RECURSIVE ONLINE PATH PLANNING FOR AGRICULTURAL MACHINES

Timo Oksanen[1], Arto Visala[2]

ABSTRACT

If the shape of field plot 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. In this paper a new algorithm to solve the path planning problem is presented. This algorithm 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 paper.

KEYWORDS. Path planning, mission planning, algorithms, field operations, simulation, field traffic, optimization, field machines, field robots.

INTRODUCTION

Agricultural field robots need automatic mission planning, and path planning is on of the key tasks in mission planning, see Oksanen et al 2006.

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 Sorensen et al. (2004) a method for optimizing 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. Oksanen et al (2005) presented an split and merge based algorithm to solve higher level planning problem for complex shaped fields. As the 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.

[1] TKK Helsinki University of Technology, Automation Technology laboratory. Research Scientist, M.Sc.(tech), timo.oksanen@tkk.fi

[2] TKK Helsinki University of Technology, Automation Technology laboratory. Professor, D.Sc.(tech)

DEFINITIONS

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 adaptivity, 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.

THE ALGORITHM

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 et al. (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.

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 et al. (1993), Choi et al. (2001a) and Choi et al. (2001b). Here it is used a *straight skeleton* method, (Aichholzer et al. 1995) and (Felkel et al. 1998). The algorithm can handle convex regions as well as holes in the region.

The basic idea

Lets consider a simple field, shape of rectangle, see Figure 1. 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 a critical vertices. As the limitation 1 was set, a restricted set of movements exists. The problem is to search the best route.

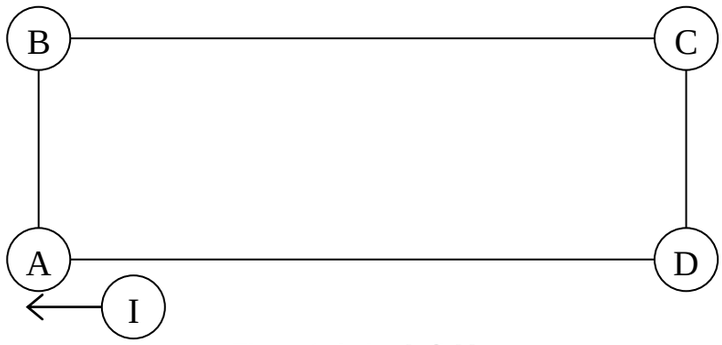

Figure 1. A simple field

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 exists 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 paper.

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', offsetted 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

Lets get back to our simple field, Figure 1. 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 direction in 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 will lead 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 12288 and for N=20 it is over 23 million. So the reasonable number of critical vertices is around 10, naturally depending on computing resources and efficiency function.

Generating routes

In order to generate all possible routes in search horizon, the polygon offsetting algorithm is utilized. The region boundary is offsetted 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.

The 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.

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.

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 offsetted 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.

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 2. 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.

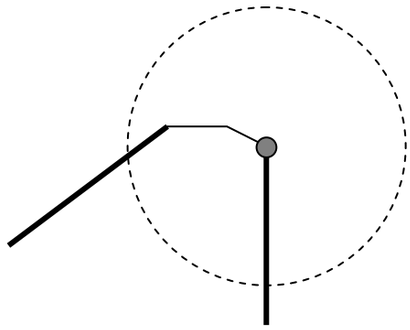

Figure 2. Merging condition

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 single region and squeeze 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 avalaible at this phase.

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 3 presents a convex field with one long curved edge. Turning routes or driving order 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.

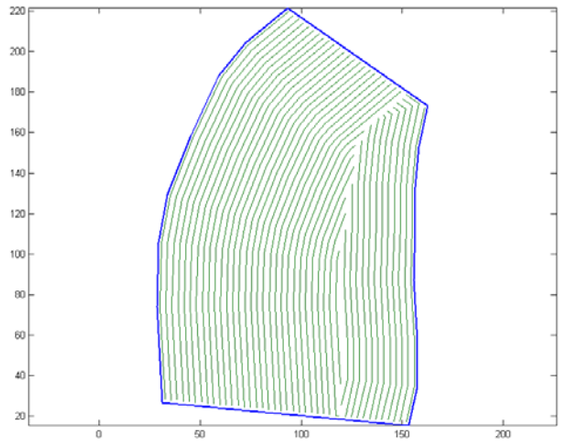

Figure 3. Field with one curve edge

A field with no straight edges is shown in Figure 4. The algorithm first drives the field around but some turnings are made back-and-forth. Also this solution is reasonable.

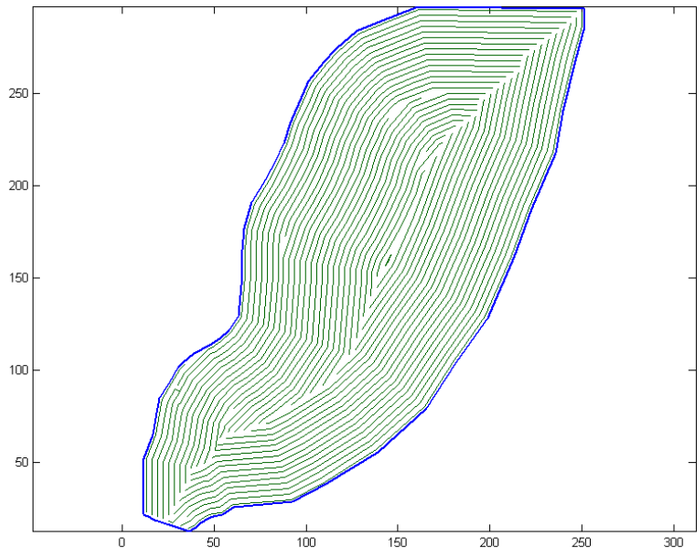

Figure 4. Field with curved edges

In Figure 5 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.

In Figure 6 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.

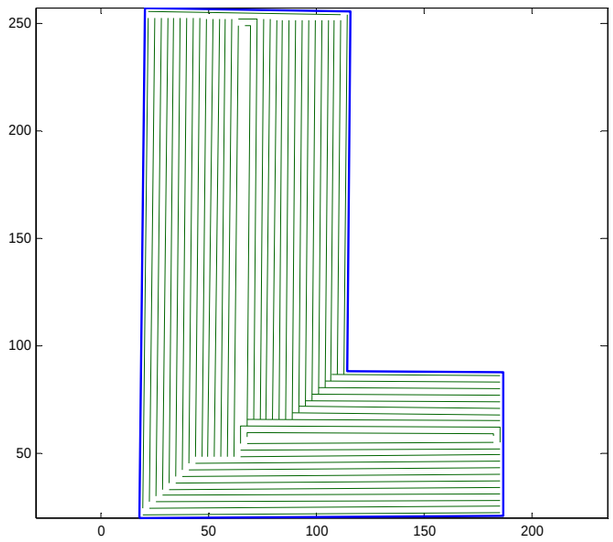

Figure 5. L-shaped field.

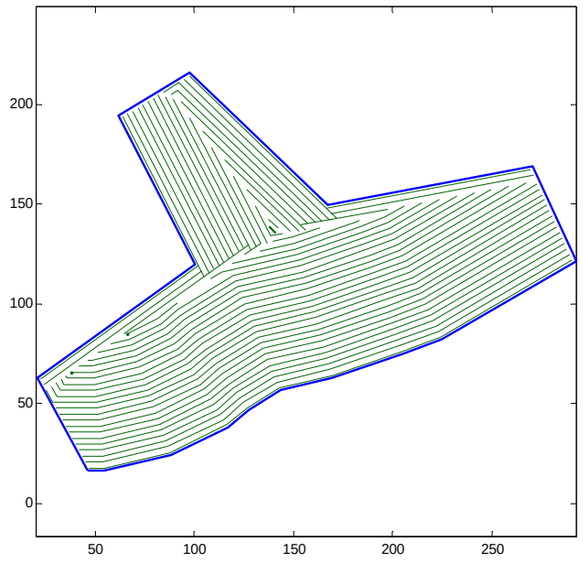

Figure 6. T-shaped field

Speed of algorithm

With a modern office computer (P4@3.2GHz), the whole calculation of L-shaped field (with 6 critical vertices) takes about 5 minutes, when the field area is 2.7 hectares and the simulated driving in that field takes 88 minutes. 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 preknown criteria (for example from earlier calculations) and after that calculate as many solution possibilities as there is time.

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 7 a boomerang shaped field is presented. On the left is a solution

from the algorithm presented in (Oksanen et al. 2005), in the middle is a forced simulation of circular or spiral type driving and on the right is the solution from the algorithm presented in this paper. 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.

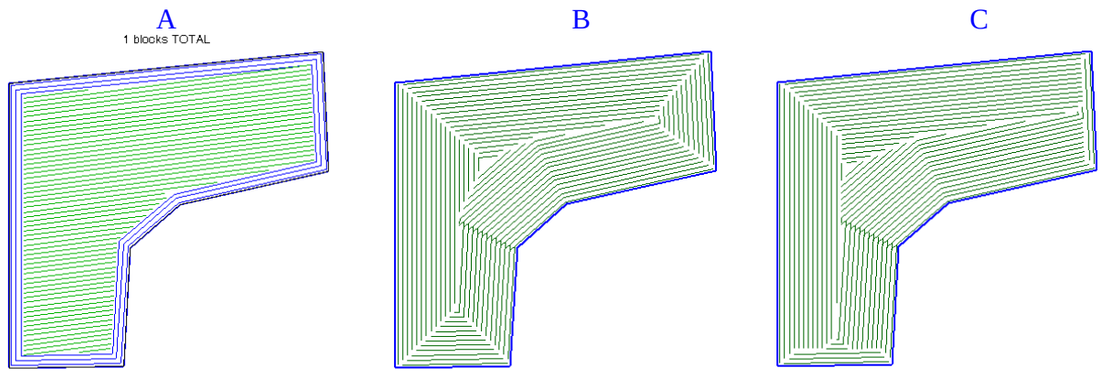

Figure 7. Comparison of driving techniques: split&merge, forced rounding, this algorithm

The statistics of results in Figure 7 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 give 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	1h 25m 39s	1h 42m 41s	1h 28m 03s
extra driving	46.7 %	62.5%	45.0%

CONCLUSION

In this paper a new algorithm for coverage path planning, especially for field operations, is presented. 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 paper 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 cannot 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.

REFERENCES

1. Acar, E. U, Choset, H., Rizzi, A. A., Atkar, P. N., Hull, D. 2002. Morse decompositions for coverage tasks. *The International Journal of Robotics Research.* 21(4): 331-344.
2. Aichholzer, O., Aurenhammer, F., Alberts, D. and Gärtner, B.. 1995. A Novel Type of Skeleton for Polygons. *Journal of Universal Computer Science,* 1 (1995):752-761.
3. Choi, H.I., Han, C.Y., Choi, S.W, Moon, H.P, Roh, K.H. and Wee N.-S. 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.
4. Choi, H.I., Han, C.Y., Choi, S.W, Moon, H.P, Roh, K.H. and Wee N.-S. 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.
5. Choset, H. 2001. Coverage for robotics - A survey of recent results. *Annals of Mathematics and Artificial Intelligence.* 31:113-126.
6. Felkel, P. and Obdrzalek, S. 1998. Straight Skeleton Implementation. *Proceedings of Spring Conference on Computer Graphics*, Budmerice, Slovakia. pp. 210-218.
7. Gray, S. A., 2001. Planning and replanning events for autonomous orchard tractors. Master's thesis, Utah State University, Logan, Utah.
8. Latombe, J.C. 1991. *Robot Motion Planning.* Boston, MA.: Kluwer Academic Publishers.
9. Maciejowski, J.M. 2002. *Predictive Control with Constraints*, Prentice Hall, England.
10. Murphy, R. R. 2000. *Introduction to AI Robotics.* A Bradford Book, The MIT Press.
11. Oksanen, T., Visala, A. 2004. Optimal control of tractor-trailer system in headlands. *ASAE International Conference on Automation Technology for Off-road Equipment*, Kyoto, Japan, pp. 255-263.
12. Oksanen, T., Kosonen, S., Visala, A. 2005. Path planning algorithm for field traffic. *ASAE Annual Meeting 2005.* ASAE Paper No. 053087.
13. Oksanen, T., Visala, A. 2006. Split and Merge Based Path Planning for Agricultural Machines. *ASABE International Conference on Automation Technology for Off-road Equipment*, Bonn, Germany.
14. 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.
15. Palmer, R., Wild, D., Runtz, K. 1988. Efficient path generation for field operations. Dept. of Computer Science, University of Regina.
16. Sørensen, C. G., Bak, T., Jørgensen, R. N. 2004. Mission planner for agricultural robotics. *AgEng 2004*, Leuven, Belgium. 8pp.
17. Yang, S.-N. and Huang, M.-L. 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.