Embedding Heuristic Rules in RRT Path Planning of Excavators

S.M. Langari and A. Hammad
School of Engineering and Computer Science, Concordia University, Canada
E-mail: s_langar@encs.concordia.ca, Hammad@ciise.concordia.ca

ABSTRACT

Improving safety and productivity of earthwork operations is of paramount importance, especially in congested sites where collisions are more probable. Automated Machine Guidance and Control technology is expected to improve both safety and productivity of earthwork operations by providing excavator operators with higher level of support regarding the path planning of excavators based on site conditions. However, in spite of the large number of studies related to automated path planning of excavators using well established algorithms from robotics, such as Rapidly-exploring Random Tree (RRT) and Probabilistic Roadmaps (PRM). These studies do not fully consider the engineering constrains of the equipment and do not result in smooth and optimal paths that can be applied in practice.

This paper aims to develop a new non-uniform RRT algorithm for the path planning of excavators by embedding heuristic rules and engineering constraints specific to excavators. The proposed method is implemented and tested in Unity 3D game engine environment for visualization and validation purposes. The initial comparative results with the basic RRT algorithm show that the proposed algorithm is able to find a high quality path in a shorter time.

Keywords - Earthwork, Excavator, Rapidly Exploring Random Tree

1 Introduction

The recent advances in computer visualization and simulation have provided a substantial platform for researchers in the construction domain to simulate and analyze construction processes at the macro and micro levels [1]. A large amount of research has been done in the area of construction simulation especially in the few past decades. Many research works have focused on simulation and visualization for path planning of large construction equipment such as tower cranes and hydraulic cranes. In these types of simulation, the safety and productivity of one or a few equipment participating in a specific task have been tested and analyzed [2, 4, 5, 6, 7, 8, 9, 13].

Among the recent sample-based algorithms, Rapidly-exploring Random Tree (RRT) proposed by La-Valle [3] is one of the quickest path planners and is widely used in micro-level simulation in research. RRT is an effective path planner especially for construction equipment, which in most cases has a large number of Degrees of Freedom (DoFs). Furthermore, the generic RRT path planner explores the search space to find a collision-free path without any considerations of the specific properties of construction equipment. This problem can be solved by embedding heuristic rules in the RRT path planner of large construction equipment considering operational and engineering constraints. For example, AlBahnassi and Hammad [4] proposed a framework for near real-time path planning of cranes considering their engineering constraints.

Another limitation of RRT is that it might not be fast enough for interactive and dynamic construction projects where there are many pieces of construction equipment working in a congested area. Besides, RRT and other sample-based path planners may be complete but they are not optimal. Completeness is the ability of the path planner to find the path when there is one [4, 7]. The jaggedness and non-optimality of the path clearly affect the cycle time of the equipment and will ultimately decrease its productivity. In a research for improving erection processes using robotic cranes, the initial generated path was not applicable to the crane due to the low quality of the path and the operational aspects of the equipment [6]. Several post-processing methods were applied to improve the quality of the path and to fit to the crane movement.

This research aims to improve the planning of construction equipment in congested sites considering the speed of the planner and quality of the path. The proposed method is expected to be exploited in near-real-time simulation that can support the excavator operator.

2 Literature review

Depending on the level of details and decision variables, the simulation of construction operations can be classified into the macro and micro levels [1]. At the

macro level, the focus is on high-level managerial concerns, such as the selection of the best combination of construction equipment and their deployment to ensure productivity and safety. In the macro-level simulation, in most of the cases, it is enough to find the path for the relocation of equipment in a 2D space using simple algorithms, such as A* [1, 7, 11]. However, the efficiency of A* decreases severely with increasing the dimensionalities of the problem. Soltani et al. [10] compared the performance of Dijkstra [10], A* and a genetic algorithm in finding the equipment path in a construction site considering three criteria: the length of the path, its safety and visibility. Then, the construction site is modeled by a rectangular grid, and the data sets representing the three criteria are assigned as layers to the site layout. The results showed that A* is capable of finding a near optimal solution more efficiently than Dijkstra. A* benefits from a heuristic function that guides it toward the goal. However, compared to Dijkstra, A* sacrifices completeness for efficiency. The level of this tradeoff is greatly affected by the heuristic function of A*. However, both algorithms suffer in high dimensionality problems resulting from the large number of DoFs of the equipment, which makes them unsuitable for such problems.

The path planning of construction equipment is a new type of micro-level simulation of construction operations that has benefitted from algorithms from robotics and computer science [2, 5, 7, 6, 8, 9, 10]. However, as explained in Section 1, these algorithms are not necessarily applicable to construction equipment. The simulation of construction operations at the micro level usually deals with the movement of construction equipment, and they are considered as robots with multiple DoFs. For example, the upper-structure of a typical hydraulic excavator has four DoFs (swing, boom up/down, stick in/out and bucket curling), and a crawler crane has seven DoFs [8]. Sample-based algorithms are proposed to handle this type of equipment with many DoFs. RRT, Probabilistic Roadmaps (PRM) [12] and their variations are capable of generating obstacle free and safe paths for construction equipment. RRT, firstly proposed by La Valle [3], is proven to be an efficient and reliable sample-based path planner in construction sites with dynamic and static obstacles, especially for equipment with a large number of DoFs [7].

As shown in Fig. 1, starting from the initial configuration as the root of the tree, the algorithm randomly and incrementally adds leaves to the tree. This process will continue until the predefined goal is reached. This agility and efficiency of RRT is mainly due to this randomness behavior; hence increasing the chance of success by reducing the search space into a set of randomly added nodes. In the case of excavators, the search space is a multi-dimensional space and each dimension represents one angle of a rotational joint as a DoF. The initial and goal configurations can be translated into two sets of angles. Then, random set of angles, representing a new configuration (i.e. position) in the Configuration Space (C-Space), would be generated. Based on this new random configuration, the tree would be lerped from the nearest leave of the tree toward that node. This process will continue until the goal is reached. The following pseudo code of Build-RRT adapted from [15] shows how the tree grows.

Build_RRT ()
T.init (x_{init});
For $k = 1$ to K do
    x_{rand} = Random_Configuration ();
    x_{near} = Nearest_Neighbor (x_{rand}, T);
    x_{new} = Extend (x_{near}, x_{rand});
    If (x_{new} =! null)
        T.Add (x_{new});
    End
Return T;

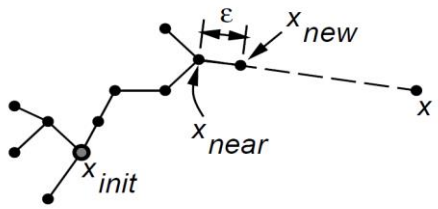


Figure 1. The tree in the C-Space [15]

Lin et al. [5] utilized RRT to simulate the trajectory of articulated construction equipment to help the site practitioners in site layout planning and replanning. The results showed that the proposed algorithm is capable of handling the problem in near real time to help the planner in decision-making and visualization. X, Y and θ are three DoFs of trucks. However, this type of car-like motion, called non-holonomic, should consider the engineering constraints of the equipment, which results in the position of equipment being a function of its vector velocity and the steering angle [5, 8]. In another research, Lin et al. [8] applied a modified version of bidirectional RRT to simulate the motion of a crawler crane with seven DoFs. The crawler crane is able to move with a load, and this motion consists of both holonomic and non-holonomic DoFs.

Due to the random nature of sampling-based algorithms such as RRT, the generated path is jagged and zigzag. This jaggedness results in an unnatural movement of the construction equipment, which means longer cycle time and lower productivity. This is considered as a limitation of these types of algorithms because although the path is obstacle free, it is still unsuitable from

practical and operational points of view. Zhang and Hammad [7] improved the quality of the path of mobile cranes by considering a cost function for the smoothness and time. The proposed algorithm is called *RRT-Connect-Connect-Mod*, which is a modified version of the bidirectional *RRT-Connect-Connect* algorithm. The results showed 11.51% improvement in the quality of the path in comparison with *RRT-Connect-Connect*. Lin et al. [8] improved the quality of the path of crawler cranes using a modified sampling and tree extension strategy, which resulted in a more practical path. Other studies aimed to generate a smoother and more practical path by improving the results of RRT using post-processing techniques [5, 6].

For path planning of tower cranes, Kang and Miranda [6] proposed three fast algorithms named, QuickLink, RandomGuess and QuickGuess. The idea behind the proposed algorithms is very close to RRT. They started searching the C-Space by randomly sampling collision-free nodes to ultimately find the obstacle free path. Then, they applied path-refining methods in consecutive steps to satisfy operational characteristics of the equipment. PRM is another sampling-based algorithm suitable for path planning purposes. Chang et al. [13] applied this algorithm to generate a safe plan of lift operations with single and dual mobile cranes. The result is satisfactory for near real-time applications while maintaining safety requirements. In most of the above-mentioned research, all obstacles are assumed to be static obstacles, and they may not be able to address the dynamic obstacles in a real construction site.

Real-time field data acquisition tools and replanning algorithms are two main assets, which have to be used in robotic construction equipment to make it able to promptly respond to dynamic environment of construction projects [14].

3 Proposed Method

3.1 Non-Uniform RRT

As explained in Section 2, RRT is proven to be a fast and effective path planner, especially with equipment that has a high number of DoFs such as excavators.

Fig. 2 shows the four DoFs of the upper structure of a typical excavator denoted by θ_j^i, where j is the index of the DoF and i is the index of the nodes on the path. θ_1 is the swing which is the angle of rotation of the upper structure with respect to the main body. A typical excavator has the ability to swing 360°. θ_2 is the angle between the boom and a horizontal plane. θ_3 is the angle between the boom and the stick. The angle between the bucket and the stick is θ_4. The limits of rotation for the last three DoFs can be extracted from the specifications of the equipment.

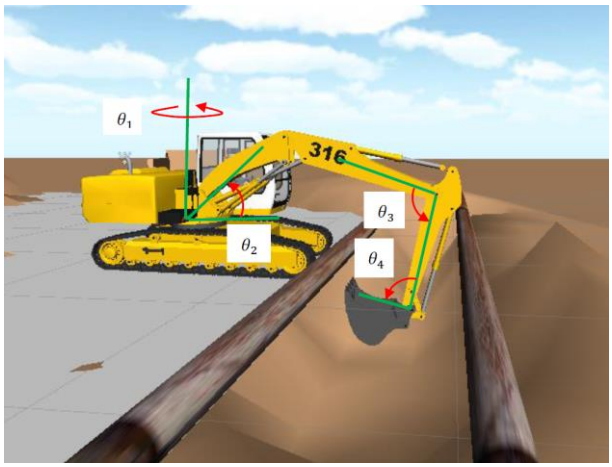


Figure 2. Four DoFs of upper structure of a typical excavator

Fig. 3 shows two simple 2D examples of C-Spaces. The initial and goal configurations are indicated and the other dots are randomly generated configurations. Fig. 3(a) shows a basic RRT where the random configurations are uniformly distributed making the tree grows symmetrically. This symmetrical growth is schematically shown by circular shapes. Fig. 3(b) represents the concept proposed in this paper for a non-uniform RRT where the random configurations are generated using a biased random number generator; making the tree grows biased toward the goal configuration instead of searching the whole C-Space.

The non-uniform RRT is expected to sacrifice completeness for a faster and more optimal path. This is similar to the comparison of Dijkstra and A* searching algorithms in Section 2. Dijkstra searches for the path considering all possible solutions; however A* benefits from a heuristic function to narrow down the search space. The advantages of the non-uniform RRT are:

- In contrast with the basic RRT that searches blindly in the C-Space, non-uniform RRT explores the C-Space toward the goal, resulting in a more efficient search.
- The path is expected to be smoother, since the tree grows toward the goal from the initial configuration.

On the other hand, due to its biasness, the non-uniform RRT has a lower possibility to success compared with the basic RRT, especially when there are too many obstacles between the initial and the goal configurations.

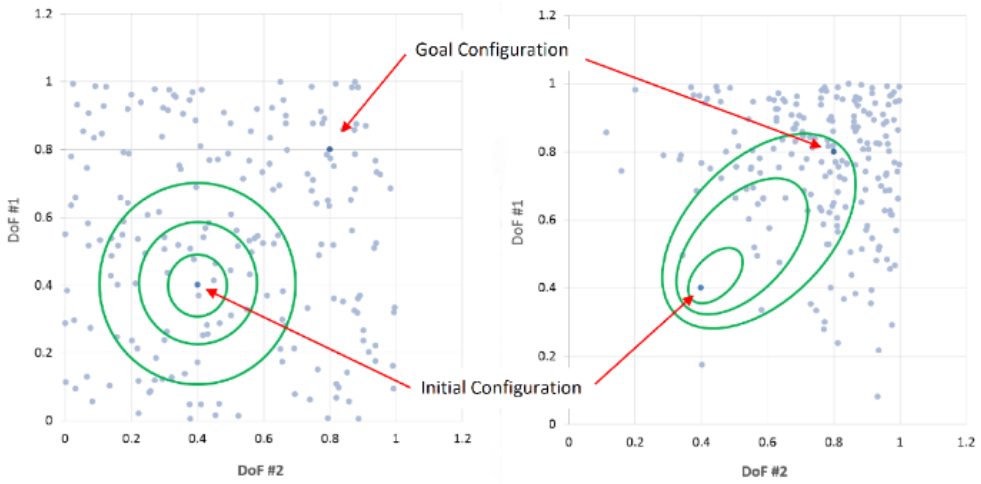


Figure 3. Schematic 2D representation of tree growth

3.2 Representing Excavator Cyclic Motion

The cyclic operation of an excavator can be divided into the following subtasks: *digging*, *moving to dump*, *dumping*, and *returning to the next dig*. An excavator's free motion planning is defined as the above-mentioned sequential subtasks, except *digging*. The present research focuses on the free motion and excludes the digging subtask because it requires considering the interaction between the bucket and the soil. Table 1 shows that a free motion planning of an excavator can be achieved in three steps. In the *moving to dump* subtask, the soil in the bucket should be maintained while moving toward the truck by constraining the tilt angle of the bucket with respect to the ground as shown in Fig. 4. The other three left DoFs can be biased in this subtask toward the goal configuration. When the excavator reaches the dump configuration, it has to unload the soil by curling out the bucket and applying a small inward rotation of stick. This subtask can be done using parametric path planning or using the non-uniform RRT algorithm because of its simplicity. The last subtask is to come back again to the dig point and all four DoFs of the excavator are free for path planning.

To generate the path using the non-uniform RRT algorithm, the initial and goal configurations must be defined for each subtask, which results in a set of sequential intermediate configurations to link the path segments of the free motion cycle.

3.3 Shape Parameters for the Non-Uniform RRT

The random number generator in the basic RRT generates random nodes equally and uniformly distributed in the whole C-Space. In order to create a biased tree for the non-uniform RRT, a new random number generator is required. There are many probability density functions (PDF) available for different engineering purposes. However, to generate biased random nodes in the C-Space, the Beta PDF is selected in this research since it can be defined in a finite range, which is compatible with the finite range of the DoFs, and the random numbers can be generated with the

Table 1. Biasing and constraining the DoFs for the subtasks within the free motion of excavator

Subtask	Initial Cnfg.	Goal Cnfg.	Remarks	Number of DoFs
Moving to dump	$\begin{bmatrix}\theta_1^0\\ \theta_2^0\\ \theta_3^0\\ \theta_4^0\end{bmatrix}$	$\begin{bmatrix}\theta_1^1\\ \theta_2^1\\ \theta_3^1\\ \theta_4^0\end{bmatrix}$	• Biasing the first three DoFs toward the goal configuration, which results in moving the boom up while swinging. • Bucket is constrained to maintain soil.	3
Dumping	$\begin{bmatrix}\theta_1^1\\ \theta_2^1\\ \theta_3^1\\ \theta_4^0\end{bmatrix}$	$\begin{bmatrix}\theta_1^1\\ \theta_2^1\\ \theta_3^2\\ \theta_4^1\end{bmatrix}$	• This subtask can be done using parametric path planning or using the non-uniform RRT algorithm because of its simplicity.	2
Returning to next dig	$\begin{bmatrix}\theta_1^1\\ \theta_2^1\\ \theta_3^2\\ \theta_4^1\end{bmatrix}$	$\begin{bmatrix}\theta_1^2\\ \theta_2^2\\ \theta_3^3\\ \theta_4^2\end{bmatrix}$	• Biasing the four DoFs toward the goal configuration	4

desirable biasness according to its parameters.

Equations (1) and (2) represent the Beta PDF in the span of [0, 1] and the associated expected value, respectively [17]:

$$PDF(x) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha,\beta)} \quad (1)$$

$$E(x) = \frac{\alpha}{\alpha+\beta} \quad (2)$$

Where α and $\beta > 0$ are shape parameters and x is the random variable. The PDF shape parameters determine the shape of the Beta PDF and its biasness. For example, $\alpha = \beta$ would result in a PDF with an expected value of 0.5. However, when $\alpha > \beta$, the PDF leans toward the lower limit of the range [0, 1] and it leans toward the upper limit of that range when $\alpha < \beta$. Depending on whether the goal configuration is bigger or smaller than the middle of the range of values for a specific DoF, a pair of α and β can be determined to result in a PDF with an expected value matching the goal configuration. Therefore, α and β can be determined for each DoF in each subtask to match the biasness of the Beta PDF with the goal of the subtask as in the following pseudo code:

If ($\theta_j^{Goal} > \theta_j^{middle}$)
    Find α & β to satisfy:
    $\theta_j^{Goal} = E(x) \times (\theta_j^U - \theta_j^L) + \theta_j^L$ And $\alpha < \beta$;
Else
    Find α & β to satisfy:
    $\theta_j^{Goal} = E(x) \times (\theta_j^U - \theta_j^L) + \theta_j^L$ And $\alpha \geq \beta$;
End

Where θ_j^{Goal}, θ_j^U and θ_j^L are the goal, upper limit and lower limit of the j^{th} DoF, respectively. θ_j^{middle} is the average of the upper and lower limits for the j^{th} DoFs. This algorithm results in an infinite set of α and β pairs, since it has one equation with two variables.

For the sake of simplicity, in this research we assumed that the smaller shape parameter is equal to one and the upper shape parameter can be calculated based on that. However, the high values of α and β will result in lower variance of the randomly generated numbers, which directly affects the dispersion of the nodes and consequently the shape of the tree. More disperse nodes mean more flexibility for the tree to grow, especially when the environment is full of obstacles, and vice versa.

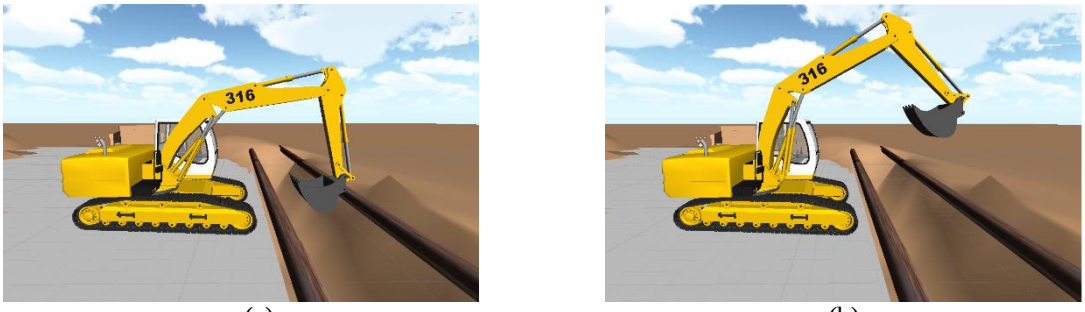
3.4 Evaluation Metrics

Metrics are necessary to evaluate the performance of the proposed method. According to previous research, metrics can be classified into two main classes, *computational efforts* (Effort Metrics), and *quality of the path* (Quality Metrics) [7, 8]. Quality Metrics evaluate the algorithm based on the quality of the generated path. A desirable algorithm is expected to generate a path that is energy efficient for the equipment. Besides, a desirable algorithm has to be computationally efficient, which is especially required when the environment is dynamic or multiple pieces of equipment are engaged.

3.4.1 Quality Metrics

(a) *Smoothness*: Due to the probabilistic nature of RRT, the generated path is jagged in the C-Space. This results in an unnecessary, inefficient and even impractical movement of the excavator. Hence, it will result in a lower productivity. Adopted from previous research, smoothness can be expressed by the following equation [7]:

$$S = \sum_{i=1}^{n-1} \sqrt{\sum_{j=1}^{m} (\theta_j^{i+1} - \theta_j^i)^2} - \sqrt{\sum_{j=1}^{m} (\theta_j^{Goal} - \theta_j^{Initial})^2} \quad (3)$$



(a) (b)

Figure 4. While loaded, the bucket has to be constrained with respect to the ground to maintain the soil

Where m is the number of DoFs of the equipment and n is the number of nodes on the path. θ_i^j and θ_{i+1}^j are the j^{th} DoF of nodes i and $i+1$, respectively. The second term is added to the equation proposed in [7] since we need a metric independent from the initial and goal configurations. This modified metric only calculates the jaggedness of the path and it does not take into account the inherit difference between the initial and goal configurations. Besides, in this research we apply another modification to the equation proposed in [7] which makes the metric independent of the length of the path.

$$S' = \frac{\sum_{i=1}^{n-1}\sqrt{\sum_{j=1}^{m}\left(\theta_j^{i+1}-\theta_j^{i}\right)^2} - \sqrt{\sum_{j=1}^{m}(\theta_j^{Goal}-\theta_j^{Initial})^2}}{\sqrt{\sum_{j=1}^{m}(\theta_j^{Goal}-\theta_j^{Initial})^2}} \quad (4)$$

(b) *Length of the path*: In a simple three-dimensional translation of an object, the length of path can easily be calculated as the sum of all relocations along the path. This length can be defined both in 3D-space and C-Space. Lin et al. [8] considered the movement of the load for a crawler crane as a metric for measuring the path length. This formulation has a physical meaning for the case of a crawler crane, which is compatible with different types of its DoFs. But in the case of excavators, it has to be modified to reflect the excavator's rotational DoFs. The following formulation is adapted from Lin et al. [8] to measure the length of the path for excavators:

$$L = \sum_{i=1}^{n-1}\sum_{j=1}^{m} r_j \times |\theta_j^i - \theta_j^{i+1}| \quad (5)$$

Where m is the number of DoFs, n is the number of nodes in the path, and r_j is the effective arm for that rotational joint DoFs. It should be noted that r_j is not constant and is a function of the overall configuration of the excavator.

3.4.2 Effort Metrics

The *time* it takes for the algorithm to find the collision-free path is the primary measure for computational efforts. Another type of effort metrics is the *number of nodes* in the tree generated by the algorithm to find the path. This metric is independent from the processor performance of the computer. The *number of rejected nodes* is the number of nodes that could not be added to the tree due to collision. This is another metric that shows computational efforts; since collision detection is a heavy function and requires considerable computational efforts [8].

4 Case Study

In order to test the proposed model, a real case study is simulated in a game simulation environment. As shown in Fig. 5, the operator has to carefully navigate the bucket between two pipelines in order to dig the soil. After digging, the soil should be carried to the dump position. The site is congested with other pieces of equipment, workers, etc. Therefore, the operator has to pay special attention to avoid collisions. For example, when a worker is in the excavator's workspace, the operator brings the stick toward the body of the excavator to avoid collisions.

4.1 The Simulation

As shown in Fig. 6, the same scenario is implemented in Unity game engine [16] to test the efficiency of the proposed method. The proposed algorithm is implemented in C# scripts, which are connected to the scene game objects to find the collision-free path for the excavator. In this simulation, the excavator has two models: a visual model (realistic model) and a simplified model. The simplified model is composed of a few basic convex volumes representing the parts of the excavator and is used for path planning calculations. When the path is found, the visual model is used to visualize the motion of the excavator.

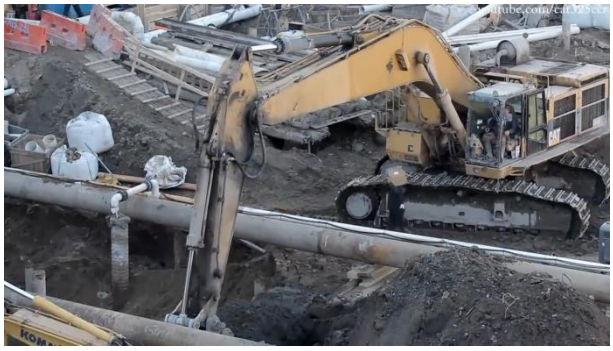


Figure 5. Operator runs the excavator carefully to dig the earth between two pipelines in the congested site

Using the simplified model to find the path has two main benefits. First, collision detection for this model is easier; hence it greatly reduces the computational efforts of RRT. Second, a buffer can be added around the simplified model to conservatively consider safety [4]. The user can define the initial and goal configurations in the simulation. After digging, when the bucket is full, it has to remain constrained in order to maintain the soil. Considering these issues, the proposed method has been tested in the game environment and it showed

satisfactory results in finding a collision-free path. The quantitative comparison between the proposed method and the basic RRT is given in the following section.

4.2 Comparing Non-uniform RRT with Basic RRT

In order to get reliable comparison of the evaluation metrics explained in Section 3.4, both the non-uniform and basic RRT path planners are executed 50 times on a computer with a CPU of 3.40 GHz and memory of 6 GB. Table 2 represents the average results of each metric. The results show a significant improvement in computational efforts and the quality of the path.

Table 2. Comparison of evaluation metrics

Path planner	Average No. of Nodes	Average run duration (ms)	Average length (m)	Average Smoothness
Basic RRT	716.2	186	1057.5	268.9
Non-uniform RRT	40.0	12	453.3	41.6

5 Conclusions and Future Work

In this paper, a new non-uniform RRT algorithm is proposed to solve the path-planning problem of the excavators by embedding heuristic rules for biasing the generation of nodes of the RRT. A scenario is extracted from the real world to validate the proposed algorithm. The excavator has to pass the loaded bucket between two pipelines and move it above the truck to dump. The simulation showed that the algorithm is capable of efficiently finding the collision-free path from a given initial to a goal configuration.

From the qualitative point of view, the movement of the excavator is quite smooth without any zigzag behavior. Besides, the movement keeps the bucket and the stick near the body of the excavator when moving back to the next dig. This results in a less workspace and safer actions. The algorithm is on average 15 times faster than the basic RRT and is capable of generating a significantly smoother final path. Besides, it requires much less energy for the excavator to run that path since the length of the path is less than half that of the basic RRT. The proposed algorithm can be applied in dynamic path planning with multiple excavators working in a tight area. The proposed path planning method can support the excavator operator in a way to increase the safety and productivity of construction projects.

Future work will consider the engineering constraints related to the speed of movement of the parts of the excavator and their effects on path planning. Another area of future work is to address the dynamic safety challenges of the construction site. For example, when a worker enters the workspace of the heavy equipment, it has to be able to re-plan its path.

Acknowledgments

The authors would like to acknowledge the help of Mr. Homam Albahnassi in implementing the RRT algorithms.

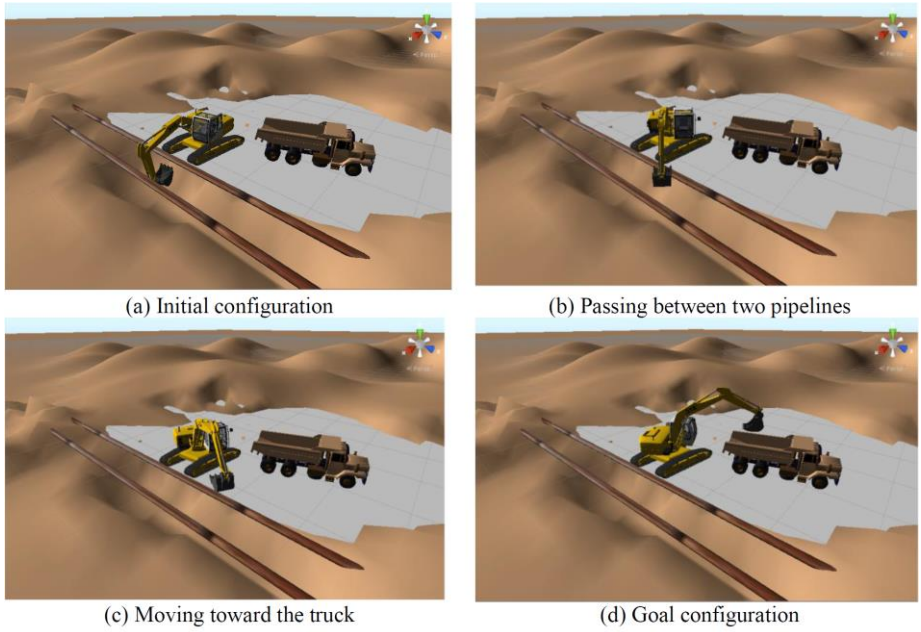


Figure 6. Simulated excavation environment with two pipelines and a truck

References

[1] Hammad A., El Ammari K., Langari M., Vahdatikhaki F., Soltani M., AlBahnassi H., Paes B. Simulating macro and micro path planning of excavation operations using game engine. In *proceedings of Winter Simulation Conference*, pages 4111-4112, Savannah, USA, 2014.

[2] Marzouk M. and Ali H. Modeling safety consideration and space limitations in piling operations using agent based simulation. *Journal of Expert System with Application*, 40, 4848-4857, 2013.

[3] La Valle S.M. Rapidly-exploring random trees: a new tool for path planning. Computer science department, Iowa state University. On-line: <msl.cs.uiuc.edu/~lavalle/papers/Lav98c.pdf> (June 25, 2010).

[4] AlBahnassi H. and Hammad A. Near real-time motion planning and simulation of cranes in construction: Framework and system architecture. *Journal of Computing in Civil Engineering*, 26(1): 54-63, 2011.

[5] Lin J.J., Hung W., Kang S. Motion planning and coordination for mobile construction machinery. *Journal of Construction Engineering and Management*, 2014.

[6] Kang S. and Miranda E. Planning and visualization for automated robotic crane erection process in construction. *Journal of Automation in Construction*, 15: 398-414, 2006.

[7] Zhang C. And Hammad A. Improving lifting motion planning and re-planning of cranes with consideration for safety and efficiency. *Journal of Advanced Engineering Informatics*, 26: 396-410, 2012.

[8] Lin Y., Want X., Wu D., Wang X., Gao S. Lift path planning for a nonholonomic crawler crane. *Journal of Automation in Construction*, 44: 12-24, 2014.

[9] Kim S.K., Russell J.S., Koo K.J. Construction robot path-planning for earthwork operations. *Journal of Computing in Civil Engineering*, 17(2): 97-104, 2003.

[10] Soltani A.R., Tawfik H., Goulermas J.Y., Fernando T. Path planning in construction sites: performance evaluation of the Dijkstra, A*, and GA search algorithms. *Journal of Advanced Engineering Informatics*, 16: 291-303, 2003.

[11] Hart P.E., Nilsson N.J., Raphael B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transaction on Systems Science and Cybernetics*, 4(2): 100-107, 1968.

[12] Kavraki L.E., Svestka P., Latombe J.C., Overmars M.H. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4): 566-580, 1996.

[13] Chang Y.C., Hung W.H., Kang S.C. A fast path planning method for single and dual crane erections. *Journal of Automation in Construction*, 22:468-480, 2012.

[14] Zhang C., Hammad A., Bahnassi H. Collaborative multi-agent systems for construction equipment based on real-time field data capturing. *Journal of Information Technology in Construction*, 14: 204-228, 2009.

[15] La Valle S.M., Kuffner J. Rapidly-exploring random tree: progress and prospect. 2000.

[16] Unity game engine. On-line: http://unity3d.com, Accessed: 28/01/2015.

[17] Montgomery D.C. and Runger G.C. *Applied statistics and probability for engineering*, volume 4(12). John Wiley & Sons, United States of America, 2014.