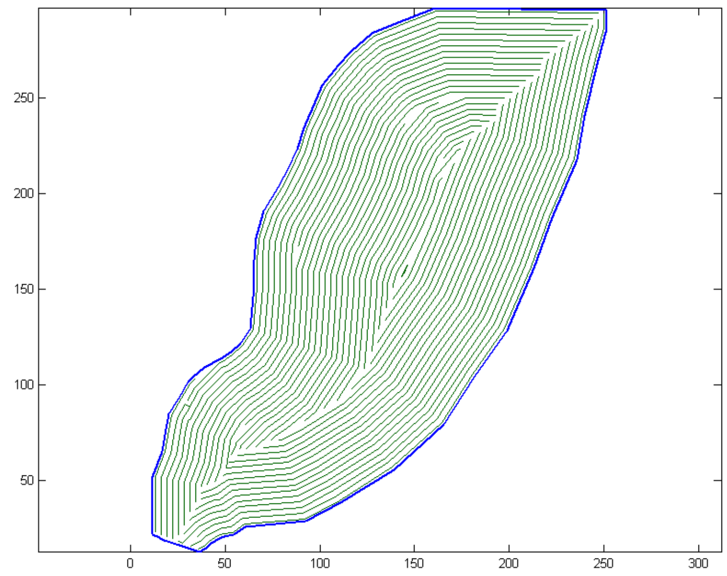


Figure 11. Field with curved edges.

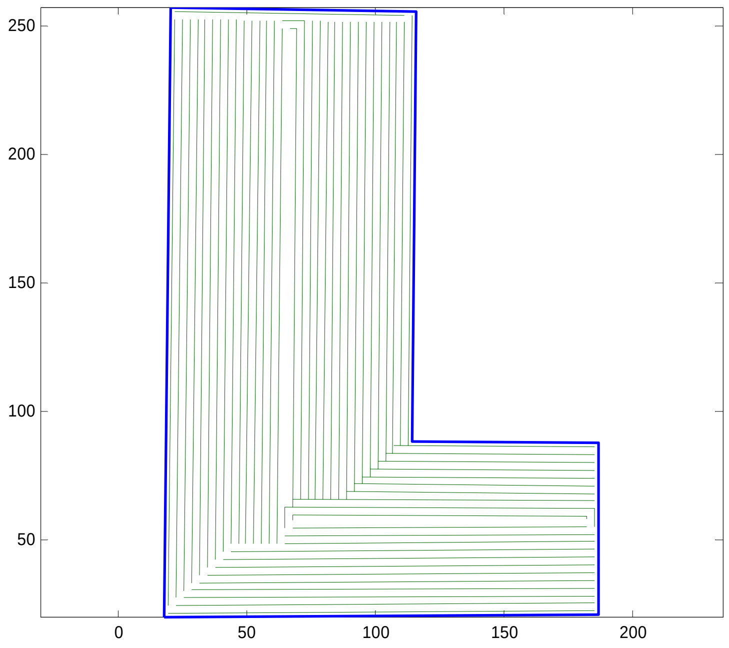


Figure 12. L-shaped field.

In figure 13 the field plot has the shape of letter T. At the beginning the simulation drives the longest edge back-and-forth until the turning to short edges parallel to long becomes quicker. Then the longest edge and the two short edges are driven around a couple of times, by skipping the gap between two short edges. After a couple of rounds the turning cost will become more or less the same between these two and simulation goes back to driving the field back-and-forth. Finally the last "bay" is driven. It may be more efficient to operate the area between two short edges instead of skipping it, but in this approach is was set a limitation that each swath must be side-by-side with some earlier one, or the edge of field.

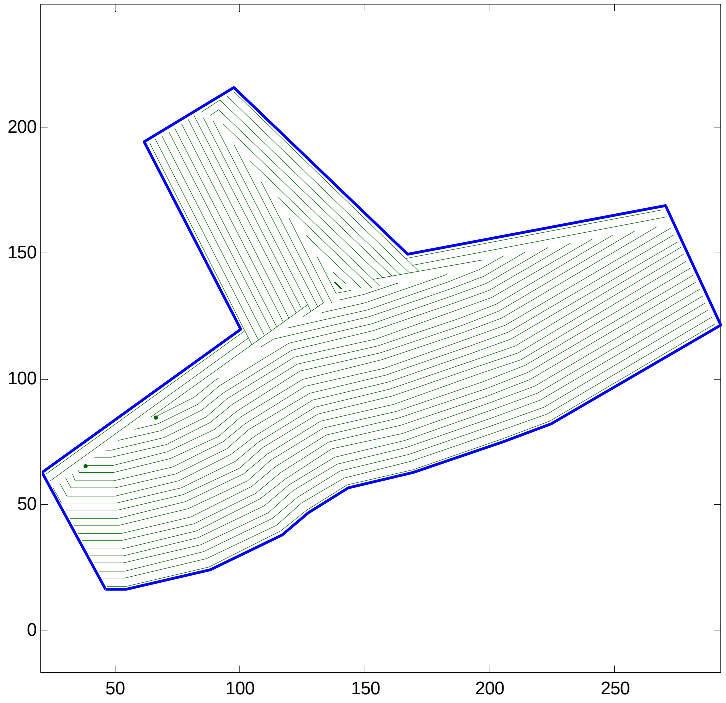


Figure 13. T-shaped field.

3.11 Speed of Algorithm

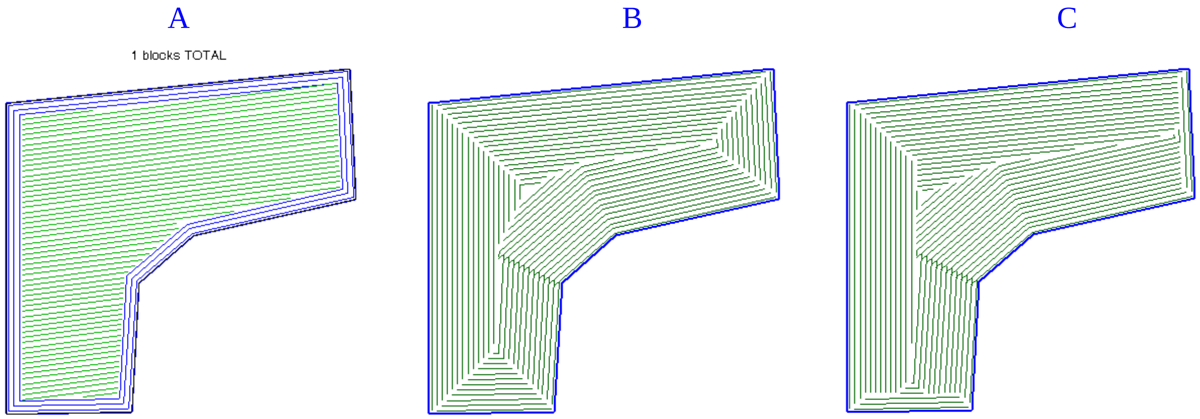
With a modern office computer (Intel P4 at 3.2 GHz), the whole calculation of L-shaped field (with 6 critical vertices) takes about 5 min, when the field area is 2.7 ha and the simulated driving in that field takes 88 min. With this complexity, the algorithm work well in real time but it cannot be guaranteed.

If this algorithm is applied in real time, the edge being driven currently should be simulated to the end, when the driver has decided it, and the search can be made for the next corner. The time to make calculation decreases as the field is operated, at least if driving around-style. For a real-time assisting system it must be noted that there is no guarantee that the calculation is finished in certain time. Therefore some heuristics must be used to either calculate first rough solution or sort the possible solutions based on some pre-known criteria (for example from earlier calculations) and after that calculate as many solution possibilities as there is time.

4. QUICK COMPARISON OF DRIVING TECHNIQUES

A quick comparison of algorithms is made in this phase. This comparison is not an accurate one, but gives some idea.

In figure 14, a boomerang shaped field is presented. On the left is a solution from the former algorithm presented in this article, in the middle is a forced simulation of circular or spiral type driving and on the right is the solution from the latter algorithm presented in this article. In case A, needed headlands are taken into consideration, in case B the headlands are created automatically, but in case C, the headlands are not required so from this is coming a little difference.



vir forc

The statistics of results in figure 14 are compared in table 1. Extra driving means the amount of additional driving distance due to turnings. This is calculated by comparing the total driving distance to the value of field area divided by working width. It can be seen that the solutions A and C will give almost equal efficiency in this field. Together with operation or mission planning, the other requirements will be decisive; for example in row crop farming the parallel lines may be better. A comparison with other simple shaped fields without obstacles gives similar results. It seems that algorithm C is a slightly better than A when the edges of field plot are curved.

Table 1. Comparison of driving techniques.

	A	**B**	**C**
Number of turnings	74	85	70
Total driving distance	12.2 km	13.5 km	12.0 km
Total driving time	1 h, 25 min, 39 s	1 h, 42 min, 41 s	1 h, 28 min, 03 s
Extra driving	46.7%	62.5%	45.0%

5. CONCLUSION

Path planning for robots working in fields is not yet solved. Various algorithms for path planning have been introduced, but they are still more like a collection of algorithms. In this article two different algorithms to solve coverage path planning problem are presented.

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 to larger blocks, using search algorithm select the best driving direction and recursing the search until the whole field has been divided. The algorithm belongs to the set of exact cellular decompositions. Trapezoidal decomposition has been utilized as a part in the algorithm. Algorithm can solve the routes for any field, with any number of obstacles and any kind of shape. In the latest version the headlands are automatically generated where needed. With prohibited regions the previous operations, underdrains and steep gradients can be taken into account. In prohibited regions certain driving directions are marked prohibited. One drawback of the algorithm is that it only can use straight driving lines. Some fields do not have straight boundaries. Especially in fields which are narrow, long and curved, the solution is far from optimal. Refilling or emptying of the machine should be included in path planning. A general usable coverage path planning algorithm should be able to adapt to agricultural task specific requirements.

The other new algorithm presented in this article uses bottom-to-top approach. Due to limitations set, the algorithm is not able to find the global absolute optimal solution, but it is always suboptimal. However the global optimality is not goal itself, if it is hard to find, in most cases suboptimal, suitable solution is sufficient. Because the optimal solution is not available, it is difficult to see how near to optimal the solutions of various algorithms are. The algorithm works well for convex fields with no obstacles. The algorithm can handle field plot as complex as ten corner points, in which the actual turning is needed. A couple of simplification methods are presented in the article in order to speed up computing. Still more simplification is needed in order to support strongly non-convex fields. More heuristics is needed to struggle with the curse of dimensionality. The algorithm works currently in real time with convex and moderate shaped fields, with less than 10 critical vertices. In the future the full support for non-convex fields and obstacles should be developed and there the requirement for real-time usability becomes more important. The headlands or the turning area should be taken into consideration, so that this algorithm would be usable from the beginning of operation.

A quick comparison of driving techniques was presented. Based on this comparison, nothing general can be concluded. A more comprehensive study is required and probably it will show that one algorithm works better for some field shapes and the other for another kind of shapes. As it is practically impossible to find an absolute optimal route with all kinds of machines and with all kinds of field plots, the problem must be restricted in some direction. In the real world several algorithms for path planning may be tried offline and the best overall solution is applied.

6. ACKNOWLEDGEMENTS

This research has been founded by Graduate School of Academy of Finland in Electronics, Telecommunication and Automation, GETA.

7. REFERENCES

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

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

Choi, H.I., C.Y. Han, S.W. Choi, H.P. Moon, K.H. Roh, and N.-S. Wee. 2001. *Two-dimensional O®sets via Medial Axis Transform I: Mathematical Theory.* Preprint. Available at: http://newton.kias.re.kr/~swchoi/homepage/offset_theory.pdf. Accessed 17 May 2006.

Choi, H.I., C.Y. Han, S.W. Choi, H.P. Moon, K.H. Roh, and N.-S. Wee. 2001. *Two-dimensional O®sets via Medial Axis Transform II: Algorithm.* Preprint. Available at: http://newton.kias.re.kr/~swchoi/homepage/offset_alg.pdf. Accessed 17 May 2006.

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

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

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

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

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.

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

Oksanen, T., and A. Visala. 2004. Optimal control of tractor-trailer system in headlands. *Proceedings of ASAE International Conference on Automation Technology for Off-road Equipment*, Kyoto. Japan, pp. 255-263.

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

Reid, J.F. 2004. Mobile Intelligent Equipment for Off-road Environments. *Proceedings of ASAE International Conference on Automation Technology for Off-road Equipment*, Kyoto. Japan, pp. 1-9.

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

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