Pattern Dependent Optimized Mowing of Football Fields with an Autonomous Robot

Tahir Majeed
Department of Informatics
Lucerne University of Applied Sciences and Arts
Lucerne, Switzerland
email: tahir.majeed@hslu.ch

Ramón Christen
Department of Informatics
Lucerne University of Applied Sciences and Arts
Lucerne, Switzerland
email: ramon.christen@hslu.ch

Michael Handschuh
Department of Informatics
Lucerne University of Applied Sciences and Arts
Lucerne, Switzerland
email: michael.handschuh@hslu.ch

René Meier
Department of Informatics
Lucerne University of Applied Sciences and Arts
Lucerne, Switzerland
email: rene.meier@hslu.ch

***Abstract*—This paper addresses the football field mowing problem using an autonomous mowing robot. To reduce the cost of maintaining and preparing the football field for a professional match, the football field is mowed using an autonomous robot. Football fields are typically mowed according to a pattern; therefore, the mowing path of the robot must be optimized with respect to time while obeying the constraint that mowing should result in a predefined pattern. In addition to finding the optimized path for the autonomous robot, a planning software has also been developed that creates and provides the data required by the optimizer. This data contains information on the pattern, the dependencies between individual lanes and the time required to mow a lane and to transition between lanes. The mathematical model to find the optimal mowing path was developed using Integer Programming. The proposed model is computationally efficient, mows the required pattern, fulfills the dependency constraints between the lanes and optimizes the path of the robot so that the football field is mowed in the least possible time.**

***Index Terms*—mowing robot, path optimization, discrete optimization, integer programming.**

I. INTRODUCTION

Football is a famous sport which is played and liked by billions over the world. Many people like to watch a match in a stadium while others prefer watching it from the comfort of their home and in front of their television screens. Before professional matches can take place, a lot of planning and resources are required to properly prepare the football field. Fédération Internationale de Football Association (FIFA) defines a set of standards [1] that every ground must meet. Among other requirements, FIFA standards specify: the maximum length of the grass on the football field; an area of at least 3 m around the playing field must be obstacle free for the safety of the players; the minimum and the maximum allowed width of the playing field; rules relating to the luminosity of the football field; and that every part of the field should be lit. All these rules can be found in the FIFA manual [1].

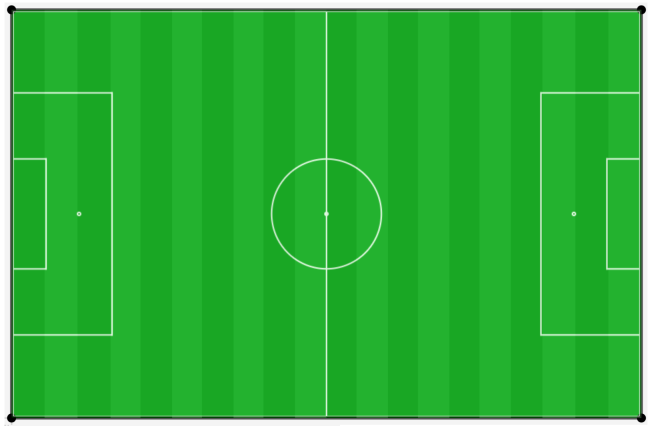


Figure 1: Football field with mowed pattern.

A pattern is mowed in the football field [2] as shown in Figure 1 to make the field aesthetically pleasing [3] for the audience sitting in the stadium or watching on television screens. Mowing the pattern is not a requirement of the FIFA standards, however, the standards do define the maximum height of the grass on the field. It is the responsibility of the ground manager to prepare the football field before the start of a match. On the orders of the ground manager, the grass is mowed and during the mowing simultaneously a pattern is mowed in the field. This practice is traditionally followed in all stadiums around the world. Mowing the pattern does not require any special or additional equipment. The pattern can be easily mowed by folding the grass one way or another, which in turn depends on the direction of the mowing vehicle (either top to bottom or bottom to top).

To comply with the FIFA defined regulations and guidelines requires the employment of a large human workforce. This increases the cost of maintaining the ground significantly. A human driven combustion engine grass mowing vehicle weighs [4] around 530 kg to 835 kg. This weight can destroy the grass; therefore, it is desirable that the weight of the vehicle be reduced. When the mowing vehicle is driven by a human, the additional weight of the human makes this even worse. This can be remedied by utilizing an automatic mowing robot. This results in avoiding the additional weight of the human driver and the combustion engine.

An autonomous mowing robot has been proposed to reduce the cost of maintaining the football field according to FIFA rules and regulations while simultaneously reducing the weight of the vehicle. The idea is to upload the pre-optimized path in the vehicle's on-board memory, which allows the autonomous mower to mow the field in the minimum time. This paper presents a novel approach for calculating the optimal mowing path for the autonomous mowing vehicle using integer programming. The contribution of this paper is the creation of an integer programming model for searching the optimal mowing path with respect to mowing time and additionally fulfilling dependency constraints.

The mowing problem is casted as discrete optimization problem, where the football field is to be mowed according to the given pattern in the minimum amount of time. Optimizing the mowing path of the football field is similar to a "Traveling Salesman Problem (TSP)" [5] and a "Lawn mowing & milling" problem [6]. TSP has been one of the classical and extensively studied problems in discrete optimization. In a TSP problem, a number of cities have to be visited by a salesman starting from a specific city and returning to the starting city covering minimum distance [7]. Many different approaches have been proposed for solving TSP which includes meta-heuristic approaches, for example, genetic algorithms [8], evolutionary algorithms [9][10], tabu search [11], simulated annealing [12][13], ant colony optimization [14] and integer programming [15][16]. Bektas [5] has given an overview of different integer programming formulations for TSP.

In a lawn mowing & milling problem, a given region must be mowed using a specified mowing blade shape. The region must be mowed with respect to some defined minimization criteria [6]. Generally, the turn cost is minimized [17] which implies that the region should be mowed in minimum number of 90° turns. This kind of problem arises in various domains of our daily life, such as spray painting, geographic surveys, engravings, and drones sweeping regions [6][18]. The lawn mowing and milling problems is in general NP-hard [19]. Arkin *et al.* [6] has proposed a solution for the lawn mowing problem by dividing the mowing region into a set of connected polygons and then using geometric TSP approach to find the feasible solution. Arkin *et al.* [17] proposes to use the geometric TSP approach together with minimizing the turn cost to find feasible solution. Fekete *et al.* [18] extends upon the work of Arkin *et al.* [6][17] to computes a minimum-turn cycle cover for a given region using integer programming based approximate solution.

The problem addressed in this paper is based on earlier work of the authors on a different problem [16]. The domain of the problems addressed in this paper is completely different from the one addressed in [16]. A major contribution of this paper is in formulating a lawn mowing milling problem into a dependency based workflow problem. The requirement to mow a pattern allowed to use the dependency based integer programming formulation of [16]. Furthermore, the natural constraints between the lanes had been transformed into a dependency based constraints. This allowed to use the structure and formulation of the dependency based workflows. The basic structure of the problem addressed in this paper is TSP with dependencies that should be obeyed, however, there are no multiple TSP as solved by Majeed *et al.* [16]. The problem addressed in this paper is also related to the general mowing problem with additional constraints and dependencies. Instead of minimizing the turn cost as normally used in the mowing problem, the objective function minimized in this paper is the total mowing time as given by 1. A similar strategy to the one proposed in this paper has been suggested by Arkin *et al.* [6] but the rectangular lane structure arises naturally from within the problem itself due to the pattern that should be mowed in the field.

The requirement to optimize the mowing time generates from the autonomous robot manufacturing industry. Furthermore, from the mathematical perspective, it is also important that the mowing time is optimized. If the model optimizes the number of turns then there is no constraint that restricts the mower taking a lane 10- or 20-lanes away instead of taking a lane which is 2-lanes away. The turn cost will be the same for both if either a faraway lane or a lane closer was chosen. The mowing time will be vastly different for the both aforementioned cases. As there are equal number of top- and bottom-entering lanes or at maximum can differ by one lane, minimizing the turn cost is not relevant. As the autonomous mowing robot will generally be battery powered, therefore, it is also relevant that the mowing time is optimized.

The paper is organized as follows: Section II provides details of the mowing planning software that generates and provides the input data for the optimization model. Section III provides the details on the mathematical formulation and the optimization model. The settings and the results of our evaluation are presented in Section IV and, finally, Section V provides conclusion and the possibilities on extending the model to solve a larger class of optimization problems.

II. Football Field Planning Software

Optimizing a mowing track requires the knowledge of the relevant environment. Captured in a manner accessible for optimization algorithms, a model representing the area containing information about the surface of the ground, the dimension of the football field, as well as additional navigation areas or obstacles is required. The planning software models the football field and identifies the location and size of the obstacles that must be avoided by the autonomous mowing robot. It further captures information on the mowing lanes, the pattern to be mowed in the football field, the time required to mow a lane and the time required to transition between the lanes.

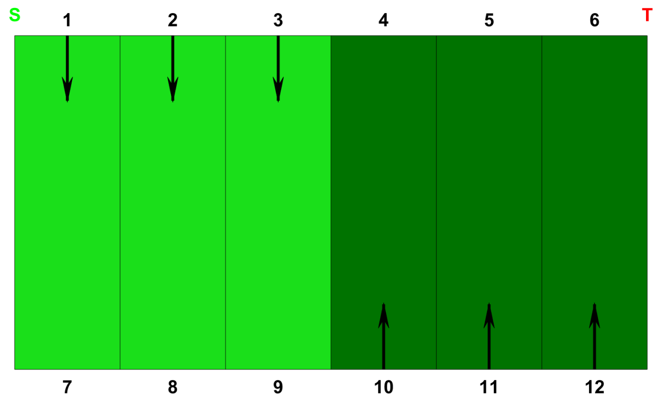


Figure 2: The figure shows a football field pattern with an equal number of top and bottom-entering lanes.

With regard to the calculation of a mowing track containing coordinates for the navigation in a second order space, the model is mapped in a two dimensional vector space containing multiple polygons assigned to one of the distinguished navigation and mowing areas, respectively: pitch (mowing area with pattern restriction), as can be seen in Figure 1 created using the developed software. As input for the model serve the real measured distances and properties of surfaces and objects.

The variation of patterns for the football field ranges from simple lanes in vertical or horizontal orientation to a mixture of vertical and horizontal lanes to complex graphics and logos. However, since our mowers are equipped with spindle blades, our work focuses on patterns based on combinations of straight lanes because of the mower's limited radial mowing ability. The football field will be sub-divided into mowing lanes whose width must not exceed the width of the mower cutting blade. The mowing lanes serves as the smallest unit that must be mowed and the order of the lanes mowed is to be determined by the optimizer. The football field setting and pattern to be mowed are captured by the developed software. The software then computes the mowing lanes. The width of the mowing spindle is $0.85\ m$, to ensure that the grass at the edge of the mowing spindle is properly cut, the mowing lanes are overlapping with a width of $10\ cm$. Using this information, the software divides the football field into a series of mowing lanes and saves the information on which lanes can be entered from the top and which should be entered from the bottom to get the desired pattern. The direction of the mowing is important, that is, either top-to-bottom or bottom-to-top because this produces the light and the dark shades in the grass. The direction of the mowing makes the grass fall one way or the other, which results in grass exhibiting the light or the dark shade.

It should be observed that the pattern mowed in the field is an opposite reflection (not mirroring) across the center line of the football field, therefore, the total number of top-entering lanes is equal to the total number of bottom-entering lanes. This is an important observation which is used in the construction of the optimization model. The optimization model works for an equal number of top- and bottom-entering lanes. The optimization model can also optimize a model with a maximum difference of one lane between the top- and bottom-entering lanes with a minor restriction that can be handled in the planning software as will be explained below.

The modeled environment and the applied pattern in the software results in two matrices called transition matrix $\mathbf{E}$ and lane dependency matrix $\mathbf{D}$ containing the input information for the optimization. The details of the two matrices are illustrated using the basic example shown in Figure 2, where vertical mowing lanes are considered. The mowing lanes are the smallest unit that can be mowed by a mower and the task of the optimizer is to decide the order in which the mowing lanes are to be mowed. The direction of the arrows defines the mowing direction. The mower is only allowed to enter a lane in the direction of the arrow.

For the purpose of the mathematical modeling each mowing lane is defined by two end points, Top-Point (TP) and Bottom-Point (BP) so named as they appear in the figure because of vertical mowing lanes. To get the desired pattern, some mowing lanes can only be entered from the top (shown in Figure 2 by down pointing arrows in lanes 1, 2 and 3) while other mowing lanes can only be entered from the bottom (shown in Figure 2 by up pointing arrows in lanes 10, 11 and 12). The pattern is correctly mowed in the field if the mower moves from point 1 to point 7 with blade down. This will fold the grass in the correct direction while the opposite pattern can be correctly mowed in lane if the mower moves from point 10 to point 4. The same applies to the remaining mowing lanes. There are two additional special points called Start-Point S and Terminal-Point T. S is denoted by numerical index 0 while T is denoted by numerical index given by the equation $T = 2N + 1$. In the example under consideration $N = 6$, therefore, $T = 13$ as can be seen in Figure 2. Generally, the notations S and T are used instead of their numerical index in this paper. The TP of each lane are numbered from 1 to N, while the BP of each lane are numbered from $N + 1$ to $2N$. For the example under consideration the TP are numbered from 1 to 6 while the BP are numbered from 7 to 12 as shown in Figure 2. Let n be the total number of points in the model where $n = 2N + 2$.

A transition cost matrix $\mathbf{E}$, $\mathbf{E} \in \mathbb{R}^{n \times n}$ defines all possible transitions from a top and bottom point of a mowing lane to all other points as shown in Table I. The matrix $\mathbf{E}$ is the cost matrix for the transition of the mower between different lane points. The time required to move from one point to another point is used as the cost. The travel time between two points depend on the velocity, turn angle and acceleration. Matrix $\mathbf{E}$ matrix can be divided into four sections as can be seen in Table I. Each section of the $\mathbf{E}$ defines a different possibility.

In the example shown in Figure 2 and Table I, suppose that the grounds manager chose the first lane to be the starting point of the mowing. This is reflected by $e^{S,1} = 1$ in Table I. This tells the optimizer that the transition from S to lane-1 costs 1 unit while the rest of the entries in row S are all ∞ cost. All other transitions from S to any other lane have an

Table I: MATRIX E, COST MATRIX PROVIDING MOWER TRANSITION TIME (MILLI-SECONDS) BETWEEN LANES.

	S	1	2	3	4	5	6	7	8	9	10	11	12	T
S	∞	1	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
1	∞	∞	166	333	499	642	784	1000	∞	∞	∞	∞	∞	∞
2	∞	166	∞	166	333	475	618	∞	1000	∞	∞	∞	∞	∞
3	∞	333	166	∞	166	309	451	∞	∞	1000	∞	∞	∞	∞
4	∞	499	333	166	∞	142	284	∞	∞	∞	500	∞	∞	1
5	∞	642	475	309	142	∞	142	∞	∞	∞	∞	500	∞	1
6	∞	784	618	451	284	142	∞	∞	∞	∞	∞	∞	500	1
7	∞	500	∞	∞	∞	∞	∞	∞	166	333	499	642	784	1
8	∞	∞	500	∞	∞	∞	∞	166	∞	166	333	475	618	1
9	∞	∞	∞	500	∞	∞	∞	333	166	∞	166	309	451	1
10	∞	∞	∞	∞	1000	∞	∞	499	333	166	∞	142	284	∞
11	∞	∞	∞	∞	∞	1000	∞	642	475	309	142	∞	142	∞
12	∞	∞	∞	∞	∞	∞	1000	784	618	451	284	142	∞	∞
T	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞

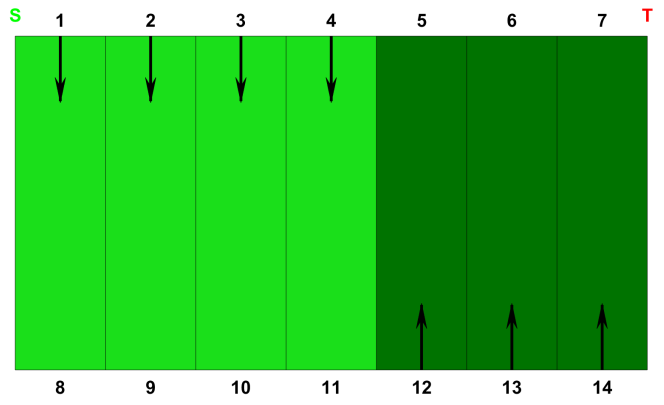


Figure 3: The figure shows an odd number of lanes with an unequal number of top entering lanes and bottom entering lanes. This example shows four top entering lanes and three bottom entering lanes.

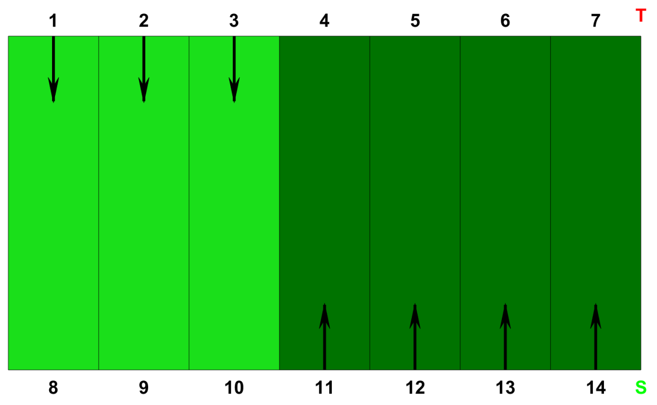


Figure 4: The figure shows an odd number of lanes with an unequal number of top entering lanes and bottom entering lanes. This example shows three top entering lanes and four bottom entering lanes.

infinite cost. The transition cost defined by $e^{i,j}$ is read as the cost of the mower's transition from point-i to point-j. Defining an infinite cost for something has the effect of defining a hard constraint and disallowing such transitions. In the example under consideration the mower is allowed to move only from S to lane-1. This is not a restriction of the model. The model itself allows any of the lanes to be chosen or the optimizer can select any one of the lanes if the infinite cost is replaced by 1 for row S. The cost used in matrix $\mathbf{E}$ reflects the real-world domain knowledge extracted from the domain expert.

From the perspective of the optimization model any of the lanes can be chosen to be the starting lane in cases where the top entering lanes are equal to the bottom entering lanes. However, in cases where the top and bottom entering lanes differ by 1, that is, either there are 4 top entering lanes and 3 bottom entering lanes (see Figure 3) or 3 top entering lanes and 4 bottom entering lanes (see Figure 4), in such cases the model defines the starting point to be on the side with more lanes as shown in the corresponding figures. If there are 4 top entering lanes and 3 bottom entering lanes, the starting point S must be located on top which has one more lane than the other side. Similarly, if there are 4 bottom entering lanes and 3 top entering lanes then the starting point S must be located at the bottom. Our model has no need to be able to handle lane difference of 2 or more lanes and hence, we omit a discussion of such a requirement.

Once the mower has left S, it is forbidden to go back to S, therefore, the column S is filled with ∞ cost. It is also forbidden to go back to the point itself, that is, it makes no sense physically to allow the mower to go from point 1 to point 1 itself, therefore, all the self-points are tagged with ∞ cost on the diagonal. Once the mower reaches T, it is not allowed to move to any other point, therefore, row T is assigned ∞ cost.

If a mower is at point 1, there are two possibilities to transition. The first possibility is that the mower can either transition to another top-point (which could be any of the points 2, 3, 4, 5, and 6) since point 1 itself is a top-point and the second possibility is that it can transition to the bottom-point of its lane which in this case is point 7. The transition time to another top-point can be computed reliably using the mower speed and acceleration equations. The transition time from 1 to 7 can also be reliably computed using the same set of equations. It should be noted that transition time from point

1 to point 7 is mowing the lane in the direction of the pattern. Transitioning from top-point 1 to any other bottom-point other than 7 ($|t \pm N|$) is disallowed by the model with infinite cost in row 1 of the Table I. Transitioning from a top point to a bottom point is depicted in the top-right section of the Table I. The same is repeated for top-points 2, 3, 4, 5 and 6 which fills the top left section of the Table I. As the mower is moving from point 1 to point 7 in the direction of the pattern, therefore, it will be moving with its blade down. When the blade is down the grass is mowed and simultaneously the pattern is mowed in the grass. When the blade is down the speed of the mower reduces by half compared to moving with blade up. This can be seen in the table with the cost $e^{1,7} = 1000$ compared to $e^{4,10} = 500$.

From top-point 4 the mower can transition to top points 1, 2, 3, 5, 6 or it can move in the direction of the bottom-point 10 with mowing blade up. The direction of the mowing for the lane 4-10 is from point 10 to point 4 (as suggested by the arrow from 10 to 4 in Figure 2), therefore, if the mower moves from point 4 to point 10 the mower is moving in the opposite direction of the desired pattern. If the mowing blade is up, a mower can move through a lane without disturbing the underlying pattern. When the mower moves in the direction opposite to the pattern then it does so by having the mowing blade up, which will leave the pattern unchanged on the ground and allows the mower to move at twice the speed.

Transitions between the bottom points 7, 8, 9, 10, 11 and 12 fills up the bottom right section of the Table I while the transition from the bottom point to a top point fills up the bottom left section of the Table I. This is how the whole cost matrix $\mathbf{E}$ is constructed. The mowing terminates when the mower reaches the terminal point T. The mower will only move to the terminal point when all the lanes have been mowed as shown below when the constraints are presented. The terminal point can only be reached from bottom-points 7, 8, 9 and top-points 4, 5, 6. This is shown in the $\mathbf{E}$ matrix by assigning 1 to the specific entries of column T, while all other entries have been assigned ∞ cost. Allowing the mower to move from a specific set of points to terminal points has been derived from domain knowledge.

The dependency matrix $\mathbf{D}$ contains the dependency information between end points of the mowing lanes. A Boolean value serves the information if a start or an end point depends on another, which means that the mower must pass a point before or after the other. As the dependency information is given at the mowing line end point level, it also contains the information about the mowing direction for the desired pattern.

The optimizer returns a sequence of line start and end points defining a track in an abstract manner. This information needs to be added to the autonomous mowing robot so that it can follow the optimized path. For navigation and for a proper execution of the mowing task by the autonomous mower, the optimized track information needs to be enriched with application related information. The abstract track information (in the form of track end points) is transformed into a series of track points by a software developed by the authors. These track points are $0.04\ m$ apart and are enriched by additional information that maps every track point to the two-dimensional mowing area in the field and adds orientation, speed, and blade information to each track point. The distance from an end point of a track segment to the next is between zero and the preset distance. There are some adjustments that need to be made while computing the track points because of the obstacles that are outside the playing field and $3\ m$ obstacle free area around the playing field. The obstacles are initially not considered while computing the optimized mowing path. However, it is not difficult to add the obstacle information to the optimized model. Currently, the time to transition from one lane to another lane is assumed to be obstacle free. This obstacle free transition time is added to the cost matrix $\mathbf{E}$. Replacing the obstacle free transition time with the one that considers the obstacle will enrich the model that considers the obstacles as well.

III. The Integer Programming Model

Our football field mowing problem consists of a set of lanes $\mathbf{J} = \{j^1_{1,7}, j^2_{2,8}, \ldots, j^N_{u,v} u := \{1 \ldots N\}\}; v := u + N$ with N lanes. Each lane has two end-points given by the subscripts $j_{u,v}$. Let $\mathbf{D}$ be a binary 2D matrix of mowing lanes which represents the dependencies between the lanes. If a mowing lane $j_{u,v}$ which is a bottom entering lane must be mowed before $j_{u',v'}$ which is a top entering lane. As $j_{u,v}$ is a bottom entering lane then point v must be mowed before u and similarly if $j_{u',v'}$ is a top entering lane then $u^{'}$ must be mowed before $v^{'}$. The set of dependencies that must be encoded are $u \perp\!\!\!\perp v$, $v^{'} \perp\!\!\!\perp u^{'}$, $v^{'} \perp\!\!\!\perp v$ and $u^{'} \perp\!\!\!\perp u$. The symbol $a \perp\!\!\!\perp b$ is read as *a depends on b*. The encoded dependencies are read as; $u \perp\!\!\!\perp v$ states that u must be mowed before v, $v^{'} \perp\!\!\!\perp u^{'}$ states that $v^{'}$ must be mowed before $u^{'}$, $v^{'} \perp\!\!\!\perp v$ states that $v^{'}$ must be mowed before v and $u^{'} \perp\!\!\!\perp u$ states that $u^{'}$ must be mowed before u.

The dependencies listed in the last paragraph are encoded in the dependency matrix given by Table II and its corresponding Figure 2. Figure 2, shows that lane $j^3_{3,9}$ is a top entering lane while the lane $j^4_{4,10}$ is bottom entering lane. The information of which lanes are top entering lanes and which lanes are bottom entering lane is also encoded in the dependency matrix $\mathbf{D}$. In $\mathbf{D}$, the entry $\mathbf{D}(4, 10) = 1$ which specifies that $4 \perp\!\!\!\perp 10$ which basically states that the 10 must be mowed before 4 thus lane $j^4_{4,10}$ is a bottom entering lane. Similarly, the entry $\mathbf{D}(9, 3) = 1$ which specifies that $9 \perp\!\!\!\perp 3$, therefore, 3 must be mowed before 9 thus lane $j^3_{3,9}$ is a top entering lane. To encode the dependency between the opposite direction lanes $\mathbf{D}(10, 9) = 1$ which means $10 \perp\!\!\!\perp 9$, therefore, 9 must be mowed before 10 and $\mathbf{D}(4, 3) = 1$ which means $4 \perp\!\!\!\perp 3$, that is, 3 must be mowed before 4. All these dependencies ensure that lane $j^3_{3,9}$ is mowed before lane $j^4_{4,10}$. The zero entries in $\mathbf{D}$ specifies independence relationship between the lane points. The optimizer is free to choose in which order these lane points are mowed. The underlying dependency graph is a directed acyclic graph which ensures that there are no circular dependencies. These direct dependencies between the lanes are

Table II: MATRIX D

	S	1	2	3	4	5	6	7	8	9	10	11	12	T
S	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1	0	0	1	0	0	0	0	0	0	1	0	0	0
5	1	0	0	0	0	0	0	0	0	0	0	1	0	0
6	1	0	0	0	0	0	0	0	0	0	0	0	1	0
7	1	1	0	0	0	0	0	0	0	0	0	0	0	0
8	1	0	1	0	0	0	0	0	0	0	0	0	0	0
9	1	0	0	1	0	0	0	0	0	0	0	0	0	0
10	1	0	0	0	0	0	0	0	0	1	0	0	0	0
11	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12	1	0	0	0	0	0	0	0	0	0	0	0	0	0
T	1	1	1	1	1	1	1	1	1	1	1	1	1	0

generally found between the mowing lanes where the pattern changes from light to dark or dark to light. The input data $\mathbf{D}$ & $\mathbf{E}$ matrices for the optimizer are obtained from the planning software as explained in Section II. Let $d_{ij} \in \mathbf{d}$ be the set of binary decision variables $d_{ij} = \{0, 1\}$ $\forall i, j = 0, \ldots, n-1$. d_{ij} is 1 if the mower takes the route from i to j and 0 if it does not take this route.

$$\text{Objective: } \min \ \mathbf{Obj} = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} d_{ij} * e_{ij}. \tag{1}$$

The following assumptions are made: the transition time between lanes is known and fix; all the dependencies between the lanes are known before the start of the optimization. To make the integer programming formulation easier, dummy nodes S and T are added which serve as the start and the end of the mowing route (see Figure 2). If a route d_{ij} is written as d_{Sj} then this denotes a route from start point S to any point j and a route d_{iT} denotes a route form any point i to terminal point T. All lanes must be visited, and they should be visited only exactly once.

To create the optimized mowing path, a mathematical model must be constructed with an objective function that defines a cost in terms of some measure (for example, time, money, distance etc.), which is minimized to obtain the optimal solution. The objective function $\mathbf{Obj}$ given by 1 can be defined for minimizing the travel time $d_{ij} * e_{ij}$ which in turn is equivalent to minimizing mowing time of the football field. In the formulation presented, the mower is not allowed to go back in the lane. In addition to optimizing the mowing time, the model also considers the dependencies of the pattern and if there are lanes that must be mowed before other lanes. Figure 1 shows an example of a mowed pattern in a field. The pattern lanes are generally 5 m in width. They are further broken down into mowing lanes by the planning software which results in a pattern similar to the one shown in Figure 5.

$$\sum_{j=1}^{n-1} d_{Sj} = 1 \tag{2}$$

$$\sum_{i=0}^{n-1} d_{iT} = 1 \tag{3}$$

The first constraint given by 2 states that the mower should move from the start point to the first lane to be mowed. This equation shows that the mower is allowed to move from starting-point S to any of the mowing lane points. Corresponding 2, the mower should move to the Terminal-Point T at the end when all the lanes have been mowed as given by 3. The next set of constraints deal with either mowing the lane and then transitioning to another lane or transition from another lane and then mowing the lane. Two sets of equations are presented for each of the top-points (see equations 4 & 5) and then for the bottom-points (see equations 6 & 7).

$$d_{j+N,j} - \left(\sum_{i=1}^{N} d_{j,i} \right) = 0 \qquad \forall j = 1 \ldots N \tag{4}$$

$$\left(\sum_{i=0}^{N} d_{i,j} \right) - d_{j,j+N} = 0 \qquad \forall j = 1 \ldots N \tag{5}$$

$$d_{j-N,j} - \left(\sum_{i=N+1}^{n-1} d_{j,i} \right) = 0 \qquad \forall j = N+1 \ldots 2N \tag{6}$$

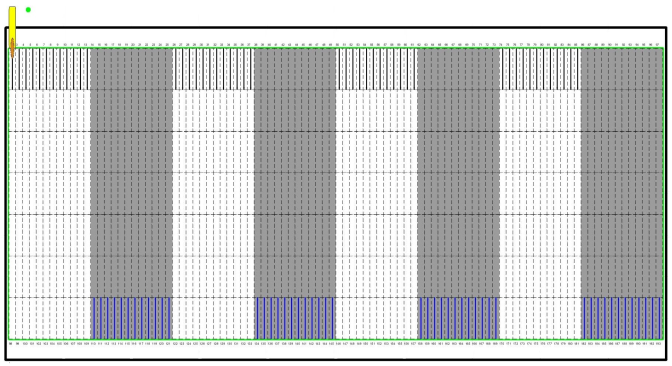
$$d_{S,j} + \left(\sum_{i=N+1}^{2N} d_{i,j} \right) - d_{j,j-N} = 0 \qquad \forall j = N+1 \ldots 2N \tag{7}$$

The next set of constraint are sub-tour elimination constraints [5]. The sub-tour constraints given by 8 are the same as suggested by Miller *et al.* [15].

$$u_{i-1} - u_{j-1} + (n-1) \times d_{i,j} < n - 2 \tag{8}$$

The dependency constraints given by the dependency matrix $\mathbf{D}$ are added to ensure that the mower mows the left lane before the right lane. This information is encoded in the dependency matrix $\mathbf{D}$ given by Table II which is added in the optimization model in the form of dependency constraints given by 9. Equation 9 ensures that the lanes are mowed in an order that satisfies the dependency matrix D. This constraint is called precedence constraint and is used in job-shop scheduling optimization community [16][20][21]. To decrease the search space and find an optimal solution as early as possible some heuristics were employed. The classical TSP assumes that the salesman can visit any site in any order. However, for the given problem the ∞ cost is assigned (see Table I) to unreasonable and unrealistic possibilities and possibilities that are disallowed due to model constraints. The corresponding decision variable $d_{i,j}$ to the ∞ cost are set to 0 during variable creation.

$$u_{j-1} - u_{i-1} < 0 \tag{9}$$


(a) 96-mowing lane filed.

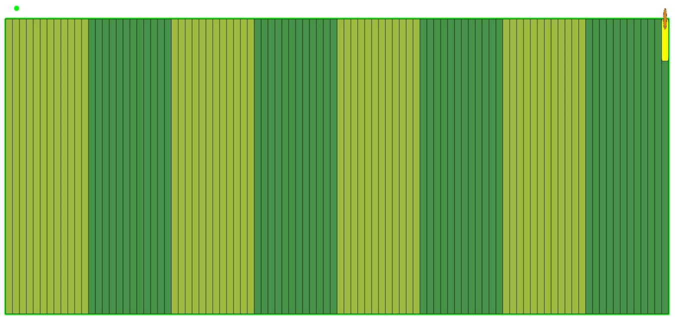

(b) Optimized pattern.

Figure 5: Optimization result.

IV. Results

Gurobi 8.0 [22] was used as the optimizer to find the optimal solution for the presented model. Java 8 was used as the programming language and it was tested on an Intel(R) Core(TM) i7-5600U CPU 4-Core 2.60 GHz and 8GB RAM running Windows 10 Enterprise 64-bit. The model was tested on different datasets created using our planning software. The results are presented in Table III. Table III shows the number of lanes (Lanes), solution optimality (Status), objective value (Obj), the gap between the upper and lower bound (Gap), and the run time (Time) of the solver until the optimal solution was found. The solver was able to prove the optimality of the found solution within seconds which shows that the model is computationally efficient. For common cases where the football field is 120 m in length and 60 m in width with a mower blade width of 0.85 m there were 160 lanes, and the model was able to optimize the path of mowing the field within seconds. If, however, the size of the model increases, for example, if the width of the mowing blade is reduced to 0.4 m then the number of lanes doubles and the computational time increases.

Table III: RESULT WITH DIFFERENT NUMBER OF LANES.

Lanes	Status	Obj	Gap(%)	Time(sec)
16	Optimal	41474	0.0	00.13
40	Optimal	231185	0.0	00.49
68	Optimal	539995	0.0	01.22
96	Optimal	2113341	0.0	00.43
146	Optimal	5166207	0.0	15.43
192	Optimal	6761169	0.0	30.56

V. Conclusion and Future Work

An optimization model for optimizing the path of an autonomous football field mowing robot has been presented in this paper. Our model enables such a robot to mow a pattern in a football field to make it aesthetically pleasing for the audience in the stadium and watching on television screens while reducing maintenance cost. The autonomous mowing robot follows the optimized path to create the desired pattern and mow the grass in minimal time.

Mowing the pattern increases the complexity of the problem and makes it different from lawn mowing robots where the purpose is to just mow the grass minimizing some cost function, such as minimizing the number of turns. The input data ($\mathbf{E}$ & $\mathbf{D}$ matrices) for the optimization model is created by our football field planning software. Mowing the field according to a pattern is what makes the problem challenging as the robot can only enter either from the top or bottom for each lane. This makes the transition between the lanes important as the mower is allowed to go over the lanes exactly once. An integer programming model has been presented that optimizes the path of the mowing robot with respect to time and the given pattern. The results of our evaluation show that the proposed model is efficient as it can compute an optimal solution within a few seconds.

The presented model can optimize models where the top entering lanes and bottom entering lanes differ by one. The proposed model assumes that there are no circular dependencies. In future work we aim to relax the restriction on the model and cover situations where the difference between the top and bottom entering lanes is more than one.

Acknowledgment

The authors would like to thank Innosuisse – The Swiss Innovation Agency for funding this work in part.

References

[1] FIFA, "Football Stadiums Technical Recommendations and Requirements," tech. rep., FIFA, 2011. Available at https://digitalhub.fifa.com/m/6c66fa18f65c0a78/original/rcrtvaelvfae84czze1w-pdf.pdf retrieved: September, 2021.

[2] "Warum hat der Rasen Streifen? (English: Why does the lawn have stripes?)." Online. Available at https://www.dfb.de/news/detail/warum-hat-der-rasen-streifen-70569/ retrieved: September, 2021.

[3] F. Tietjen, "So erhält der Stadionrasen sein Muster. (English: This is how the stadium turf

gets its pattern.).” Online, 2015. Available at https://www.netzathleten.de/lifestyle/sports-inside/item/5737-so-=rhaelt-der-stadionrasen-sein-muster retrieved: September, 2021.

[4] “Commercial ZTrak Zero-Turn Mower.” Online, 2019. Available at https://www.deere.com/assets/publications/index.html?id=917cd17c#6 retrieved: September, 2021.

[5] T. Bektas, “The Multiple Traveling Salesman Problem: An Overview of Formulations and Solution Procedures,” *Omega*, vol. 34, no. 3, pp. 209–219, 2006.

[6] E. M. Arkin, S. P. Fekete, and J. S. Mitchell, “Approximation Algorithms for Lawn Mowing and Milling,” *Computational Geometry*, vol. 17, no. 1, pp. 25–50, 2000.

[7] I. Kara and T. Derya, “Formulations for Minimizing Tour Duration of the Traveling Salesman Problem with Time Windows,” in *Procedia Economics and Finance*, vol. 26, pp. 1026–1034, 2015.

[8] S. Akter, N. Nahar, M. ShahadatHossain, and K. Andersson, “A new crossover technique to improve genetic algorithm and its application to TSP,” in *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, pp. 1–6, February 2019.

[9] Y.-F. Liao, D.-H. Yau, and C.-L. Chen, “Evolutionary algorithm to traveling salesman problems,” *Computers & Mathematics with Applications*, vol. 64, pp. 788–797, September 2012.

[10] X. Wang and G. Xu, “Hybrid differential evolution algorithm for traveling salesman problem,” *Procedia Engineering*, vol. 15, pp. 2716–2720, 2011.

[11] V. Karamcheti and M. Malek, “A TSP engine for performing tabu search,” in *Proceedings of the International Conference on Application Specific Array Processors*, pp. 309–321, IEEE Comput. Soc. Press, 1991.

[12] Y. Liu, S. Xiong, and H. Liu, “Hybrid simulated annealing algorithm based on adaptive cooling schedule for TSP,” in *Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation - GEC '09*, pp. 895–898, 2009.

[13] X. Geng, Z. Chen, W. Yang, D. Shi, and K. Zhao, “Solving the traveling salesman problem based on an adaptive simulated annealing algorithm with greedy search,” *Applied Soft Computing*, vol. 11, pp. 3680–3689, June 2011.

[14] A. Shetty, A. Shetty, K. S. Puthusseri, and R. Shankaramani, “An Improved Ant Colony optimization Algorithm: Minion Ant(MAnt) and its Application on TSP,” in *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1219–1225, November 2018.

[15] C. E. Miller, A. W. Tucker, and R. A. Zemlin, “Integer Programming Formulation of Traveling Salesman Problems,” *Journal of the ACM*, vol. 7, no. 4, pp. 326–329, 1960.

[16] T. Majeed, M. Handschuh, and R. Meier, “Automatic Scheduling of Dependency-Based Workflows,” in *Distributed Computing and Artificial Intelligence, 14th International Conference*, pp. 309–317, Springer International Publishing, June 2017.

[17] E. M. Arkin, M. A. Bender, E. D. Demaine, S. P. Fekete, J. S. B. Mitchell, and S. Sethia, “Optimal Covering Tours with Turn Costs,” *SIAM Journal on Computing*, vol. 35, pp. 531–566, January 2005.

[18] S. P. Fekete and D. Krupke, “Covering Tours and Cycle Covers with Turn Costs: Hardness and Approximation,” in *Lecture Notes in Computer Science*, pp. 224–236, Springer International Publishing, 2019.

[19] E. M. Arkin, S. P. Fekete, and J. S. B. Mitchell, “The Lawnmower Problem,” in *Proceedings of the 5th Canadian Conference on Computational Geometry, Waterloo, Ontario, Canada, August 1993*, pp. 461–466, University of Waterloo, 1993.

[20] J.-S. Chen and J.-S. Yan, “Model Formulations for the Machine Scheduling Problem with Limited Waiting Time Constraints,” *Journal of Information & Optimization Sciences*, vol. 27, no. 1, pp. 225–240, 2006.

[21] D. P. Ronconi and E. G. Birgin, “Mixed-integer programming models for flowshop scheduling problems minimizing the total earliness and tardiness,” in *Just-in-Time Systems*, pp. 91–105, Springer New York, October 2011.

[22] I. Gurobi Optimization, “Gurobi Optimizer Reference Manual,” 2017. Available at https://www.gurobi.com/documentation/9.1/refman/index.html retrieved: September, 2021.