International Journal of Geo-Information

MDPI

Article

Convolutional Neural Network-Based Approximation of Coverage Path Planning Results for Parking Lots

Andrius Kriščiūnas [1,*], Dalia Čalnerytė [1], Tautvydas Fyleris [1], Tadas Jurgutis [2], Dalius Makackas [1] and Rimantas Barauskas [1]

[1] Department of Applied Informatics, Faculty of Informatics, Kaunas University of Technology, 44249 Kaunas, Lithuania; tautvydas.fyleris@ktu.lt (T.F.); dalius.makackas@ktu.lt (D.M.); rimantas.barauskas@ktu.lt (R.B.)
[2] UAB Datava, 54373 Kaunas, Lithuania
* Correspondence: andrius.krisciunas@ktu.lt

Abstract: Parking lots have wide variety of shapes because of surrounding environment and the objects inside the parking lot, such as trees, manholes, etc. In the case of paving the parking lot, as much area as possible should be covered by the construction vehicle to reduce the need for manual workforce. Thus, the coverage path planning (CPP) problem is formulated. The CPP of the parking lots is a complex problem with constraints regarding various issues, such as dimensions of the construction vehicle and data processing time and resources. A strategy based on convolutional neural networks (CNNs) for the fast estimation of the CPP's average track length, standard deviation of track lengths, and number of tracks was suggested in this article. Two datasets of different complexity were generated to analyze the suggested approach. The first case represented a simple case with a working polygon constructed out of several rectangles with applied shear and rotation transformations. The second case represented a complex geometry generated out of rectangles and ellipses, narrow construction area, and obstacles. The results were compared with the linear regression models, with the area of the working polygon as an input. For both generated datasets, the strategy to use an approximator to estimate outcomes led to more accurate results compared to the respective linear regression models. The suggested approach enables us to have rough estimates of a large number of geometries in a short period of time and organize the working process, for example, planning construction time and price, choosing the best decomposition of the working polygon, etc.

Keywords: coverage path planning (CPP); optimization; convolutional neural networks (CNNs)

Citation: Kriščiūnas, A.; Čalnerytė, D.; Fyleris, T.; Jurgutis, T.; Makackas, D.; Barauskas R. Convolutional Neural Network-Based Approximation of Coverage Path Planning Results for Parking Lots. *ISPRS Int. J. Geo-Inf.* **2023**, *12*, 313. https://doi.org/10.3390/ijgi12080313

Academic Editors: Wolfgang Kainz and Maria Antonia Brovelli

Received: 20 June 2023
Revised: 19 July 2023
Accepted: 26 July 2023
Published: 30 July 2023

1. Introduction

The track generation for parking lot paving is one of the application areas of the coverage path planning (CPP) problem. The goal of the CPP problem is to determine the trajectories a vehicle or a robot must follow in order to cover all points of the area of interest without overlaying paths or colliding with obstacles [1]. CPP has a wide range of applications in the field of robotics, such as planning paths for cleaner [2] or disinfecting robots [3], agriculture [4–6], rescue operations [7], airport pavement disease detection [8], etc. CPP-based software enables farmers to plan the field management operations and quantitatively evaluate different operational plans [9]. The CPP problem can be analyzed at several granularity levels. The basic level is CPP for a single robot for a convex area of interest without obstacles [4]. A more complex level of CPP takes into account multiple obstacles [10–13]. The highest complexity level of CPP is the problem of multiple agents covering the area of interest [5,14].

Based on the application area, various requirements are considered in the CPP problem. For example, in the field of agriculture, the path planning algorithm should result in an "optimal" path that takes into account economic and environmental factors, such

as fuel consumption [4], configuration of machinery [15], curvature constraints [16], 3D geometrical representation [17,18], multiple filling points [19], and working time [14]. The CPP problem is based on solving the traveling salesman problem [20] and it is proved that this approach results in solving the NP-hard problem; therefore, it is not possible to obtain an optimal solution in a reasonable time.

The machinery type is also an important issue in CPP. The cleaner robots can usually change their track direction orthogonally [2]. In the field of agriculture, turns are constrained by the minimum turning radius of the vehicle [20]. The algorithm to generate corner turns for a vehicle with tillage operations in a paddy field with minimum turning radius taken into account was suggested in [21]. The Dubin's curve was applied to connect ordered field tracks, and minimum bounding was used to simplify the complex geometry of the area of interest [12].

There is a variety of approaches applied to solve the CPP problem. It can be formulated as the traveling salesman problem (TSP) and solved using ant colony optimization after decomposing the area of interest into blocks without obstacles [10]. The reinforcement learning approach was employed to find a coverage path as a TSP in grid environments by training a recurrent neural network [22]. A genetic algorithm was applied to generate the coverage path of a vacuum cleaner robot in a room [2]. A combination of a genetic algorithm and dynamic programming was presented to find the close-to-optimal path of a robot in an unknown environment with obstacles [23]. A real-valued genetic algorithm was employed to optimize a sequence of track blocks and their enter and exit points and therefore obtain the field coverage path of an agricultural vehicle [24]. A pseudo-spanning tree-based algorithm with virtual nodes and edges using approximate cellular decomposition was proposed to solve the CPP problem [25]. A comprehensive survey of the literature on the CPP is provided in [1]. The majority of the CPP algorithms use decomposition of the area of interest. Based on the decomposition techniques, the approaches were grouped into cellular decomposition, landmark-based, grid-based, and graph-based methods [1]. The algorithms for coverage path planning in robotics were categorized into classical and heuristic approaches in [26]. The examples of classical approaches are dynamic programming, spanning tree coverage, and chaotic coverage. The greedy search, graph search, and bio-inspired algorithms were classified as heuristic-based methods [26].

In this article, an algorithm to generate paving paths for a parking lot area under construction is suggested. The problem can be defined as the CPP problem for a single vehicle in a non-convex polygon with internal obstacles. The research is organized as follows. Section 2 presents the methods and materials, namely, the general workflow, problem description, and CPP algorithm, including the obstacle elimination approach, generation of the representative dataset, construction of the convolutional neural network (CNN) model, and its application in the decomposition algorithm. Section 3 is dedicated to numerical experiments, that is, training of the approximation model and its validation and analysis of results under different conditions. The discussion is presented in Section 4. The conclusions are given in Section 5.

2. Materials and Methods

2.1. General Workflow

The conventional CPP approach is straightforward. Based on the initial data, such as geometry, vehicle dimensions, and other restrictions, an algorithm to generate the coverage path is designed and the final output is the generated trajectories for the given problem. The scheme for the conventional CPP approach is provided in Figure 1.

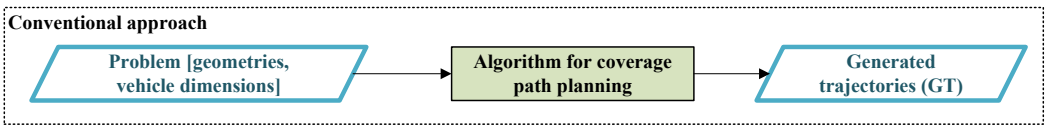


Figure 1. Scheme for conventional coverage path finding workflow.

Such an approach works well for simple problems. However, in practice, the problem under analysis is more complicated as regards the complex geometry and number of vehicles that are employed in the process. Thus, a lot of parameters must be considered in the objective function, including the features that play the most important role in the decomposition process. Despite the direct approach of splitting the geometry into smaller areas and examining them separately, the placement of depots, change in technology or equipment due to many obstacles, workers' convenience, and risk management should be taken into account. In most cases, an evaluation of these factors cannot be performed due to a lack of data and uncertainties. For example, several machines operating in the same area simultaneously can cause underestimated slow down, although the hard constraints of the problem are maintained. In such cases, the final decision for the coverage path is made by the experts in the field, whereas the CPP algorithm is used as a recommendation and evaluation tool which helps to faster evaluate different coverage strategies. However, comparison of different coverage path strategies is time- and computationally expensive because of the optimization in the CPP procedure, which may take several minutes for one strategy, and therefore comparing all possible strategies leads to practically unacceptable computational time, especially if an expert must make the decision in the working place fast. Thus, the regression model can be applied to predict the results and make a fast comparison of different strategies. Although any regression model can be used in general, the CNN regression model has been employed in this research to consider geometric features of the problem. This leads to the extended CPP process workflow (Figure 2). The process presented in Figure 1 is complemented with the additional steps of building a convolutional neural network (CNN)-based model to predict the numerical evaluation of the CPP with input presented as an image. To train the model, the representative dataset is generated artificially in advance.

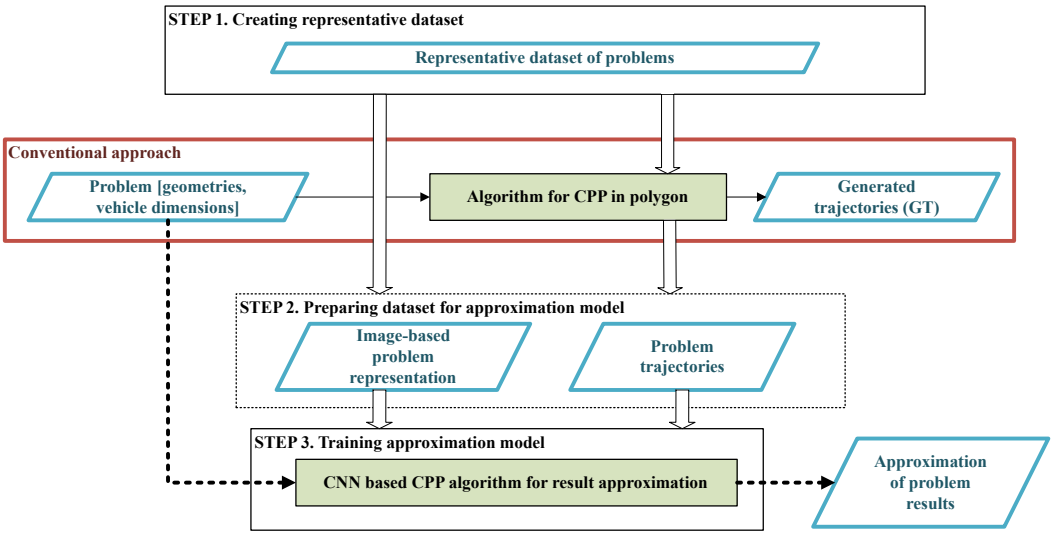


Figure 2. The extended CPP process workflow.

Using the model, an expert can perform fast evaluation of the specific problem and consider various strategies of the process organization under minimized time and computational resources. Approximate results of the algorithm may be used even in the automatic decomposition algorithm with a well-defined objective function, when decomposition is performed based on the results predicted for the decomposed part. Of course, the decomposition may be applied using results obtained using the CPP algorithm itself; however, such an approach is significantly slower due to the computational time compared to image inference using the CNN model.

2.2. Problem Description

The CPP problem is focused on finding a set of tracks that cover the area of interest. The surface of the three-dimensional (3D) terrain is considered in agriculture. However, parking lots are usually installed in flat areas, so the two-dimensional (2D) area is considered. The region where paving needs to be applied is called a working area. The region in which the construction vehicle can move is called the construction area. The working area is inside the construction area. In some cases, it is possible that the boundaries of the construction area and working area coincide. For example, if the working area is surrounded by a wall, the construction vehicle cannot move outside it. A construction vehicle moves in tracks. The track is not necessarily a straight line, but it consists of straight lines (segments). Finding an optimal order of tracks is a different type of optimization problem not considered in this research. Obstacles are the areas inside the region of interest which must not be covered, and it is considered that the construction vehicle cannot even move in the obstacle area. The definitions are illustrated in Figure 3.

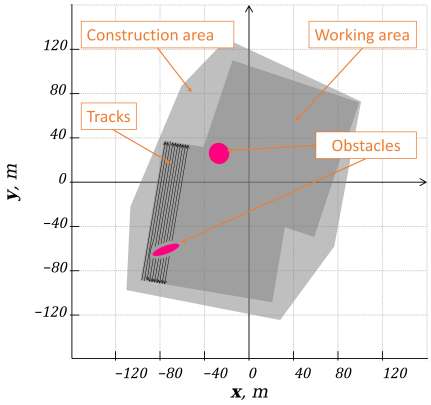


Figure 3. Geometrical representation of the problem.

Obviously, the track line depends on the parameters of the construction vehicle. It results in the constrained length of the possible distance between the boundaries of the construction area and the end points of the track, the bounded angle that two adjacent segments of the track can intersect, and other track features. The track line is discretized into segments with a minimal possible segment length defined in order to prevent an infinite number of segments. After generating, each track is validated to meet the parameters defined in Table 1.

Table 1. The parameters of the construction vehicle and track validation.

Parameter	Value	Parameter	Value
Minimum track discretization step	0.1 m	Vehicle width	3.2 m
Implemented width	2.9 m	Distance between vehicle axles	3.08 m
Working width (milling rotor width)	2.5 m	Front wheel width	0.65 m
Required track overlap	0.2 m	Front turning radius	5.96 m
Working rotor diameter	1.0 m	Outer front turning radius	7.495 m
Distance from the rear axle to the vehicle back	1.1 m	Inner front turning radius	4.425 m
Vehicle length	8.45 m	Turning radius with lifted milling rotor	5.102 m

The aim of the CPP problem is to find a set of tracks which cover the area of interest. In the CPP for the parking lot paving, the covered area should be maximized because the uncovered area must be covered, requiring manual labor, and therefore, the construction process results in higher cost. Moreover, there are more requirements that need to be considered. Firstly, the excessive overlap between tracks should be minimized. Excessive track overlap leads to increased material usage and elongated coverage process. Secondly, during

the coverage process, the construction vehicle incurs non-working driving maneuvers and working mode changeovers (e.g., lowering or raising a milling rotor) at the beginning and end of each track. Such intermittent setups are time-consuming tasks. Thus, the CPP solution with fewer tracks is considered as more efficient compared to the one with a larger number of tracks. Thirdly, there is a set of vehicle geometry and kinematics constraints which must be considered while generating the track, for example, the minimum vehicle rotation radius, which constrains the angle between two segments, or the minimum segment length, which constrains the discretization step.

2.3. Algorithm for CPP in Polygon

The suggested CPP algorithm is based on the idea that the tracks should be aligned with one of the boundaries of the working polygon. The algorithm input consists of the geometry of the working and construction polygons, vehicle parameters, and algorithm parameters. Vehicle parameters, such as turning radius, width, and length, limit the curvature of the track and define the track discretization level. The number of approximation nodes in boundary noise reduction, the minimum allowed length of the track, the discretization step, and other parameters related to algorithm performance are defined as algorithm parameters. However, by taking into account that only the CPP algorithm itself is the object of this research and that algorithm parameters can be adjusted to meet the problem aim, the CPP algorithm steps are provided without the analysis of different parameter configurations. The flow diagram of the algorithm for the CPP in polygon is provided in Figure 4.

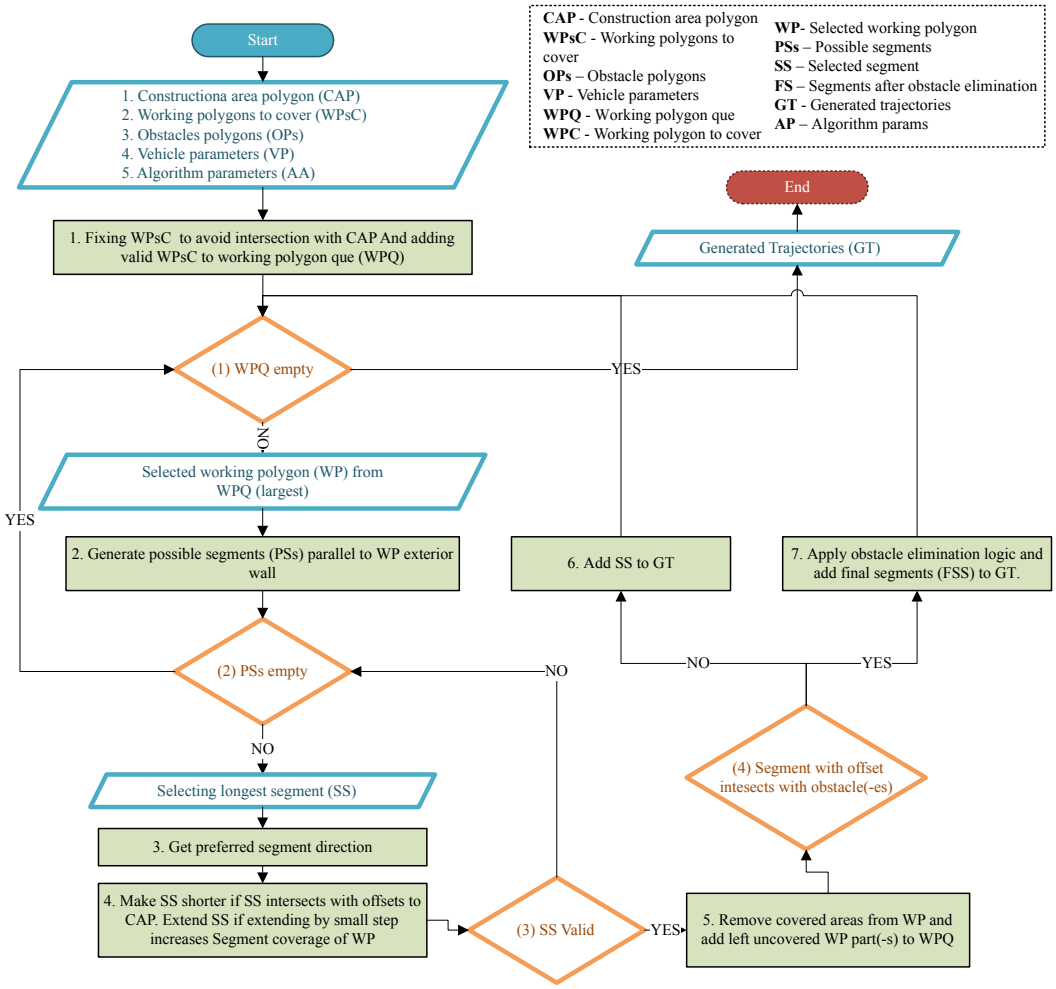


Figure 4. The flow diagram of the CPP algorithm.

The algorithm consists of seven steps. In the first step, the queue of working polygons is prepared for the later calculations. If the geometry of the problem is generated regarding the points measured in the field, measurement errors emerge due to limited precision of the measurement devices. Thus, the geometry of the working polygons must be fixed to fit into the construction area. Boundary noise reduction is also performed in this step. At the second step, the tracks parallel to the exterior boundary are generated for the largest working polygon. Then (steps 3–4) the search for the valid track is performed by selecting the direction of the reference track and expanding or shortening it to fit it to the working polygon with consideration of the construction area. By default, the longest track is selected as the reference track. In case the track is not valid (for example, does not fit the *minSegmentLen* constraint), another candidate track is analyzed. If the track is valid, the region covered by the track is subtracted from the working polygon, and the resulting polygons are added to the working polygon queue as new working polygons (step 5). If the track has no intersection with obstacles, it is added to the list of the generated tracks (step 6). The intersection with an obstacle causes additional obstacle elimination logic, detailed in Section Processing of Obstacles (step 7). Finally, the steps are reiterated until all working polygons are analyzed. The output of the algorithm is a set of generated tracks which satisfy the requirements, namely, all segments in the track are longer that the predefined minimum segment length, the angle between two consecutive segments corresponds to the turning radius of the vehicle, and all segments are inside the working area.

Processing of Obstacles

Firstly, the tracks are generated without considering obstacles. Then the formatted tracks are modified to overtake the obstacles. The tracks to overtake the obstacle were generated using the Dubin's curve [27] with known outer segments and the parameters of the vehicle. The generation of Dubin's curve uses an analytical approach and the possible overtake path is calculated directly using formulas. However, obstacle elimination even in one generated track may lead to a multi-objective evaluation function. Because of the vehicle dimensions (coverage is performed at the end of vehicle), the detected obstacle results in the overtaking maneuver that starts before the obstacle and leads to uncovered area in front of the obstacle. The uncovered area around an obstacle could be covered by creating additional opposite-direction track, but such generated track is usually too short and thus it is not preferred due to an increase in required non-working maneuvering time. Alternatively, there is an option to split the initial segment and try to avoid applying the overtaking maneuver altogether. In this article, we propose a parameterized approach, which allows us to adopt a logic based on three main analysis cases. The approach has the following managed parameters:

1. d_b—distance in front of the obstacle to start the overtake obstacle;
2. d_m—maximum distance to overtake an obstacle;
3. a_m—maximum start/end position angle.

Based on the obstacle location in the pre-formatted track, the algorithm works as follows:

Case 1. If the obstacle is too close to the beginning of the segment, that is, the buffer of the vehicle enters the obstacle at the beginning of the segment:

- From the starting point of the segment, an attempt is made to go around the obstacle with an initial angle from 0 to a_m.
- The beginning of the segment is pushed further if the vehicle fails to go around the obstacle.

Case 2. If the obstacle is too close to the end of the segment, that is, the buffer of the vehicle enters the obstacle at the end of the segment:

- Check if the vehicle can enter the end of the track at an angle in range $0, \ldots, a_m$.
- End of the segment is shortened to the obstacle if the vehicle fails to go around the obstacle.

Case 3. If the segment intersects with an obstacle in the middle of the segment:

- If it is possible to overtake the obstacle in d_m, generate a path for the overtake in the minimal distance possible.
- If it is not possible to overtake the obstacle in distance d_m, the segment is divided into two parts.

In the event of multiple obstacles, a straightforward extension of the above procedure is applied when finding intersection(s). The track is analyzed from the beginning to the end in a discrete step with each obstacle or a group of close obstacles fixed as one. The path to overtake the obstacle is generated using the following procedure:

1. Find the minimum-length curve which overtakes the obstacle by increasing the step from 0 to d_m. The overtake path is examined with the maximum detour from both sides.
2. Find the minimum detour of selected valid length segment (this allows us to minimize uncovered area, for example, if the obstacle is in the form of an ellipse).

Finally, finding curve to overtake the obstacle leads to a controlled computational time procedure, because all procedure steps (finite set of angle values in discrete step and curve length that changes in range from 0 to d_m in Cases 1, 2) depend on the discretization step, which could be selected with respect to available computational time and precision requirements.

2.4. CNN-Based Approximator of CPP Algorithm Results

Two datasets of different complexity have been generated to demonstrate the potential applications of approximators. Case 1 represents CPP problems with simple geometry and Case 2 represents problems of nearly real-life complexity in the parking lot coverage process:

1. **Case 1.** Simplified problem with the following features:
   - Working area geometry consists of merged rectangles without internal obstacles.
   - Construction area boundary is far enough from working area boundary, not affecting the path coverage process due to vehicle geometry and offsets.
2. **Case 2.** Complex problem with the following features:
   - Working area geometry consists of merged rectangles and circles.
   - Different number of various obstacles may occur.
   - Construction area boundaries may disturb path coverage process.

Depending on the case, the analysis is performed in different ways. For Case 1, an automatic decomposition algorithm has been taken into account. The objective function is to minimize the total number of tracks when the geometry splitting decision is made with respect to the approximation results. The splitting is performed according to the possible segments, which are formatted parallel to the WP exterior wall (a detailed explanation is provided in Section 2.4.1). The representative dataset used for training was composed of an initially generated dataset and decomposed problems (the generation of the representative dataset is explained in Section 2.4.2). For Case 2, the initial geometry is more complex and so the optimal decomposition can be the result of different approaches, orienting tracks parallel to the obstacle borders, taking into account construction area limitations, etc. Such straightforward decomposition approaches might not provide better simple objective function results. Also, as explained in Section 2.1, in practice, even the objective function itself might be subjective and thus difficult to express numerically. Given these Case 2 considerations, we limited our research to the model training and initial results analysis. The CPP approximation model architecture and metrics results are provided in Section 2.4.3.

2.4.1. Search of Best Polygon Decomposition Using Convolutional Neural Network

The proposed CPP algorithm (Section 2.3) uses a heuristic approach to select the longest possible segment generated of possible segments with respect to the exterior wall of the working polygon (WP). However, based on the problem objective, such an approach

may not lead to the optimal solution. In this research we will examine a simplified case (Case 1) in detail. In this case, the decomposition approach and the objective function are straightforward. The objective is to cover an area with the lowest possible number of tracks. An example of how track number differs based on the initial decomposition is provided in Figure 5. If the longest track is selected out of possible boundary tracks as a reference track, the resulting number of tracks is 93 (Figure 5a). In comparison, if the reference track is selected using an expert's knowledge or is based on a different algorithm, the number of tracks is 79 (Figure 5b). This example illustrates the fact that decomposition should be performed with consideration of the construction area geometry.

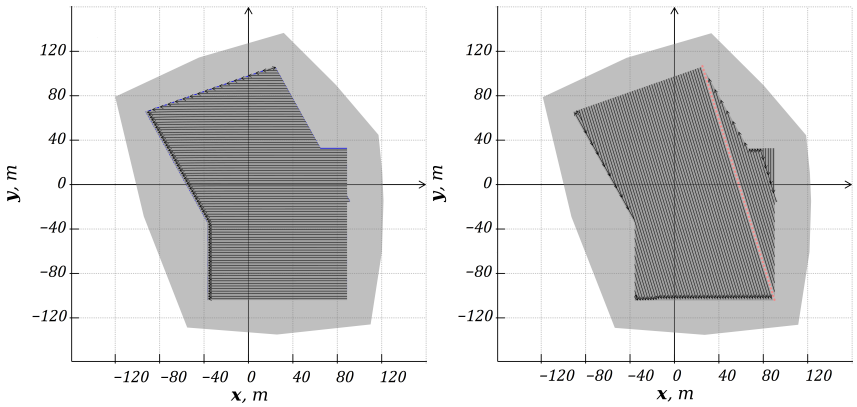


Figure 5. The resulting coverage path generated by selecting longest possible track as a reference track (**a**) and manually choosing the reference track (**b**). The reference tracks are shown in red.

In order to find the optimal decomposition which results in the minimum number of tracks, all possible decompositions must be considered. However, the CPP algorithm is a time-consuming process and considering all possible decompositions is not practically acceptable. In this case, an approximator of the CPP algorithm results may be used to predict the number of tracks for the decomposed parts and therefore the whole geometry. The CNN-based approximator application schema is provided in Figure 6.

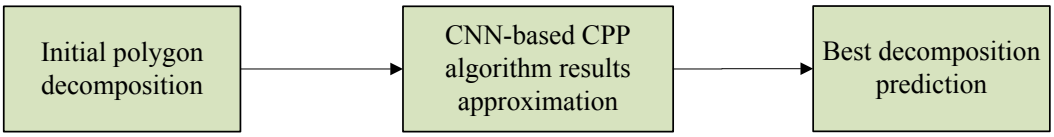


Figure 6. CNN-based approximator application schema.

The schema consists of three main parts, that is, decomposing the initial working polygon, predicting the number of tracks for each subarea, and searching for the best decomposition. In the decomposition step, the working polygon is partitioned with respect to the reference track. There may be several levels of the decomposition; however, in this research, the decomposition level was limited to the first one. In the prediction step, the number of tracks is predicted for each decomposition result. For example, geometry G_i can be decomposed in K different ways and K sets of geometries are generated: $\{G_{i11}, G_{i12}, \dots\}$, $\dots$, $\{G_{iK1}, G_{iK2}, \dots\}$. The number of tracks p_j for the jth set of geometries is calculated as

$$p_{ij} = \sum_{l} p_{ijl} + 1 \tag{1}$$

where p_{ijl} is the number of tracks for the lth part in the jth set of geometries with initial ith geometry. The sum is augmented by 1 as the geometries are generated by subtracting the polygon covered by the reference track. The decomposition algorithm provided for Case 1 dataset generation can be directly applied in this case. An example of the initial problem

decomposition, if splitting is performed based on possible segments formatted parallel to the SP exterior wall extension and its node connection, is provided in Figure 7.

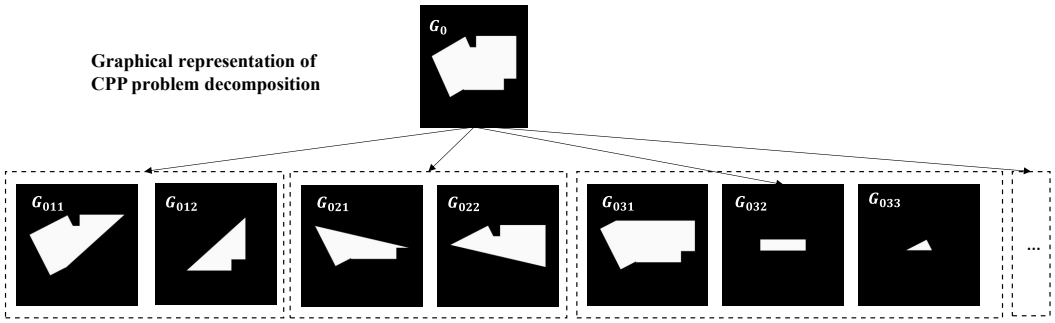


Figure 7. Tree of decomposition cases.

2.4.2. Generation of Representative Dataset

Training the CNN model requires many samples to capture the geometry features. The automatic generation of the dataset enables us to create a desired number of samples and therefore the training dataset is not limited by the real-world examples. Thus, the datasets in the research were generated artificially. A parameterized algorithm has been constructed in order to prepare the representative dataset for the approximation model. The typical geometric features can depend on the country because of urban state laws for parking lots. They can also change in time because of the new technologies and dynamic environmental requirements. By changing the parameters of the dataset generation algorithm, a desired number of examples with typical geometric features can be generated. Each case is formatted as follows:

1. The probabilities for every shape to become a rectangle or circle are $p_{rectangle}$ and $p_{circle} = 1 - p_{rectangle}$, respectively.
2. Generate a random base shape (circle or rectangle) with predefined dimensions in the interval $[b_{min}, b_{max}]$. Here, b_{min} and b_{max}, respectively, are the min and max values of height and width for a rectangle or radius for a circle.
3. Generate m number of sub-shapes in the same way as in Step 1.
4. The center of every additional shape is randomly placed inside already generated shapes. The final polygon is constructed as the union of shapes.
5. Perform the resulting polygon external boundary discretization by step in the range $[bstep_{min}, bstep_{max}]$, and for every point, apply random offset $[buffer_{min}, buffer_{max}]$ along the curve.
6. Randomly place (with the same probability) k ellipse- or rectangle-type obstacles inside a working area formatted as follows (in both cases, the rotation angle is randomly selected from 0 to 360):
    - Ellipse: The parameters of the ellipse are $[obs_{e\_min}, obs_{e\_max}]$ for the dimensions of an ellipse along the x and y axes, respectively.
    - Rectangle: Create the rectangle with random width and height from the range $[obs_{r\_min}, obs_{r\_max}]$. The shear transformation is applied with a probability of 0.5. The shear angle is a random value from the interval $[-15°, 15°]$.

In all dataset generation cases, the same parameters for values $[b_{min}, b_{max}] = [100, 150]$ were used. In cases with obstacles considered, the parameters to generate them are $[obs_{e\_min}, obs_{e\_max}] = [obs_{r\_min}, obs_{r\_max}] = [1, 20]$. The main difference is that Case 1 samples do not have any obstacles (k=0), the construction area does not disturb tracks ($buffer_{min} > truckLen$), and geometries are created out of rectangles ($p_{rectangle} = 1$). Both types of disturbances can occur in the Case 2 dataset ($[buffer_{min}, buffer_{max}] = [0.5, 20]$), and the ellipse shape may occur with probability $p_{circle} = 0.1$.

Finally, the geometries were transformed into grayscale images of 250 × 250 pixels orienting them in the middle of the image to represent an area of 400 × 400 m. By considering

that Case 1 problems depend only on the geometry of the working polygon (the construction polygon does not impact the results), the problem is represented as a white polygon against a black background. For the Case 2 dataset, two additional colors, "light gray" and "dark grey", are used for the construction area and obstacles, respectively. Examples of randomly generated cases with different values of m and k and their image representations are provided in Figure 8.

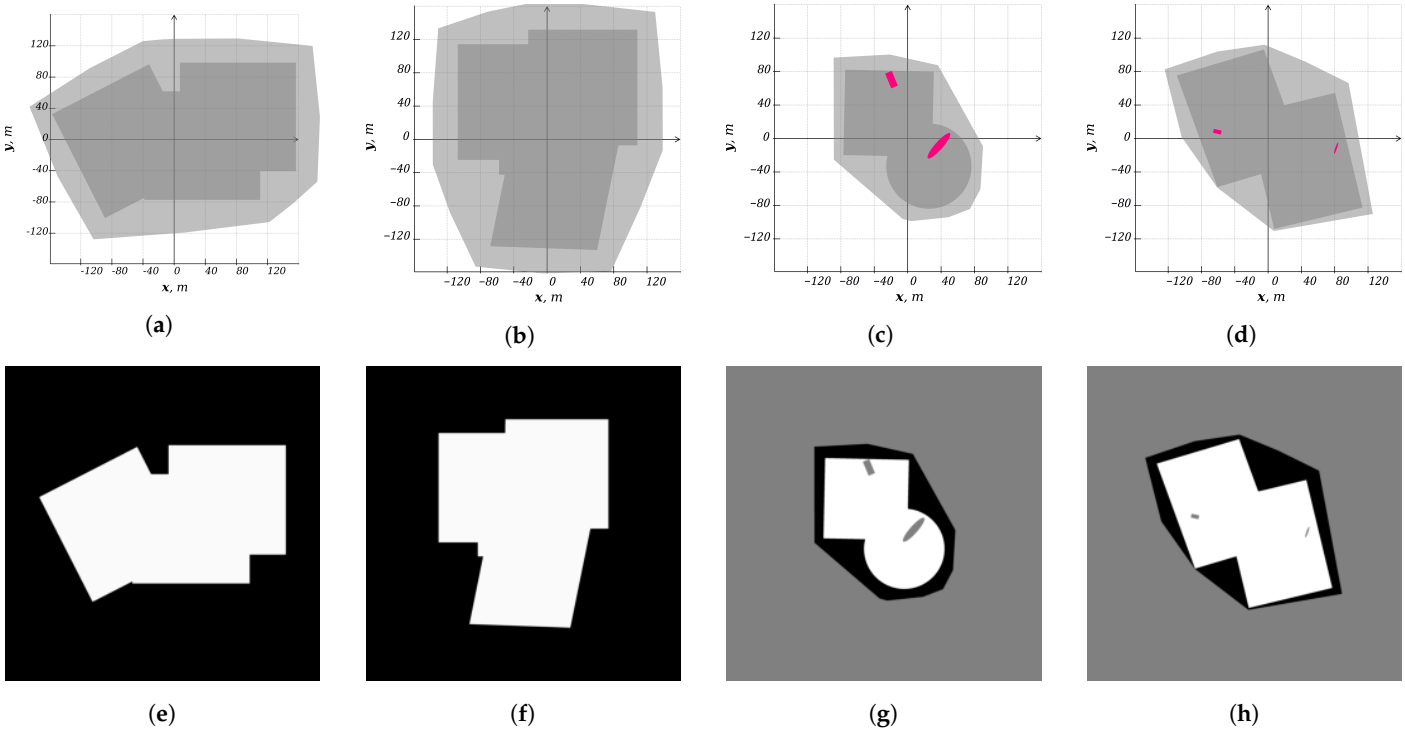


Figure 8. The examples of the samples generated in order to represent cases in which the construction vehicle can freely move out of the working area to the construction area (**a**,**b**) and cases in which the motion of the construction vehicle is restricted by the boundaries of the construction area (**c**,**d**) and their image representations (**e**–**h**), respectively.

For the Case 1 dataset, 1000 geometries were generated for each number of polygons $T_{poly} = \{1, 2, 3, 4\}$, that is, 4000 geometries in total. The geometries were split into training, validation, and test datasets in the ratio 60/20/20% using a stratified sampling technique with respect to the number of polygons. Each image was decomposed with respect to the procedure provided in Section 2.4.1, and each decomposed subproblem is stored as an independent one in the same subset as the initial problem. Finally, the dataset for Case 1 consisted of 69,784 problems, with 41,924, 13,771, and 14,089 problems in the training, validation, and testing datasets, respectively. For Case 2, a relatively small dataset has been created because of limited computational resources to obtain an exact solution for each case. Different types of polygons (including circles and ellipses) and considering obstacles result in many possible decomposition cases. Thus, analyzing all possible decompositions by exactly solving the CPP problem for each decomposed geometry is a time-consuming process. In this research, the representative dataset of 1000 cases for each combination of parameters $m \in \{0, 1\}$ and $k \in \{0, 1, 2\}$ has been created (zero or one additional subshape, and zero, one, or two obstacles). In total 6000 cases have been created, and these were divided by the same ratio into 3400 cases for training and 1200 cases for validation and testing.

2.4.3. Convolutional Neural Network (CNN) Model

CNN models are mainly applied to analyze images in the form of multi-dimensional matrices. To evaluate the geometry of the working and construction areas using the CNN

model, they were represented as a square grayscale image. The architecture of the CNN model was based on the AlexNet architecture [28]. Compared to the recently developed architectures, AlexNet is a light CNN model that demonstrates high performance. The input size of AlexNet (224 × 224 × 3 pixels) was changed to 250 × 250 × 1 pixels to fit the dimensions of the input image. The number of channels was reduced from three, relevant to RGB images, to one, since a grayscale image was analyzed and one channel is enough to describe the geometry classes, such as working area, construction area, obstacles, and background. Thus, the number of trainable parameters changed accordingly. The model has five convolutional layers, with max pooling layers after the first, second, and fifth convolutional layers. The convolutional layers are followed by two fully connected layers, with a dropout layer after each of them and five fully connected layers. To apply the CNN model to the regression problem, the output layer consists of three numbers, which represent an average of track lengths, its standard deviation, and total tracks for the analyzed geometry. The architecture of the CNN used in the regression model is provided in Figure 9 with parameters defined for each layer.

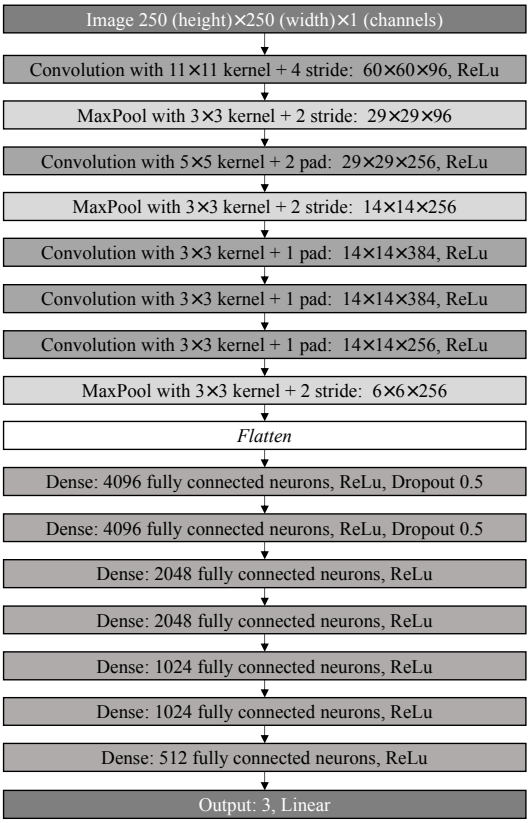


Figure 9. The architecture of the CNN-based regression model.

The loss function is constructed of terms that represent average track length, the standard deviation of track lengths, and the number of tracks. These terms allow an expert to assess the quality of the results. The desired result is to cover the geometry in a low number of tracks of similar practically acceptable track length. Thus, all the included components are important to know and therefore are used in the CNN approximation model as output. The averaged mean squared error (MSE) was used as the loss function in the training process:

$$MSE = \frac{1}{3n}\left(\sum_{i=1}^{n}(y_{len_i} - \hat{y}_{len_i})^2 + \sum_{i=1}^{n}(y_{std_i} - \hat{y}_{std_i})^2 + \sum_{i=1}^{n}(y_{tot_i} - \hat{y}_{tot_i})^2\right) \quad (2)$$

where n is the number of samples. Each term represents the difference between actual and predicted track lengths $(y_{len_i} - \hat{y}_{len_i})$, standard deviations $(y_{std_i} - \hat{y}_{std_i})$, and total tracks $(\hat{y}_{tot_i} - \hat{y}_{tot_i})$ for the i-th case. Here, y and $\hat{y}$ represent actual and predicted values of different metrics, respectively.

3. Numerical Experiments

3.1. Results of CPP Algorithm

The CPP results are demonstrated for the datasets with distinct characteristics. Firstly, the CPP results were shown for the selected samples from the benchmark dataset [6]. The same examples have been analyzed with the condition that the construction area is large enough for the construction vehicle to fully leave the working area and with the condition that the construction area may disturb the vehicle's motion, including randomly rotated obstacles of rectangular or ellipsoidal form (Figure 10).

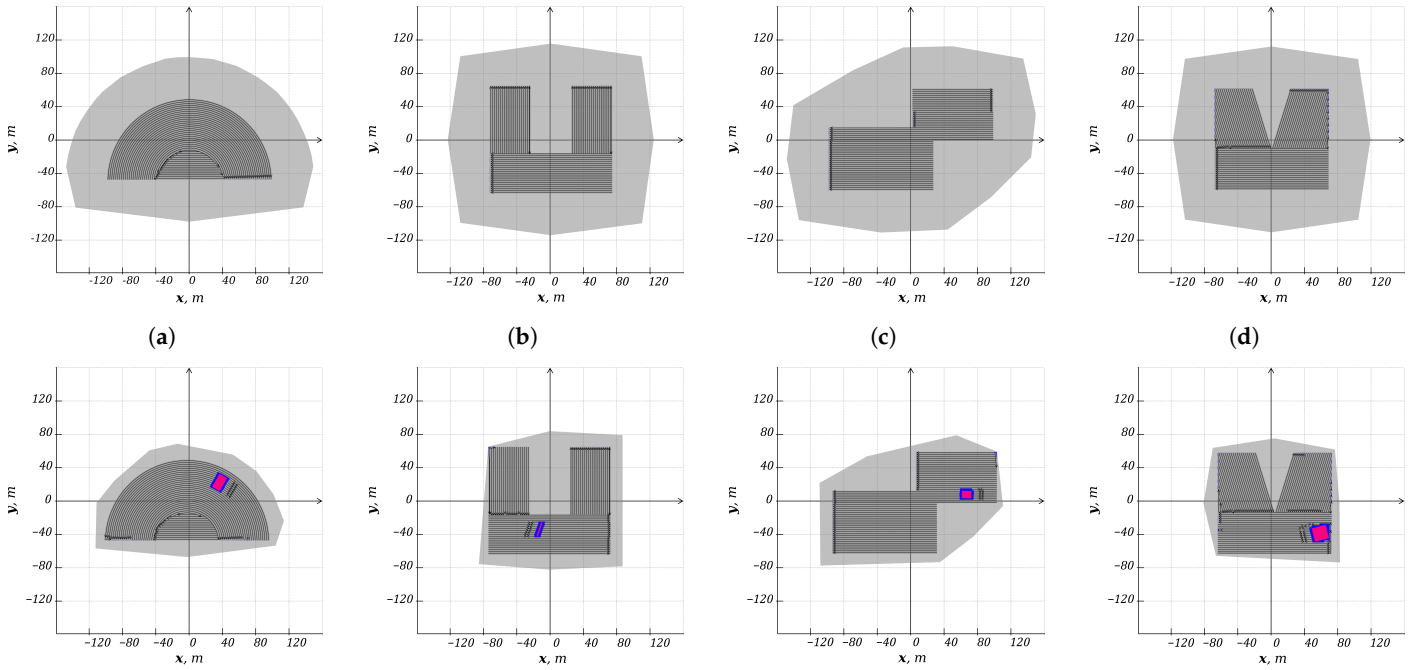
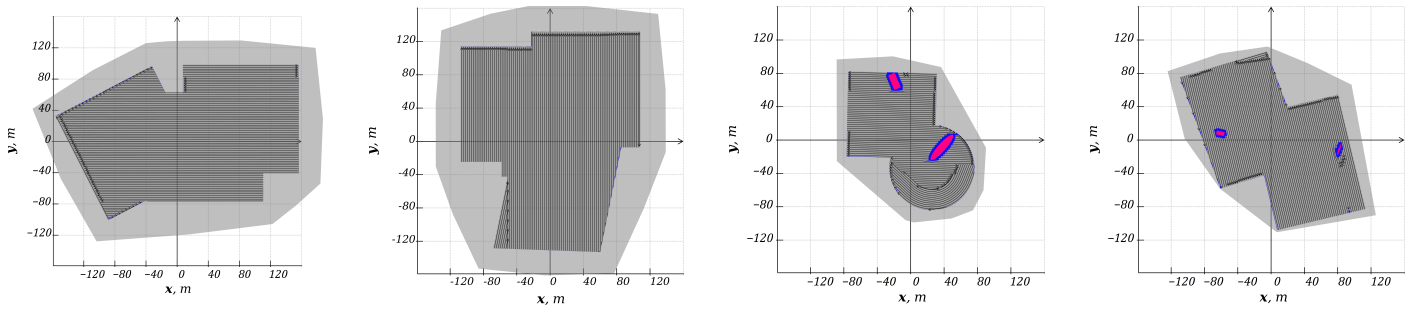


Figure 10. The CPP results for shapes from the benchmark dataset with large construction area (**a**–**d**) and narrow construction area (**e**–**h**).

The geometries with narrow construction area and obstacles usually have a higher number of tracks because of the maneuvers the vehicle must perform to cover the working area. The obstacles also create additional tracks if they cannot be overtaken. In addition, for some cases, the planned path does not fully cover the working area and there are regions that must be filled manually. The number of tracks and other performance may differ for known geometry because of the randomization of the obstacles and construction area. The subset of the benchmark dataset [6] has been used only for the Case 1 dataset in further analysis of the results' approximation (see Section 2.4). Tracks for the samples from Section 2.4 are provided in Figure 11.



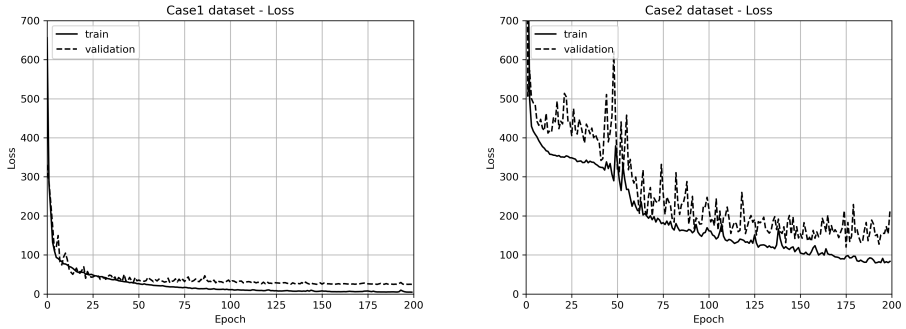
(**a**) (**b**) (**c**) (**d**)

Figure 11. Tracks for the samples from Case 1 (**a**,**b**) and Case 2 (**c**,**d**) datasets.

The final datasets are prepared by passing all the generated problems of the respective dataset (Case 1 or Case 2) through the CPP algorithm to calculate tracks in order to train the model with the actual results. This process is time-expensive, so it has been parallelized on technical equipment with 64 cores. Also, it must be mentioned that several cases (∼1%, 70 of 6000) of Case 2 did not pass successful track coverage. However, the research focus is the process of applying an approximation model for the decomposition, so the failed cases have been removed from the dataset in further analysis to avoid inappropriate training data for model construction.

3.2. Model Training and Results

The model has been trained for 200 epochs. Loss functions and mean absolute errors of averaged Case 1 and Case 2 datasets for training and validation datasets are provided in Figure 12.



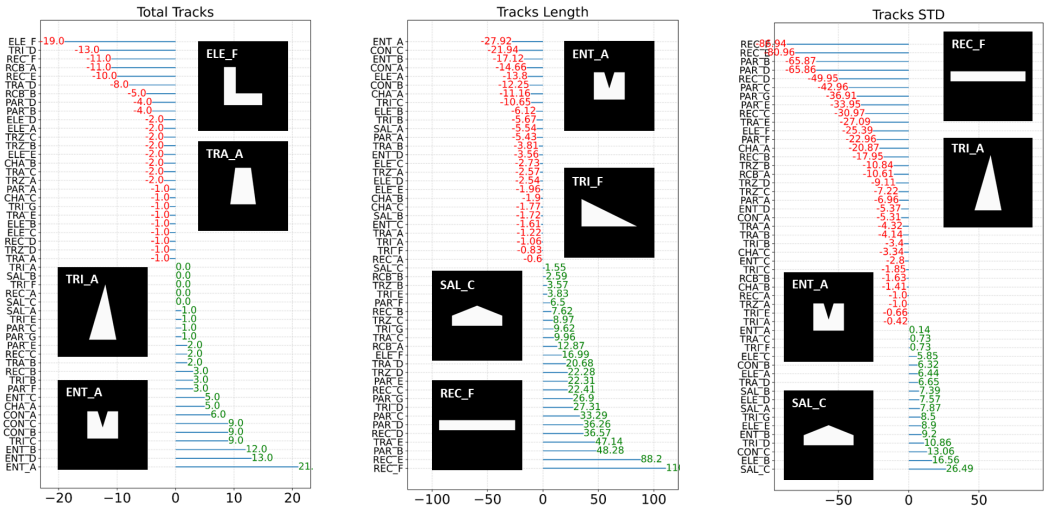
(**a**) (**b**)

Figure 12. Loss function values during model training: (**a**) Case 1 dataset; (**b**) Case 2 dataset.

The results demonstrate that the training curve is slightly below the validation one. This means that for Case 1 data, the model adapts too much for the training data. For such a case, the model can be simplified or the variety of training data should be increased to obtain accurate results. However, for the Case 2 dataset, the same model architecture results in noisier loss function curves for the training and validation dataset. For the Case 2 dataset, better results can be obtained by increasing the number and variety of the generated samples or using a different model architecture. However, this research is focused on the process of CNN-based model application instead of tuning the model architecture and its hyperparameters for a specific problem. Thus, the same model architecture and parameters are used for both cases.

3.2.1. Analysis of the Benchmark Dataset

In order to perform analysis for the simplified problem, the [6] dataset has been investigated. The circle-based samples were eliminated because there were no circle polygons included in the training dataset. The analysis of differences between the approximation and actual values for the geometry in the benchmark dataset is provided in Figure 13.



(**a**) (**b**) (**c**)

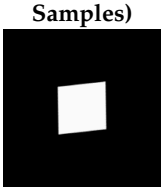
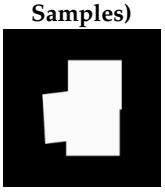
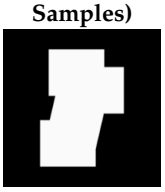
Figure 13. The differences in total tracks (**a**), length of tracks (**b**), and standard deviation of tracks (**c**).

The best- and worst-approximated shapes are additionally displayed in Figure 13 (all shapes by title are provided in [6]). It can be stated that problems of triangle-based geometry provide results close to the real ones. However, for the small rectangles or other shapes with track output that strongly differs from the average, the approximation errors are much larger, i.e., ELE_F has 20 tracks, the ENT_A shape of the same area has 78 tracks, and the predictions made by the model are 39 and 56, respectively. The errors may be related to the dataset used for training the approximation model, which was generated by the procedure provided in Section 2.4.2 and does not cover the specific cases presented in the benchmark dataset of long narrow rectangles and symmetric geometries. Nevertheless, the differences are practically acceptable in the majority of cases to give an initial estimate of the CPP problem results.

3.2.2. Analysis of Decomposition Efficiency for Simplified Problem

To evaluate the performance of decomposition-based decision making, the test Case 1 dataset has been analyzed. The objective was to minimize the number of tracks by performing initial splitting decisions based on the results of the approximation model. It must be mentioned that the approximation model does not guarantee the best splitting option; it may even suggest splitting the geometry when the non-splitting option provides better results. The impact of using decomposition is provided in Table 2 for the geometries generated out of 1, 2, 3, and 4 polygons (DS1, DS2, DS3, and DS4, respectively). The numbers of tracks for the algorithm that guarantee the optimal decomposition and the CNN-based algorithm for the decomposition are compared to the numbers of tracks calculated for the decomposition without any decomposition applied.

Table 2. Decomposition results for Case 1 types with geometry combined of 1, 2, 3, 4 polygons (DS1, DS2, DS3, DS4, respectively).

Dataset Type		**DS1 (200 Samples)**	**DS2 (200 Samples)**	**DS3 (200 Samples)**	**DS4 (200 Samples)**
Case Example					
Splitting applied by algorithm (optimal solution)	total samples with decomposition applied	0	40	19	22
	difference in track number	0	284	178	211
Splitting applied by CNN-based approximation	total samples with decomposition applied	0	37	19	17
	expected difference in track number	–	186	140	133
	actual difference in track number	0	184	58	103

The results in Table 2 demonstrate that current splitting strategy can improve performance for ∼20% cases of DS2 type problem, and ∼10% for DS3 and DS4 type problems. The problems of DS1 type are too simple and decomposition can only increase the number of tracks. No splitting recommendations were generated for this dataset type by CNN-based approximation model too. For the other problem types (DS2, DS3, DS4), the number of impacted samples for both optimal and approximation strategies are similar. However, the difference in track number is significantly different. The optimal decomposition results in reducing the number of tracks by 284, 178, and 211 for DS2, DS3, and DS4 type problems whereas the decomposition strategy to use to the CNN-based model recommendations results in reducing the number of tracks by 184, 58, 103 for DS2, DS3, and DS4 type problems respectively. For all types of the problems the difference for the approximation model is positive, thus, the CNN-based approximation model is able to identify the beneficial decomposition scenarios for the case if decomposition is considered as an optimal strategy for only ∼10–20% of the problems.

3.3. Analysis of Result Approximation for Case 2

The analysis for the more complex case (Case 2) was performed by taking into consideration subsets generated under different combinations of number of polygons m and obstacles k. The results of the approximation model were compared to the prediction made of the linear regression (LR) model trained with the data obtained using theconventional CPP algorithm:

$$y = \beta_1 \cdot S + \beta_0 \tag{3}$$

where y is the predicted values (number of tracks, average length of tracks, standard deviation of the track length), S is the working area of the problem geometry, and β_0, β_1 are coefficients of the LR model. The linear regression model covers the case when only the working area is considered, without taking into account its shape. LR is a simple and fast approximation technique that defines relationships between the variables. An intuitive approach is to estimate the cost of the parking lot pavement using area as a variable, although the actual time and cost of the construction work depend on the track plan (track length, number of tracks, difference in track length). Thus, comparing the results obtained

using LR- and CNN-based approaches enables us to evaluate the influence of geometry on CPP problems. The mean absolute error (MAE) for each predicted parameter and configuration has been provided in Table 3.

Table 3. The accuracy evaluation of linear regression and approximated predictions for track parameters.

DS Type	MAE of Average Track Length		MAE of STD		MAE of Number of Tracks	
	LR	Approx	LR	Approx	LR	Approx
$m = 0, k = 0$	22.96	5.44	17.54	8.52	12.91	2.86
$m = 0, k = 1$	12.25	7.22	8.35	5.64	7.45	4.85
$m = 0, k = 2$	20.74	7.57	10.88	4.07	13.78	6.90
$m = 1, k = 0$	18.44	11.38	16.43	12.07	12.37	6.96
$m = 1, k = 1$	12.48	10.95	12.36	9.75	9.15	8.31
$m = 1, k = 2$	19.38	8.95	11.99	8.61	14.97	8.73
All problems	17.71	8.60	12.95	8.14	11.77	6.44

The results demonstrate that using the LR approach has been outperformed by the CNN-based approximation model. The better results obtained using approximation mean that even for complex cases, the model evaluates the relationship between the shape of the working area and the predicted results. Examples of all configurations with actual values and values predicted by the approximation model are provided in Figure 14.

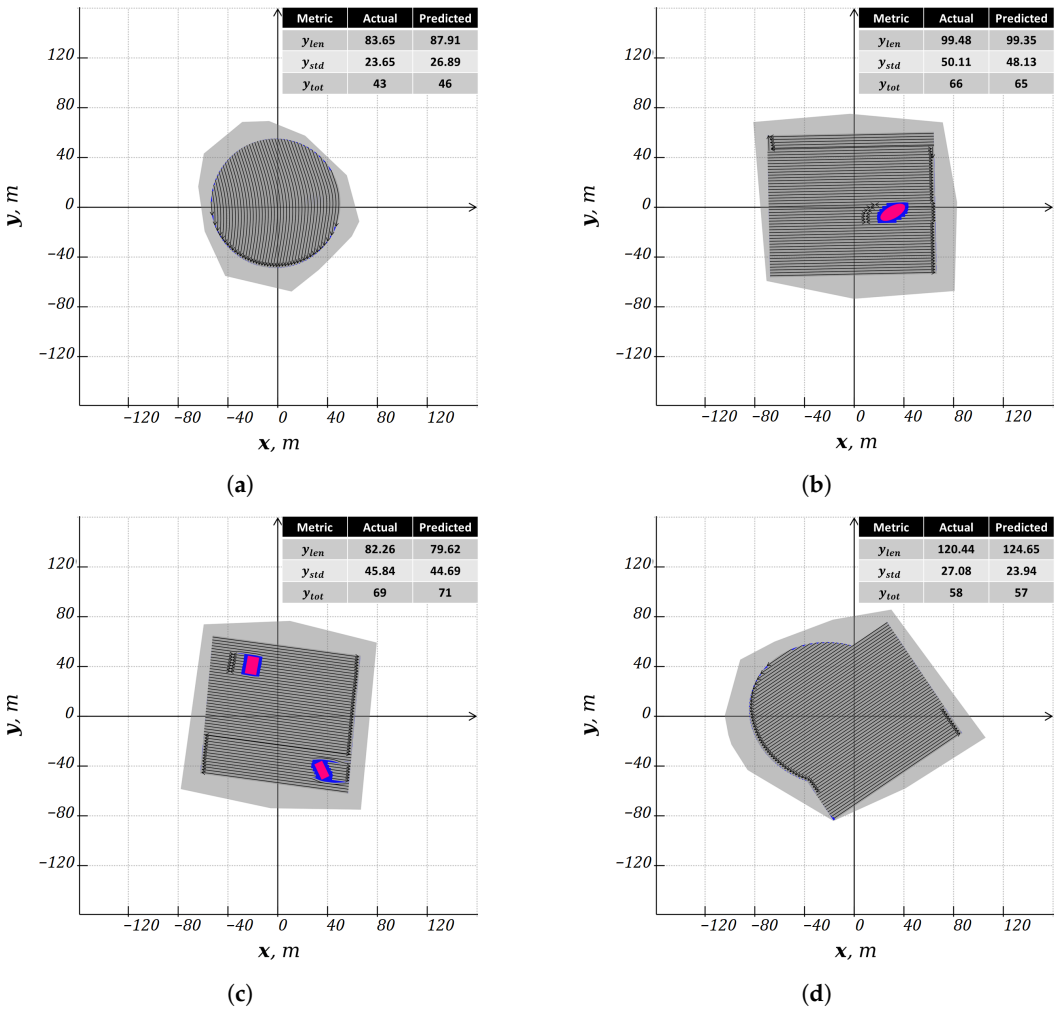
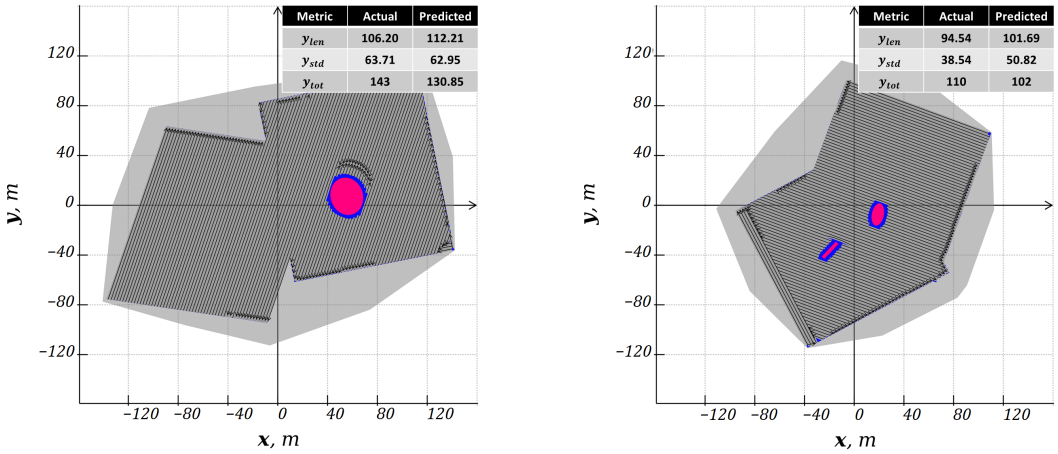


Figure 14. *Cont.*



(**e**) (**f**)

Figure 14. Examples of approximation modelmodel results for each DS type from Case 2 dataset: (**a**) $m = 0$, $k = 0$; (**b**) $m = 0$, $k = 1$; (**c**) $m = 0$, $k = 2$; (**d**) $m = 1$, $k = 0$; (**e**) $m = 1$, $k = 1$; (**f**) $m = 1$, $k = 2$.

4. Discussion

The CPP is a complex problem with many constraints related to the dimensions of the construction vehicle, environment, computational time, and resources. Thus, the estimation of CCP problem results has a practical value in planning the work schedule and resources, analyzing a large set of problems in order to choose the best decomposition, and other tasks. A strategy to train the CNN model and use it for this purpose has been suggested in this article. The CNN model is employed to predict the average length of the tracks, the standard deviation of the track lengths, and the number of tracks. The results show that CNN outperforms the linear regression model with area as input for both generated datasets with different levels of complexity. This demonstrates that the CNN model detects the relationship between the geometry and predicted features.

An obvious disadvantage of using CNN is the adjustment to the environment of the problem, such as dimensions of the construction vehicle, the analyzed geometry, and other values. That is, if a vehicle with another set of parameters is used or the geometry does not fit into the boundaries, the output value should not be interpreted directly as a number of tracks or track length. However, it still can provide a reasonable approximation of which decomposition case would result in a smaller number of tracks. In this research, the quality of the planned coverage path was evaluated with respect to the number of tracks, the length of tracks, and standard deviation. The lower number of tracks results in a lower number of stops during the movement of the construction vehicle. Similarly, the other parameters related to the track length enable us to evaluate whether the tracks are of a similar length and to prevent short tracks. Thus, the other coverage path or other decomposition can be chosen as optimal after the definition of the coverage path quality metric is modified. Although CNN model is used for fast approximation of model results, the CPP algorithm still takes a long time to process. The application of generative artificial intelligence models for the CPP problem can be an objective of future research.

5. Conclusions

In this paper, a CNN-based strategy to approximate the results of CPP has been proposed. The strategy enables us to evaluate a large number of configurations in a short period of time and select the best one for the considered problem. The approximation results have been compared with the results obtained using a linear regression model with area as input. Two generated datasets of different complexity levels have been used in the numerical experiments. The numerical experiments demonstrated that the suggested strategy outperforms the direct approach of linear regression for both analyzed datasets. Moreover, an example of CNN-based model application for finding the best decomposition

strategy has been demonstrated. The performance of the presented strategy depends on the geometry of samples used to train the CNN model. By determining the typical geometries, generating the representative dataset, and training the CNN model, the presented approach can be extended to be suitable for more complex CPP problems.

Although the algorithm was created to solve the CPP problem for the parking lot pavement, it can almost directly be transferred to CPP for agricultural purposes.

Author Contributions: Conceptualization, Dalia Čalnerytė and Rimantas Barauskas; Methodology, Andrius Kriščiūnas, Dalia Čalnerytė, Tadas Jurgutis, Dalius Makackas and Rimantas Barauskas; Software, Andrius Kriščiūnas, Dalia Čalnerytė, Tautvydas Fyleris, Dalius Makackas and Rimantas Barauskas; Validation, Andrius Kriščiūnas, Dalia Čalnerytė, Tadas Jurgutis and Dalius Makackas; Investigation, Andrius Kriščiūnas; Data curation, Tautvydas Fyleris and Tadas Jurgutis; Writing—original draft, Dalia Čalnerytė; Writing—review & editing, Tadas Jurgutis and Dalius Makackas; Visualization, Tautvydas Fyleris; Supervision, Rimantas Barauskas. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by UAB "Datava" RDE agreements nr. SV9-4435, SV9-4079. The APC was funded by Kaunas University of Technology.

Data Availability Statement: Data sharing not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Galceran, E.; Carreras, M. A survey on coverage path planning for robotics. *Rob. Auton. Syst.* **2013**, *61*, 1258–1276. [CrossRef]
2. Yakoubi, M.A.; Laskri, M.T. The path planning of cleaner robot for coverage region using Genetic Algorithms. *J. Innov. Digit. Ecosyst.* **2016**, *3*, 37–43. [CrossRef]
3. Nasirian, B.; Mehrandezh, M.; Janabi-Sharifi, F. Efficient Coverage Path Planning for Mobile Disinfecting Robots Using Graph-Based Representation of Environment. *Front. Robot. AI* **2021**, *8*, 1–19. [CrossRef] [PubMed]
4. Utamima, A.; Reiners, T.; Ansaripoor, A.H. Optimisation of agricultural routing planning in field logistics with Evolutionary Hybrid Neighbourhood Search. *Biosyst. Eng.* **2019**, *184*, 166–180. [CrossRef]
5. Seyyedhasani, H.; Dvorak, J.S.; Roemmele, E. Routing algorithm selection for field coverage planning based on field shape and fleet size. *Comput. Electron. Agric.* **2019**, *156*, 523–529. [CrossRef]
6. Nilsson, R.S.; Zhou, K. Method and bench-marking framework for coverage path planning in arable farming. *Biosyst. Eng.* **2020**, *198*, 248–265. [CrossRef]
7. Ai, B.; Jia, M.; Xu, H.; Xu, J.; Wen, Z.; Li, B.; Zhang, D. Coverage path planning for maritime search and rescue using reinforcement learning. *Ocean Eng.* **2021**, *241*. [CrossRef]
8. Cheng, S.; Shi, A.; Hu, Z.; Sun, L.; Liu, J. Partition Coverage Planning Method Based on Adjustable Rectangular Decomposition for Airport Pavement. In Proceedings of the 2021 the 6th International Conference on Control, Robotics and Cybernetics Partition, Shanghai, China, 9–11 October 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 106–112.
9. Nilsson, R.S.; Zhou, K. Decision support tool for operational planning of field operations. *Agronomy* **2020**, *10*, 229. [CrossRef]
10. Zhou, K.; Leck Jensen, A.; Sørensen, C.G.; Busato, P.; Bothtis, D.D. Agricultural operations planning in fields with multiple obstacle areas. *Comput. Electron. Agric.* **2014**, *109*, 12–22. [CrossRef]
11. Zhou, K.; Jensen, A.L.; Bochtis, D.; Nørremark, M.; Kateris, D.; Sørensen, C.G. Metric map generation for autonomous field operations. *Agronomy* **2020**, *10*, 83. [CrossRef]
12. Hameed, I.A. Coverage path planning software for autonomous robotic lawn mower using Dubins' curve. In Proceedings of the 2017 IEEE International Conference on Real-time Computing and Robotics (RCAR), Okinawa, Japan, 14–18 July 2017; pp. 517–522. [CrossRef]
13. Graf Plessen, M. Optimal in-field routing for full and partial field coverage with arbitrary non-convex fields and multiple obstacle areas. *Biosyst. Eng.* **2019**, *186*, 234–245. [CrossRef]
14. Seyyedhasani, H.; Dvorak, J.S. Reducing field work time using fleet routing optimization. *Biosyst. Eng.* **2018**, *169*, 1–10. [CrossRef]
15. Řezník, T.; Herman, L.; Klocová, M.; Leitner, F.; Pavelka, T.; Leitgeb, Š.; Trojanová, K.; Štampach, R.; Moshou, D.; Mouazen, A.M.; et al. Towards the development and verification of a 3d-based advanced optimized farm machinery trajectory algorithm. *Sensors* **2021**, *21*, 2980. [CrossRef] [PubMed]
16. Xie, H.; Jiang, K.; Song, K. An Optimal Path-planning Algorithm for Unmanned Rollers with Constraints on Roller Attitude. In Proceedings of the 2021 IEEE 10th Data Driven Control and Learning Systems Conference (DDCLS), Suzhou, China, 14–16 May 2021; Volume 21, pp. 964–971. [CrossRef]
17. Hameed, I.A.; La Cour-Harbo, A.; Osen, O.L. Side-to-side 3D coverage path planning approach for agricultural robots to minimize skip/overlap areas between swaths. *Rob. Auton. Syst.* **2016**, *76*, 36–45. [CrossRef]

18. Shen, M.; Wang, S.; Wang, S.; Su, Y. Simulation Study on Coverage Path Planning of Autonomous Tasks in Hilly Farmland Based on Energy Consumption Model. *Math. Probl. Eng.* **2020**, *2020*. [CrossRef]
19. Vahdanjoo, M.; Zhou, K.; Sørensen, C.A.G. Route planning for agricultural machines with multiple depots: Manure application case study. *Agronomy* **2020**, *10*, 608. [CrossRef]
20. Lewis, J.S.; Edwards, W.; Benson, K.; Rekleitis, I.; O'Kane, J.M. Semi-boustrophedon coverage with a dubins vehicle. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 5630–5637. [CrossRef]
21. Jeon, C.W.; Kim, H.J.; Yun, C.; Han, X.; Kim, J.H. Design and validation testing of a complete paddy field-coverage path planner for a fully autonomous tillage tractor. *Biosyst. Eng.* **2021**, *208*, 79–97. [CrossRef]
22. Kyaw, P.T.; Paing, A.; Thu, T.T.; Mohan, R.E.; Vu Le, A.; Veerajagadheswar, P. Coverage Path Planning for Decomposition Reconfigurable Grid-Maps Using Deep Reinforcement Learning Based Travelling Salesman Problem. *IEEE Access* 2020, *8*, 225945–225956. [CrossRef]
23. Sadek, M.G.; Mohamed, A.E.; El-Garhy, A.M. Augmenting Multi-Objective Genetic Algorithm and Dynamic Programming for Online Coverage Path Planning. In Proceedings of the 2018 13th International Conference on Computer Engineering and Systems (ICCES), Cairo, Egypt, 18–19 December 2018; pp. 475–480. [CrossRef]
24. Hameed, I.A.; Bochtis, D.; Sørensen, C.A. An optimized field coverage planning approach for navigation of agricultural robots in fields involving obstacle areas. *Int. J. Adv. Robot. Syst.* **2013**, *10*. [CrossRef]
25. Ranjitha, T.D.; Guruprasad, K.R. Pseudo spanning tree-based complete and competitive robot coverage using virtual nodes. *IFAC-PapersOnLine* **2016**, *49*, 195–200. [CrossRef]
26. Tan, C.S.; Mohd-Mokhtar, R.; Arshad, M.R. A Comprehensive Review of Coverage Path Planning in Robotics Using Classical and Heuristic Algorithms. *IEEE Access* **2021**, *9*, 119310–119342. [CrossRef]
27. Backman, J.; Piirainen, P.; Oksanen, T. Smooth turning path generation for agricultural vehicles in headlands. *Biosyst. Eng.* **2015**, *139*, 76–86. [CrossRef]
28. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]