CSCE Annual Conference

Growing with youth – Croître avec les jeunes

Laval (Greater Montreal)

June 12 - 15, 2019

COMPARISON OF LIFT PATH FINDING ALGORITHMS FOR MOBILE CRANE OPERATION IN HEAVY INDUSTRIAL PROJECTS

Serim Park[1,3], SangHyeok Han[2]
[1] Concordia University, Canada
[2] Assistant Professor, Concordia University, Canada
[3] Serimle.park@gmail.com

Abstract: Heavy industrial projects are constructed by modules prefabricated in factories, transported to sites and installed by mobile cranes. The mobile crane operations are planned by the following four studies: (1) crane location selection; (2) crane type selection; (3) crane support design; and (4) crane lift path planning. One of the key success factors in heavy industrial projects is the lift path planning because the heavy industrial construction projects generally involve congested sites that can lead to crane accidents causing production reduction. In this respect, researchers have been paying attention to plan safe and practical crane lift paths using optimization algorithms such as A*, RRT, and genetic algorithms. However, comprehensive comparison of these algorithms is not examined yet based on features of heavy industrial projects and practical rules of mobile crane operation. This research compares the algorithms for the lift path planning of the mobile crane in the modular-based heavy industrial project to find the competent method which searches collision-free lift paths with the lower operation cost and less computation time. The algorithms' results are compared by the measurement metrics such as computation time, number of movements, success rate, linear traveling distance, and crane operation time. The proposed comparison is implemented in a case study that includes a considerable number of module lifts installed by the mobile crane. This comparison will show which algorithm is more effective for the crane path planning in heavy industrial projects and suggest the direction of further research.

1 INTRODUCTION

Modular construction, which is off-site construction that delivers the preassembled modular units to the site, is increasingly recognized as a cost-effective method that reduces onsite labor, material waste, and construction time while achieving safety and productivity. Because of these benefits, a modular approach has been widely implemented in high-rise building and heavy industrial projects (Han et al. 2015). Mobile cranes are commonly used to handle the modules on site because of its high capacity, thus utilization of a mobile crane is a key success factor in modular-based heavy industrial projects. However, insufficient planning and analysis of crane utilizations can cause less productivity, and also result in accidents with high fatality rates. According to a report on the causes of death in crane-related accidents ("ELCOSH" 2010), at least 71% of all crane-related fatal accidents are involved with mobile cranes, which are caused by crane collapses (39 %), overhead power line contacts (14%), struck by crane load (14%), struck by other crane parts (11%), and other causes such as highway incidents, falls, and caught in/between(23%). In this respect, proper lift path planning of mobile cranes is important in heavy industrial projects to improve safety and productivity.

The conventional manual lift analysis by an experienced lift engineer is not suitable in a heavy industrial project, normally consisting of huge number of units, which requires consideration of all alternates rapidly

without errors (Lei et al. 2015). Numerous studies have attempted to develop automatic path planning systems at the theoretical level with rare practical implementations due to the complexity of crane constraints and construction environment. There are three main factors to achieve the successful lift path planning: efficiency, solution quality, and success rate (Cai et al. 2016). Lift path planning with hill climbing, A* algorithm, and genetic algorithm (GA) were developed in the configuration space (C-space), that can represent high DOF environment effectively, for the single crane and cooperative cranes (Sivakumar et al. 2003; Ali et al. 2005). Cai et al. (2016) proposed a parallel GA applying hybrid configuration concepts to handle complex site conditions that considers various costs such as energy cost, human cost and workability of operator while overcoming collisions and the limitation of the operations. Research by Chang et al. (2012) implemented a probabilistic road-map (PRM) method for the crane erection planning in 2.5D environment that achieves a near real-time solution. Rapidly exploring Random Trees (RRTs), as one of the popular randomized path planning algorithms in robotic, has also been introduced (Lin et al. 2014).

Previous attempts have been made to implement various algorithms to suggest local/global optimal solutions for lift path planning of mobile cranes based on outputs of computation time, travelling distance of the lifted object, and crane configurations. However, comprehensive comparison of these algorithms are not yet executed. In this respect, this paper represents an on-going project which compares optimal lift paths designed by three popular path finding algorithms (A* search, Rapidly-exploring random tree, and Genetic algorithm) in a real case. The comparison is executed in the Python environment. The result of this paper will propose not only the crane lift path algorithm fitted for heavy industrial projects but also the direction of future research in the crane lift path planning.

2 METHOLOGY

2.1 Problem Structure

The objective of the crane lift path planning is to find sequences of crane movements based on the consideration of various pick positions and set positions. There are several requirements to be satisfied in order to be considered as a feasible path in this paper: (1) the movements should follow kinematics constraints of the crane represented as degree of freedoms (DOF) in permissible range; (2) no collision between the lifted object and obstacles which are objects already installed; (3) the total weight of lifted object should not exceed the allowable crane capacity that is provided by a manufacturer's capacity chart; and (4) dynamic site layout applying scheduled sequences of selected object should be reflected .

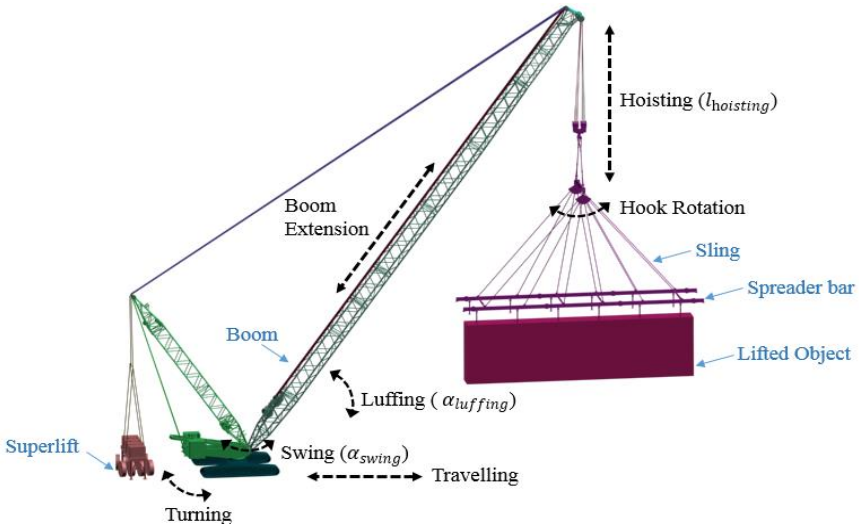


Figure 1: Relationship of lattice-boom mobile crane configurations

Due to the high DOF in mobile crane as shown in Figure 1, representing a crane configuration using the configuration space concept with active DOFs will be beneficial to solve the path planning problem. Depending on the mobility of the crane, there are two types of crane operations; (1) pick from fixed operation

(PFP), and (2) pick and walk operation (PWO), which includes turning and travelling as parts of active DOF. The scope of the current paper considers the lifting method with PFP, which is usually preferred from the practitioners perspective since it has less collision errors (Han et al. 2016). In addition, hook rotation and boom extension are practically not allowed during the lifting procedure. Accordingly, the single mobile crane has three active DOF in this paper and the corresponding configuration set is expressed as Eq. 1 and Eq. 2 as shown below:

$$[1]\ P_j = \{M_i\}_{\ i=0,1,\ldots,n-1}$$

$$[2]\ M_i = (\alpha_{swing},\ \alpha_{luffing}, l_{hoisting})$$

Where P is the lifting path, represented by a set of configurations, j is the module ID, n is the total number of configurations, M_i is i^{th} configuration in the path, α_{swing} is the swing angle in degrees, $\alpha_{luffing}$ is the luffing angle in degrees, and $l_{hoisting}$ is the hoisting length in feet. The module ID of the lifted object imports the corresponding information of dimensions, weight, set location, and the scheduled lifting sequence from the database needed for: (1) building the obstacle environment for the current lifted object to check collision; (2) checking the crane capacity and safety factor; and (3) visualizing results.

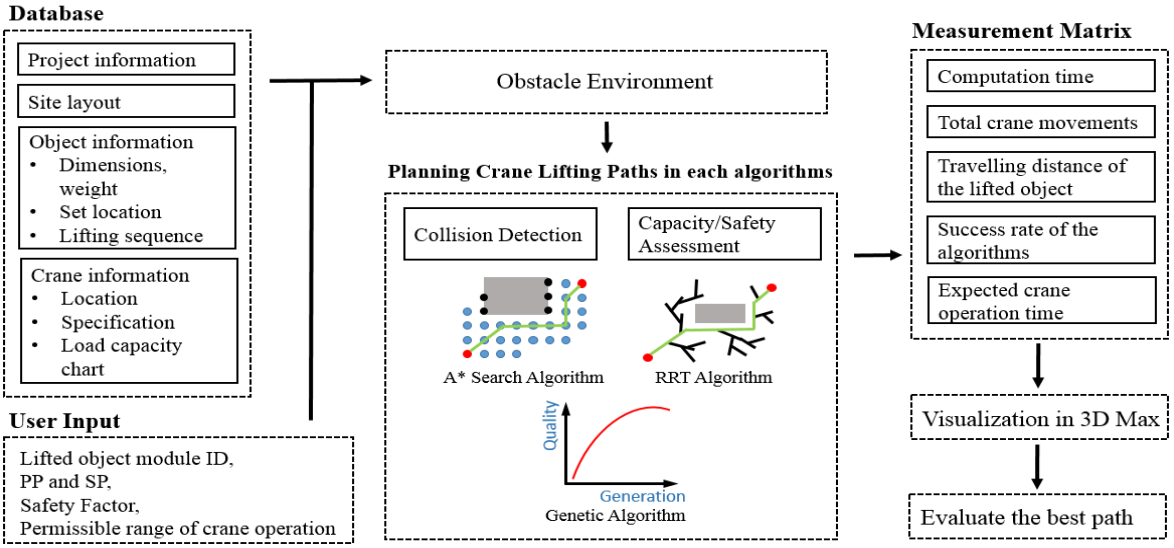
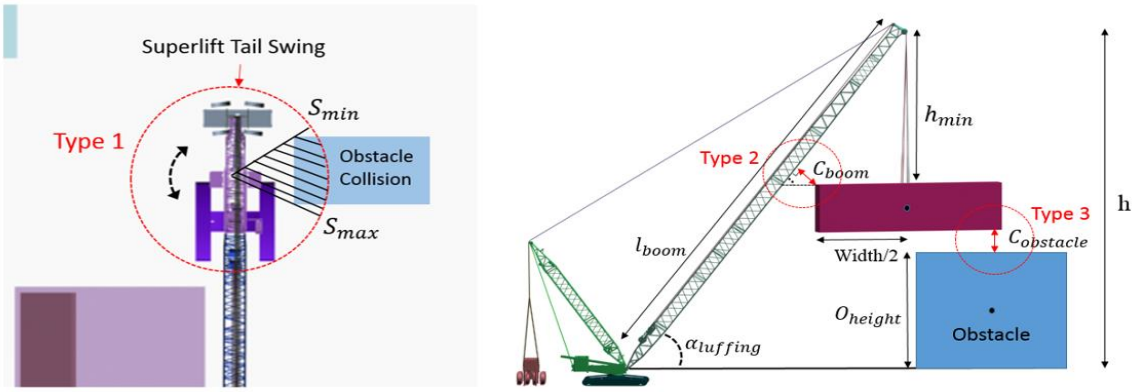


Figure 2: Process overview of the proposed methodology

In order to apply multiple algorithms in the specified problem of path finding in mobile cranes, it is mandatory to set up the base structure for a reasonable comparison and a flexible implementation. Figure 2 illustrates the base structure of the proposed methodology. The database that saved the information of the project, lifted objects, and the crane is used to generate the crane lifting path by A* algorithm, RRT algorithm, and GA, respectively. The goal of the crane path finding problem is to search the path consisting of crane movements without the violations regarding the capacity, the safety factor, collisions, and crane kinematic constraints.

Crane capacity and safety factor assessment were prepared mainly according to the procedure introduced by Han et al. (2017) based on the calculation of the required lifting weight (W_{Total}) and the working radius ($\mathrm{R_A}$). The safety factor is calculated by W_{Total} and the gross capacity at $\mathrm{R_A}$ is obtained from the capacity chart provided by manufacturers. In this paper, the crane operation with the safety factor exceeding 85% is considered as an unsafe operation. Therefore, the crane configuration within the allowable working radius and safety factor of 85 % is considered as a safe operation. This safety factor is also used to calculate the operation time (cycle time) of path by indicating the speed of the movement depending on the corresponding safety factor with a concept that the speed of movement is influenced by the safety factor, which means lower safety factor causes high speed and vice versa. To reflect the realistic analysis,

preparation time to convert the movements is considered by applying penalty time matrix of crane operations.



(a) Type 1 (b) Type 2 and Type 3

Figure 3: Three types of potential collision

During mobile crane operations, there are three types of potential collision: (1) Type 1: between the crane configurations and obstacles represented in Figure 3(a); (2) Type 2: between the crane configurations (mostly boom) and the lifted object shown in Figure 3(b); and (3) Type 3: between the lifted object and obstacles illustrated in Figure 3(b). Type 1 collision is prevented by locating the crane at the collision-free region where there are no obstacles in the superlift swing range or limiting the permissible range of swing angle between S_{min} to S_{max} as shown in Fig. 3(a). Type 2 collision is avoided by locating the lifted object distantly from obstacles or limiting the permissible range of hoisting length by satisfying Eq. 3 and Eq. 4.

$$[3]\ h_{min} = \left(\frac{C_{boom}}{\sin\ \alpha_{luffing}} + \frac{O_{width}}{2}\right) \tan\ \alpha_{luffing}$$

$$[4]\ h_{max} = h - O_{height}$$

Where h_{min} is the minimum permissible hoisting length, C_{boom} is the clearance between the boom and the lifted object set by users, $\alpha_{luffing}$ is the luffing angle, O_{width} is the width of the lifted object, O_{height} is the height of the lifted object, h_{max} is the maximum permissible hoisting length, and h is the vertical distance between boom top to the ground level.

Type 3 collision is solely examined in the algorithms with an assumption that lifted object do not rotate during the operation, which means that the lifted object is oriented in the same direction during the entire process. Collisions can be detected by identifying the interruption between lifted object and the obstacles by comparing their minimum and maximum x, y, and z coordinates. Figure 4 illustrates collision conditions when box A and box B are defined by minimum and maximum values in x, y, z axis (Ax.Min, Ax.Max, Ay.Min, Ay.Max, Az.Min, Az.Max, Bx.Min, Bx.Max, By.Min, By.Max, Bz.Min, Bz.Max).

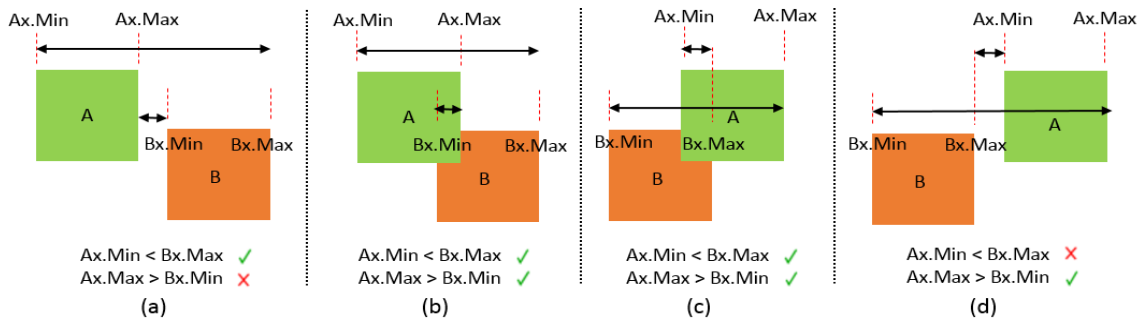


Figure 4: Examples of the collision identification

To have collision in 3 dimensions, y and z dimensions also have to be overlapped. In this method, 3D collision detection is shown below as pseudocode in Figure 5. The value of the clearance is considered at the minimum and maximum coordinates of the lifted object to guarantee the safe operation. Also, applicable obstacles present in the dynamic site layout are loaded from the database by the sequence of the lifted object. For example, if the module with sequence number 10 is lifted, the modules with sequence number 1 to 9 are loaded to build the obstacle environment.

```
CollisionDetection
O: list of obstacles
L: Lifted object
For each obstacle o in O do:
        If (ox.Min < Lx.Max) and (ox.Max > Lx.Min) and   ← X axis
           (oy.Min < Ly.Max) and (oy.Max > Ly.Min) and   ← Y axis
           (oz.Min  < Lz.Max) and (oz.Max > Lz.Min):     ← Z axis
                Collision = True
        End if
```

Figure 5: Pseudo code of process flow in collision detection

2.2 A* Search

A* algorithm is a global search method that holds information of previous configurations with the heuristic value until it reaches the goal node (Sivakumar et al. 2003). It starts from defining the start node and the end node, which are the pick configuration and the set configuration respectively. After this, the OPEN list that contains the start node and the empty CLOSED list are formed. The OPEN list will keep the expanded nodes that are not yet examined, and the CLOSED list will keep the nodes that are already examined to check whether the new expanded nodes from the current node are generated before. If the OPEN list is not empty, nodes in the OPEN list are evaluated in ascending order of the estimated total cost value expressed in Eq. 5:

[5] f(n) = g(n) + h(n)

Where g(n) is the cost of the path from the start node to the n^{th} node (current node), h(n) is the heuristic function that estimates the cost of the cheapest path from the n^{th} node to the goal node, and f(n) is the total cost of the path. Here, the cost is simply the distance that the lifted module has traveled.

After calculating the f(n) value of each nodes in the OPEN list, the OPEN list is sorted by f(n) value in ascending order. Therefore, the first node in the OPEN list has the lowest cost, which means that it is the best node to perform the cheapest path in the current OPEN list. This node is then removed from the OPEN list and added to the CLOSED list. If this node is equal to the set node, it becomes the feasible, near optimal path solution. If this node is not equal to the set node, neighbor nodes of this node are used to expand the search area by added it in the OPEN list. Generations of neighbors are done by considering the incremental value in the crane operation, which is one unit of each movement. For each neighboring nodes, several feasible checks are conducted: (1) check if the crane movement is within the permissible range of the crane kinematic constraints, (2) check if there is any collisions, (3) check if this node is in the CLOSED list, meaning the same node has already been checked, and (4) check if any nodes in the OPEN list have same configurations while having same or bigger g(n). This way, when the configurations are the same, only the nodes with cheaper costs will be saved in the OPEN list for further evaluations. This method can reduce the process time by removing superfluous calculations. If the neighboring nodes pass all these checks, those nodes are appended to the OPEN list and the same procedure is repeated until the end node is reached.

2.3 RRT

A tree is constructed starting with the start node, the tree is then expanded by 1 unit for each crane movements according to the sampling strategy. The sampling strategy has a great effect on the quality of

the path and the efficiency of the algorithm because the random sample controls the direction of the tree growth to find the optimal path while avoiding collisions and the local confinement (Lin et al. 2014).

With the probability p (sample rate), the tree expands towards the targeted node, and with the probability p-1, it expands towards a random sample node generated randomly within the exploring space. It is observed that 5 – 10 % of probability is suitable as p to bias towards the goal node while 100% of probability possibly gets the node stuck by failing to avoid obstacles. This strategy keeps the tree expanding to unsearched area with the tendency to reach the targeted node while not trapping it in local areas. The nearest node of the sample node is obtained from the tree. There are two ways to select an expanded node among the six possible movements from the nearest node: (1) the node that has the smallest distance to the end node; and (2) the node that has the smallest angular difference. Both methods are implemented in the case study and compared to verify which one results in a better path solution. After selecting the candidate of the expanded node, collision is checked. If there is a collision, the sampling process is repeated. If there is no collision and the expanded node is equal to the end node, the result path is printed. If the expanded node is not equal to the end node, it is added to the tree and the same procedure is repeated until it reaches the targeted node.

2.4 Genetic Algorithm

An initial generation of paths is randomly built within the permissible range of crane operations. Each path is evaluated by a fitness logic chart as illustrated in Figure 6. The fitness evaluation is critical in GA because it determines the quality of paths and check the constraints of crane operation. The fitness value (f_i) represents the suitability of the path. If any configurations in the path have collision or the length of the path and the number of movements are more than the maximum setting value, f_i becomes 0. If the individual has no violation in these three conditions, the last landing node of the path will be checked if it arrives at the end node. The f_i of the path landed at the goal node will be evaluated by the length of the path, the number of movements, and the scaling factor (λ_1). Lastly, the constraints of the crane's permissible range is evaluated. If the path includes any movements that are not within the permissible range of crane operations, the scale factor (λ_2) will be deducted from the overall f_i. This fitness logic was tested and it verifies the improved path for the crane lift path planning over generations.

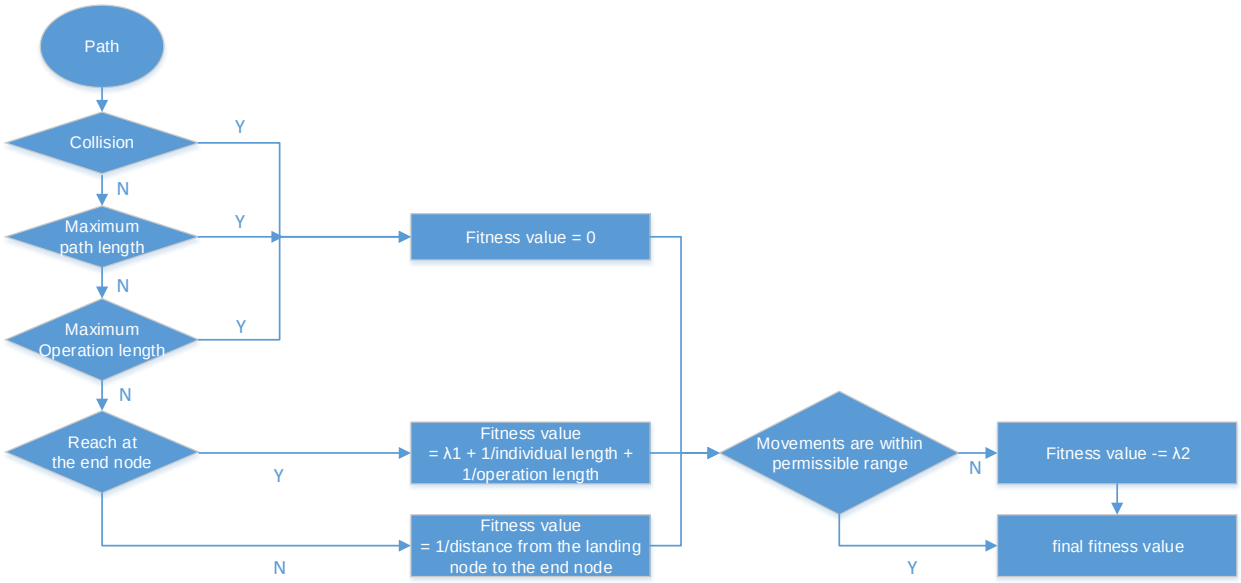


Figure 6: Fitness Evaluation

To reproduce the next generation, the paths with high f_i from the previous generation are selected with the probability of P_{best} to evolve the results by generations. Then, (1-P_{best}) of paths are randomly chosen to form a list of breeders to generate the next generation by using crossover and mutation. The crossover

enables the paths to evolve towards a local optimal solution and the mutation prevents it from being stuck in one place. From the list of breeders, two paths are randomly selected as parents. The crossover rate (r_c)) is a normally high value between 80% and 95% to guide the direction of the evolution. There are several ways to carry out the crossover based on the single point, multi-point, uniform, and arithmetic. The multi-point crossover method, which alters the segments of parents in multiple points, is adopted in this paper. If crossover is only used to form the next generation, the path will have limited evolution opportunities that could result the confinement or less optimal path. Therefore, a mutation process that generates random tweaks in the individual is used to guarantee the diversity of the genetic population by exploration. The mutation rate (r_m) is set to a low value between 0.005 and 0.5 because the high probability of mutation leads pure random searches that slows down the evolution while adequate mutation rate enables the prevention of the convergence of the path (Srinivas and Patnaik 1994). The reproduction process is repeated until it reaches the number of generations set by users.

2.5 Measurement Metric

The criteria for comparing the algorithms are (1) the computational time to run the algorithm; (2) the travelling distance of the lifted module; (3) the number of total movements and each movement; (4) the cycle time of the crane operation; and (5) the success rate, which shows the probability to find the solution. RRT and GA require to be run multiple times due to the nature of randomness. Therefore, RRT and GA's success rate will be a percentage of pass/fail of the multiple iterations and A * will a single pass/fail.

3 CASE STUDY

The case study is based on an industrial modular project by PCL industrial Management in Alberta, Canada, which includes a considerable number of module liftings with a crawler crane (Demag CC 2800). The proposed methodology is implemented in a Visual Studio Code environment with Python, and Matplotlib is used for plotting to visualize the lift paths of each algorithms. The database of PCL Industrial Management, Inc. is used as a primary input data stream, which includes: (1) module information (coordinates of set position, geometric size, and weight); (2) installation sequence; and (3) crane information (capacity in each radius and crane configurations).

Module 9 is selected as the lifted object. The difficulty of path planning for module 9 is relatively low because the installation level is ground floor and there are no obstacles in a straight line between the pick point and the end point. The scheduled sequence of module 9 is 104. Table 1 shows the permissible ranges of crane movements based on crane specifications and calculations from Eq. 3 and Eq.4.

Table 1: Crane permissible range

	Swing (degrees)	Lifting (degrees)	Hoisting (feet)
Lower Limit	0	10	13
Upper Limit	360	80	251

Table 2: Module 9 result of A* and RRT

		Iteration	Process Time (s)	Distance (ft)	Movements				Cycle time (min)
					S	L	H	Total	
	A*	-	8552	185.54	7	3	1	3	6.91
RRT	Distance	10	41.14	251.18	13.5	7.4	13.5	34.4	27.30
	Angle	10	67.59	266.65	13.5	9.8	14.8	38.1	30.39

In A*, a total of 7 crane operation steps are required to move module 9 from the pick point to the set point. The processing time to find the path was 2.38 hours, the total travelling distance of the module was 185.54

ft, and the total crane operation time (cycle time) was 6.91 minutes. There are two possible ways to proceed the RRT in terms of expanding the tree; one is based on the distance to the set point coordinates, and the other one is based on the angular difference to the set point movement. Each approach was iterated 10 times to determine the method that drew a better result. As Table 2 shows, expanding the tree based on the distance is better in all aspects. At the average result of RRT, the process time was 41 seconds, total travelling distance was 251.18 ft, total number of movements was 34, and total operation time was 27.3 minutes.

Table 3: Module 9 result of GA

Generation	Log	Process Time (s)	Distance (ft)	Movements	Cycle time (min)	Fitness	Success rate (%)
10	1.0	6.54	200.59	23.8	18.629	0.044	0
20	1.3	16.19	213.85	21.9	18.057	200.049	20
40	1.6	46.80	244.52	25.7	20.812	200.067	20
100	2.0	133.38	231.00	19.4	16.637	600.097	60
200	2.3	230.77	221.91	13.8	12.169	900.099	90
500	2.7	553.52	202.79	10.6	9.959	1000.115	100
1000	3.0	1456.07	201.70	8.9	8.547	1000.140	100
2000	3.3	2249.20	205.00	6.7	6.573	1000.180	100
4000	3.6	6142.66	203.17	6.3	6.394	1000.188	100
8000	3.9	9590.26	199.07	5.7	5.949	1000.216	100

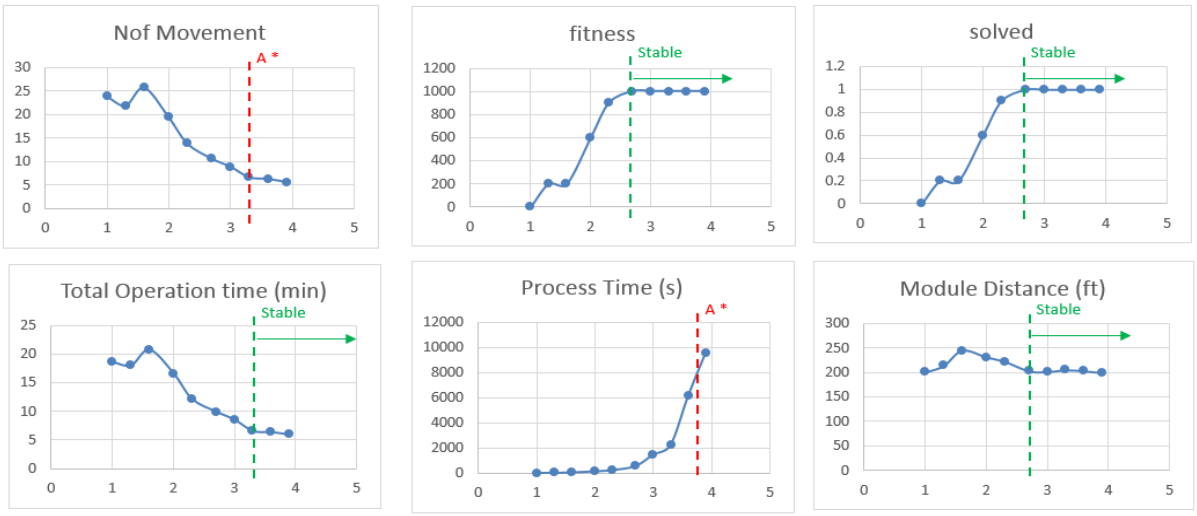
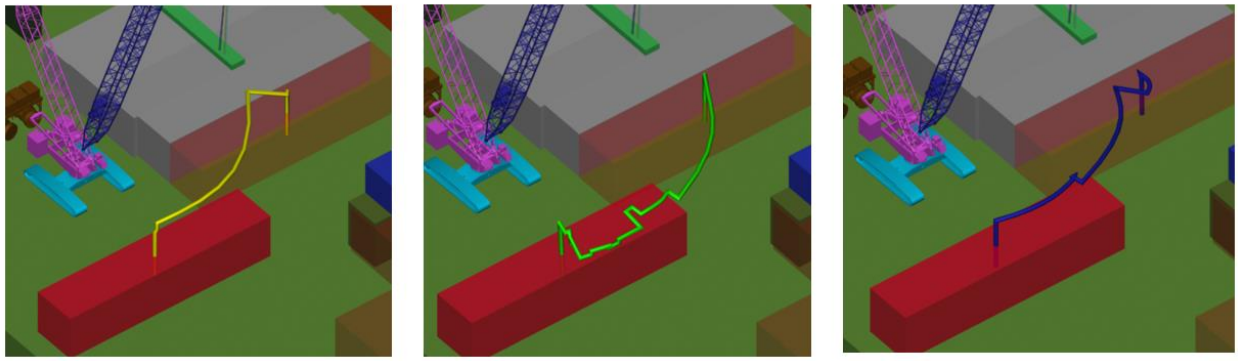


Figure 7: Module 9 GA result by generation

GA was tested with generations of 10, 20, 40, 100, 200, 500, 1000, 2000, 4000, and 8000 to observe the result tendency of increasing the generation number. Each generation was iterated 10 times to monitor the reliable outcomes. Table 3 and Figure 6 show the improved result by generations. For example, with 100 generations, the average process time was 133.38 seconds, the average module distance was 231 ft, the average number of movements was 19.4, and the average operation time was 16.6 min. Also, among 10 iterations, the success rate to find the solution was 60%.

In Figure 7, the X axis represents a log value of generations to display the result more effectively. The number of total movements, operation time, and module distance tend to decrease when increasing generations. In GA with 2000 generations, the total movements were 7, same as the A* result. The process time was 2249 seconds, 3.8 times faster than A* with 8551 seconds. As it is observed in Figure 6, the results at generation 500 tend to be stable and the success rate of 100%. Therefore, it is reasonable to use the result with 500 generations to compare GA with the other algorithms.



(a) A * search (b) RRT (c) GA

Figure 8: Path results in 3Ds Max

Figure 8 shows the visualized images of the path planning results for module 9 with the three algorithms in 3Ds Max. All the three algorithms succeeded in finding the solution path with different movement configurations. A*'s process time is the longest but results in the least amount of movements and the least operation time (cycle time). RRT only took 41 seconds to run, but the movements and cycle time were 5 times longer than A*. GA with 500 generations took 553 seconds while resulting similar quality solution as A*. Also, by increasing the generations, GA has a potential to generate a more optimal solution than A* while taking less process time.

4 CONCLUSION AND FUTURE WORK

Lift path planning is a critical process in the modular-based heavy industrial projects in terms of achieving the safety and productivity by reducing potential errors. In this respect, this paper has developed three path planning algorithms in mobile crane path optimization problem to execute the comprehensive comparison using Python. It is observed that A* search found the near optimal solution with comparatively higher computational time while RRT resulted in low quality outcome with the shortest computational time. GA resulted the most reasonable results with a flexibility to apply multiple constraints; however, applying constraints into the fitness equation is difficult with the increased complexity. Since this paper examined one case sample with low difficulty, more case studies with higher difficulty are required. In future research, more complicated DOF and constraints of the crane operations can be considered to reflect realistic environments. Furthermore, combined hybrid algorithm with the advantages of three algorithms can be suggested in the future works for the mobile crane path planning.

Acknowledgments

The authors would like to thank the case study data from PCL Industrial Management Inc. located in Edmonton, Canada.

References

Ali, M. S. Ajmal Deen, N. Ramesh Babu, and Koshy Varghese. 2005. "Collision Free Path Planning of Cooperative Crane Manipulators Using Genetic Algorithm." *Journal of Computing in Civil Engineering* 19 (2): 182–93.

Cai, Panpan, Yiyu Cai, Indhumathi Chandrasekaran, and Jianmin Zheng. 2016. “Parallel Genetic Algorithm Based Automatic Path Planning for Crane Lifting in Complex Environments.” *Automation in Construction* 62 (February): 133–47.

Chang, Yu-Cheng, Wei-Han Hung, and Shih-Chung Kang. 2012. “A Fast Path Planning Method for Single and Dual Crane Erections.” *Automation in Construction* 22 (March): 468–80.

Choset, Howie. “Robotic Motion Planning: RRT’s” https://www.cs.cmu.edu. Accessed January 29, 2019.

Han, Sang Hyeok, Shafiul Hasan, Ahmed Bouferguène, Mohamed Al-Hussein, and Joe Kosa. 2015. “Utilization of 3D Visualization of Mobile Crane Operations for Modular Construction On-Site Assembly.” *Journal of Management in Engineering* 31 (5): 04014080.

Han, SangHyeok, Ahmed Bouferguene, Mohamed Al-Hussein, and Ulrich (Rick) Hermann. 2017. “3D-Based Crane Evaluation System for Mobile Crane Operation Selection on Modular-Based Heavy Construction Sites.” *Journal of Construction Engineering and Management* 143 (9): 04017060.

Han, SangHyeok, Zhen Lei, Ahmed Bouferguene, Mohamed Al-Hussein, and Ulrich (Rick) Hermann. 2016. “3D Visualization-Based Motion Planning of Mobile Crane Operations in Heavy Industrial Projects.” *Journal of Computing in Civil Engineering* 30 (1): 04014127.

Hornaday W. C., Haas C. T., O’Connor J. T., and Wen J. 1993. “Computer-Aided Planning for Heavy Lifts.” *Journal of Construction Engineering and Management* 119 (3): 498–515.

Lei, Zhen, SangHyeok Han, Ahmed Bouferguène, Hosein Taghaddos, Ulrich Hermann, and Mohamed Al-Hussein. 2015. “Algorithm for Mobile Crane Walking Path Planning in Congested Industrial Plants.” *Journal of Construction Engineering and Management* 141 (2): 05014016.

Lin Kuo-Liang, and Haas Carl T. 1996. “An Interactive Planning Environment for Critical Operations.” *Journal of Construction Engineering and Management* 122 (3): 212–22.

Lin, Yuanshan, Di Wu, Xin Wang, Xiukun Wang, and Shunde Gao. 2014. “Lift Path Planning for a Nonholonomic Crawler Crane.” *Automation in Construction* 44 (August): 12–24.

Olearczyk, Jacek, Mohamed Al-Hussein, and Ahmed Bouferguène. 2014. “Evolution of the Crane Selection and On-Site Utilization Process for Modular Construction Multilifts.” *Automation in Construction* 43 (July): 59–72.

“eLCOSH: Understanding Crane Accident Failures: A report on the causes of death in crane-related accidents” 2010. http://www.elcosh.org/ Accessed January 24, 2019.

Sivakumar, Pl., Koshy Varghese, and N. Ramesh Babu. 2003. “Automated Path Planning of Cooperative Crane Lifts Using Heuristic Search.” *Journal of Computing in Civil Engineering* 17 (3): 197.

Srinivas, M., and L.M. Patnaik. 1994. “Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms.” *IEEE Transactions on Systems, Man, and Cybernetics* 24 (4): 656–67.