Advanced Engineering Informatics xxx (2012) xxx–xxx

Contents lists available at SciVerse ScienceDirect

Advanced Engineering Informatics

journal homepage: www.elsevier.com/locate/aei

Improving lifting motion planning and re-planning of cranes with consideration for safety and efficiency

Cheng Zhang *, Amin Hammad [1]

Concordia Institute for Information Systems Engineering, Concordia University, 1515 Ste-Catherine Street West, EV7.640, Montreal, Quebec, Canada H3G 2W1

ARTICLE INFO

Article history:
Received 9 February 2011
Received in revised form 19 December 2011
Accepted 9 January 2012
Available online xxxx

Keywords:
Motion planning
Re-planning
Cranes
Safety and efficiency

ABSTRACT

Safe and efficient operation of cranes requires not only good planning, but also sufficient and appropriate support in real time. Due to the dynamic nature of construction sites, unexpected changes in site layout may create new obstacles for the crane that can result in collisions and accidents. Previous research on construction equipment motion planning focuses on off-line support, which considers static environment or predictable obstacles. These plans may not fit the reality when the environment has any change. In this case on-site safety and efficiency can be affected. In this research, a motion planning algorithm is proposed to efficiently generate safe and smooth paths for crane motions while taking into account engineering constraints and the path quality. Path smoothness is taken into account to provide a realistic path for cranes and to reduce unnecessary movements. A dynamic motion planning algorithm is proposed to ensure safety during the execution stage by quickly re-planning and avoiding collisions. In addition, an anytime algorithm is proposed to search for better solutions during a given time period by improving path smoothness and by reducing path execution time. The proposed algorithms are compared with other available algorithms to evaluate their performance in terms of planning and re-planning time and the cost of the path. Based on the literature review, this is the first time that dual-tree RRT algorithms have been applied to crane motion planning.

1. Introduction

Previous research has indicated that machinery-related incidents were the fourth leading cause of traumatic occupational fatalities in the construction industry between 1980 and 1992, resulting in 1901 deaths (2.13 deaths per 100,000 workers) [27]. The same research has indicated that the construction equipment most frequently associated with fatalities are cranes (17%), excavators (15%), tractors (15%), loaders (9%), and pavers (7%). To improve safety on construction sites, simulation methods are used to help in selecting cranes [9], in simulating tasks [15], and in checking spatial constraints at the level of work spaces [40]. Many items need to be considered in crane selection and path planning, such as the capacity of the crane, the crane moving path, and the vision coverage of the crane operator. Among these items, a detailed lift planning is essential for critical crane lift processes. In current practice, planners have to manually check the working range of the selected crane to guarantee that it can reach the initial and goal locations of the lift object. The research in motion planning has a long history in robotics [8]. Motion planning aims to find a collision-free path for the robot system from one configuration to another. In the construction domain, path planning is a major concern and has been investigated to ensure safety and improve productivity [20,33].

Construction equipment can be treated as robots, and the same motion planning algorithms used in robotics can be applied; however, appropriate domain heuristics should be added to find a good/optimal plan within a reasonable period of time. Although some research has been focused on path planning for cranes [17,28,35], little discussion has been done about efficiently generating safe and smooth paths for crane motions while taking into account engineering and time constraints, and path quality. Furthermore, during the plan execution stage, once the environment has changed, fixing on-site problems can delay the schedule or increase the cost. These problems can be significantly reduced if efficient motion re-planning is applied based on the updated environment information.

The present paper is part of a research for developing a multi-agent system, which aims to improve construction safety and efficiency by providing intelligent assistance, such as giving warning alerts to operators and workers, and re-planning the path of a crane when a potential collision is detected [37]. The system integrates Real-Time Location Systems (RTLSs), path planning and

* Corresponding author. Tel.: +1 514 848 2424x7074; fax: +1 514 848 3171.
E-mail addresses: zha_che@encs.concordia.ca (C. Zhang), hammad@ciise.concordia.ca (A. Hammad).
[1] Tel.: +1 514 848 2424x5800; fax: +1 514 848 3171.

1474-0346/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.aei.2012.01.003

re-planning algorithms, and multi-agent communication and negotiation. The present paper focuses on investigating a new method for crane motion planning and re-planning by adapting robotic motion planning algorithms, taking into consideration the engineering constraints of cranes. The objectives of this paper are: (1) proposing a motion planning algorithm for hydraulic cranes taking into account construction equipment specifications; (2) investigating the path quality improvement of the proposed motion planning algorithm by evaluating the smoothness of the path and execution time; and (3) proposing a dynamic motion re-planning algorithm to provide near real-time support.

The proposed automation in motion planning and re-planning for cranes results in improving the safety of crane operations by avoiding collisions, and in improving the efficiency by eliminating the time of manual path planning and re-planning, and by providing a shorter motion path.

The present paper is organized as follows: Section 2 discusses related research in motion planning applications for construction equipment. Section 3 investigates the criteria for selecting motion planning algorithms taking into account engineering requirements. Based on these criteria, Rapidly-exploring Random Trees (RRTs) algorithms are selected as the basic algorithms used in the present research and several variations of RRTs are introduced. In Section 4, the methodology of applying motion planning algorithms to cranes is described, which is followed by two case studies in Section 5. Section 6 has the conclusions and future work.

2. Related research

In the construction domain, different algorithms have been used or evaluated for construction equipment path planning. For example, Soltani et al. [30] have evaluated the performance of three algorithms (Dijkstra, A* and Genetic Algorithms (GA)) for path planning in construction sites. Paths of site operatives and vehicles are evaluated based on three criteria: short distance, low safety risks and high visibility. Their experimental results have shown that Dijkstra and A* can find optimal solutions but suffer from the dimensional constraints; therefore, these two algorithms are inefficient in solving large scale problem. The GA algorithm takes relatively less time in finding a near-optimum solution; however, the limitations are less accurate solutions and a time-consuming fine-tuning process. Furthermore, the above mentioned research is based on a 2D discrete grid representation of the site, and only offline planning is investigated.

Tserng et al. [33] have proposed a methodology and several algorithms for interactive motion planning that are developed for multi-equipment landfill operations in an Automated Landfill System (ALS). This methodology simulates the operational processes of landfill vehicles and equipment in planning a landfill project. However, this system depends on pre-defined patterns to do motion planning for the equipment, which prevents the system from solving actual cases where there could be equipment on site that does not follow any of the specified moving patterns.

Kim et al. [20] have introduced a path-planning method for a mobile construction robot to find a continuous collision-free path from the initial position of the construction robot to its goal position. This work presents an improved Bug-based algorithm, called SensBug, which can produce an effective and short path in an unknown environment with both stationary and moving obstacles. However, their method did not improve the safety in all variations of bug algorithms where generated paths touch the obstacles.

Sivakumar et al. [29] have tried different algorithms, such as A* and GAs to optimize the collision-free path for cooperative lifting with two cranes. In the research of Ali et al. [2], a GA is used and compared with the A* algorithm, and the former is considered a better solution for two cranes working together. However, the authors have assumed that the site contains only static obstacles, and the proposed solutions provide only off-line planning, rather than real-time control of the movement. In addition, a fitness evaluation is applied by checking the coordination of two cranes, where the constraints from the lifting object are taken into consideration.

Kang and Miranda [16] have proposed an incremental decoupled method to plan motions for multiple cranes so that collisions among any of the cranes are avoided as are possible collisions between the cranes and the transported objects. Three different algorithms were integrated to find a path efficiently [16–18]. These three algorithms are conceptually similar to the Rapidly-exploring Random Trees (RRT) algorithm but are simplified for the specific case of a tower crane. This approach is efficient but it is incomplete, i.e., it may fail to find a solution even if there is one. Although this research considered dynamic changes on site to make the path more realistic, it was assumed that the environment information was known by exactly following the work schedule. The real situation on site is that unknown objects should be monitored using sensors and taken into account to ensure the collision-free movement of equipment [39].

In the case of motion planning for equipment on construction sites, the model-based approach should be used during the planning stage. In this approach, a 3D model of the site is available and full information about the geometry of the equipment and the obstacles is given beforehand, so path planning becomes a one-time off-line operation. During the execution stage, the dynamic environment needs sensor-based planning with the assumption that some obstacles are unknown. This lack of information is compensated for by local on-line (real-time) information coming from sensory feedback [31]. The difference between motion planning for equipment on construction sites and the robotic exploration in an unknown environment is that every task carried out on a construction site has a schedule; therefore, the unknown information can be assumed to be minor or less essential to the whole plan most of the time, and a motion re-planning approach can efficiently modify the off-line plan based on real-time sensed data.

3. Motion planning algorithm selection

Many algorithms are available for generating collision-free paths in the Configuration Space (*C-space*) [25]. *C-space* is the set of all possible configurations of a robot. A configuration is simply a point in this abstract *C-space*. The configuration of a robot system is the complete specification of the position of every point in that system. Once the motion planning problem has been formulated in the *C-space*, it becomes equivalent to finding the connected sequence of collision-free configurations running from the initial configuration to the goal configuration. Based on the data structure representation of the *C-space*, motion planning algorithms can be categorized under two major approaches [8]:

(1) *Motion planning in discrete space:* In this case, the *C-space* is defined as a state-space model with a countable finite set of states. The planning algorithms build roadmaps in the free (or semi-free) state-space and search for the feasible path. Each of these algorithms relies on an explicit representation of the geometry of the free space. Because of this, as the dimension of the *C-space* grows, these algorithms become impractical. Grid A* and Visibility Graph are representative algorithms of discrete space planning.
(2) *Motion planning in continuous space:* In this case, the algorithm is not limited to a pre-defined finite search space representation of the *C-space*. Instead, a variety of strategies are utilized for generating samples (collision-free

Please cite this article in press as: C. Zhang, A. Hammad, Improving lifting motion planning and re-planning of cranes with consideration for safety and efficiency, Adv. Eng. Informat. (2012), doi:10.1016/j.aei.2012.01.003

configurations) and for connecting the samples with paths to obtain solutions to path-planning problems in a continuous *C-space*. Sampling-based algorithms are capable of dealing with robots with many degrees of freedom (DoFs) and with many different constraints. Such algorithms do not attempt to explicitly construct the boundaries of the *C-space* obstacles or to represent cells of the free space. Instead, they rely on a procedure that can decide whether a given configuration of the robot is in collision with the obstacle or not. Efficient collision detection procedures ease the implementation of sampling-based algorithms and increase the range of their applicability. PRM (Probabilistic Roadmap Planner) and RRT (Rapidly-exploring Random Trees) are representative algorithms of continuous space planning.

Comparisons have been done among the Grid A*, Visibility Graph, PRM, and RRT [7] in terms of completeness, optimality, efficient environment updates, efficient query updates, good DoF scalability, and ability of solving non-holonomic configurations. It has been indicated that PRM and RRT are efficient in query updates, and have good DoF scalability. However RRT is not able to guarantee the generation of an optimal path based on pre-defined criteria; therefore an optimization update is required to address this point. Fortunately, for many of these algorithms, the solutions produced are not too far from optimal in practice [23]. Among the algorithms reviewed in that research, RRT is the best in terms of efficient environment update [7]. Furthermore, RRT stands out because of its high ability in solving non-holonomic configurations.

In addition to the above-mentioned algorithms, Garber and Lin [13] have proposed a constraint-based motion planning approach for virtual prototyping. This approach transforms the motion planning problem into the simulation of a dynamic system in which the motion of each rigid robot is subject to the influence of virtual forces induced by geometric constraints. However, only geometric constraints are considered in this research.

Several criteria are taken into consideration in the selection of the motion planning algorithm for cranes in the present paper. There are four major criteria that are taken into account in the present research: efficiency, optimality, reusability, and safety.

(1) *Efficiency*: Efficiency is the most important factor because decisions usually need to be taken in near real-time to cope with the dynamic nature of the environment. RRTs have been shown to be effective for solving single-shot path planning problems in complex *C-spaces* by combining random sampling of the *C-space* with biased sampling around the goal configuration [6,7]. RRTs efficiently provide solutions to problems involving vast, high-dimensional *C-space*. These solutions would be intractable using deterministic approaches.
(2) *Optimality*: Optimality is considered as the ability to find an optimal path with respect to some metrics. Single-query sampling-based algorithms are not able to guarantee the generation of an optimal path based on pre-defined criteria. Optimization updates are required to address this point [23]. The basic RRT algorithm does not take path quality into account during its search, which may produce paths that are grossly suboptimal [10]. To improve the quality of the solution path, Urmson and Simmons [34] have proposed modified RRT algorithms that take the cost of the path into account. Berg et al. [4] have considered adding the cost of the path in navigating a mobile robot.
(3) *Reusability*: This requirement is specific to motion re-planning when a new obstacle appears. An efficient re-planning algorithm should be able to plan optimal traverses in near real time by incrementally repairing the paths of the equipment as new information is discovered. Re-planning algorithms should focus on the repairs to significantly reduce the total time required for the initial path calculation and subsequent re-planning operations [32]. Deterministic re-planning algorithms such as D* efficiently repair previous planning solutions when changes occur in the environment [8]. They do this by determining which parts of the solution are still valid and which parts need to be recomputed. However, as the number of the dimensions of the search space increases, for example, in the case of multiple cranes working together, deterministic algorithms simply cannot cope with the size of the corresponding state space. On the other hand, RRT algorithms abandon the original tree and grow a new RRT path from scratch if the environment is dynamic. This can be a very time-consuming operation, particularly if the planning problem is complex. In such a case, researchers have started implementing the Dynamic RRT (DRRT) algorithm as a probabilistic analog to D* for navigation in unknown or dynamic environments [10]. DRRT aims at repairing the current RRT when new information concerning the *C-space* is received instead of abandoning the current RRT entirely. DRRT removes the invalid part of the path and grows the remaining tree until a new solution is found [11].
(4) *Safety*: To ensure the generation of safe paths for cranes, the algorithm needs to consider the engineering constraints of the crane. Taking the motion planning of hydraulic cranes as an example, the working range, the load charts, and the rules of action should be followed to ensure on-site safety.

RRTs have been shown to satisfy most of those criteria. One of the advantages of using an RRT algorithm is that it does not need an explicit representation of the *C-space* [8]. It is enough to define the range of each DoF of the equipment based on the engineering constraints, for example, the load charts and working ranges of a crane. The engineering constraints of a crane further narrow down the *C-space* into a feasible space, thereby fulfilling the feasibility of the movement according to the load charts and the working ranges of the crane. Therefore, the feasible *C-space* can be defined according to the crane-specific conditions, e.g., the lift weight, the counterweight, and the outrigger radius. In this way, safe and realistic motion plans for cranes are generated by taking into account the engineering constraints.

3.1. RRT algorithms

RRT algorithms have been selected as the basic algorithms for motion planning in the present research. Several modifications have been made to the algorithm to improve their performance and to take into consideration the specific requirements of cranes, as explained in Section 4.1.4. RRT algorithms incrementally construct a search tree rooted either at an initial configuration q_{init} or a goal configuration q_{goal}. At each iteration from 1 to m, a random configuration, q_{rand}, is sampled uniformly in the search space. The nearest configuration, q_{near}, to q_{rand} in the tree is found and an attempt is made to extend the tree, and finally connect q_{init} and q_{goal}. This method was originally developed by LaValle [22]. In the basic RRT algorithm, the *Extend* function selects the node q_{near} in the tree that is nearest to the sampled node q_{rand}, as shown in Fig. 1. Then a motion toward q_{rand} with some fixed incremental distance ε is applied. If this motion is collision-free, a new node q_{new} is added to the tree and the *Extend* function returns one of the following two values: *Reached* if q_{rand} is reached or *Advanced* if q_{rand} is not reached. If q_{new} is not collision-free, a *Trapped* value is returned. The pseudo code of the *Extend* function is given in Fig. 5.

Please cite this article in press as: C. Zhang, A. Hammad, Improving lifting motion planning and re-planning of cranes with consideration for safety and efficiency, Adv. Eng. Informat. (2012), doi:10.1016/j.aei.2012.01.003

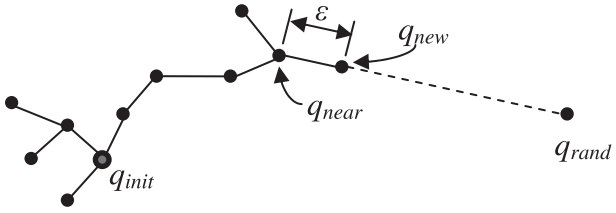


Fig. 1. The Extend operation [24].

3.2. RRT-connect-connect algorithm

Kuffner and LaValle [21] have proposed the RRT-Connect-Connect (RRT-Con-Con) algorithm, in which two trees rooted at the initial configuration and the goal configuration are built. The procedure of developing two trees consists of the following steps: (1) growing tree T_{init} (rooted in the initial state) toward a random sample node q_{rand}, resulting in a new node q_{new}; (2) growing tree T_{goal} (rooted in the goal state) toward the new node q_{new} in T_{init}; (3) growing T_{goal} toward another random sample node q'_{rand}, resulting in a new node q'_{new}; (4) growing tree T_{init} toward the new node q'_{new} in T_{goal}. This method is somewhat biasing the direction of the tree generation. It is expected that growing the two trees towards each other is a faster way of finding a solution. Furthermore, building two trees from the initial and goal configurations aims to precisely select the goal state instead of assuming that the goal state will be reached by generating nodes randomly, especially when a manipulator must reach a specific end-effector position and orientation. In RRT-Con-Con, instead of attempting to extend the tree by an ε step, the *Connect* heuristic function iterates the *Extend* step until q_{rand} is reached or an obstacle is detected. In each iteration, one tree is extended, and an attempt is made to connect the nearest node of the other tree to the new node, and then vice versa. This greedy heuristic results in reducing calculation time by a factor of three or four, especially in an uncluttered environment [21]. However, this greediness may result in an increased randomness of the whole path and makes the movement unrealistic.

3.3. Path quality improvement

As mentioned in the *Optimality* criterion, the basic RRT algorithm does not take the cost of the path into account during its search. It almost always converges to a suboptimal solution [19]. It is often the case that a feasible path is found quickly; therefore, to improve the quality of the path, additional computational time can be devoted to improving the solution with heuristics depending on how much time is allowed in the application, which is called anytime algorithm. Examples can be found in Zilberstein and Russell [41] and Berg et al. [4]. Berg et al. [4] have considered adding the cost of the path in navigating a mobile robot. Time taken to execute the path (t_{path}) and the cost of the path (c_{path}), based on all relevant metrics other than time, are considered to contribute to the overall cost of the path. Examples of the metrics included in c_{path} are the proximity to adversaries or friendly agents and communication access. Eq. (1) shows the cost function of the path.

$$C_{path} = w_t \cdot t_{path} + w_c \cdot c_{path} \tag{1}$$

where w_t and w_c are weights which add up to 1. It should be noted that the term *cost function* is equivalent to *objective function* used in optimization. Although the quality of the path is improved using the cost function, the time needed for finding a solution increases significantlywhen the costs of different paths are calculated and compared [10]. In recent research, Karaman and Frazzoli [19] have claimed that the algorithm they proposed, Rapidly-exploring Random Graph (RRG), is asymptotically optimal in the sense that it converges to an optimal solution almost surely as the number of samples approaches infinity. However, the computing time of the algorithm can be up to five times that of the RRT algorithm. Thus, that research is impractical and not applicable for solving problems involving the motion planning of cranes in near real time. Consequently, in the present paper, the advantage of the greediest dual-tree algorithm RRT-Con-Conis taken and a cost function is used to improve the path quality.

3.4. Dynamic RRT algorithm

During re-planning, the crane must wait for the new path to be computed; therefore, rapid re-planning is essential. As described in the *Reusability* criterion, Ferguson et al. [11] have proposed the DRRT algorithm, which aims at repairing the current RRT when new information is received. It removes just the invalid part of the path and grows the remaining tree until a new solution is found. The steps of DRRT are the following: (a) an initial RRT is generated from an initial position to a goal position, (b) a new obstacle is detected in the *C-space*, (c) the parts of the previous tree that are invalidated by the new obstacle are marked, (d) the tree is trimmed and invalid parts are removed, (e) the trimmed tree is grown until a new solution is generated.

However, this DRRT is specific to one tree structure, which has one root and tries to reach the goal by randomly sampling the nodes. During the execution of the path, the mechanism of generating the tree is modified by reversing the direction of the tree when an obstacle is detected so that the new direction is from the goal configuration to the current configuration of the robot. The purpose is to avoid changing the root frequently when a new partial tree has to be generated to avoid collisions. Once an obstacle is detected, the whole path is checked to remove the nodes that may collide with the obstacle. However, in a dual-tree structure, two trees are generated simultaneously from the root and the goal. The mechanism of connecting two trees is missing in the DRRT algorithm which is designed for a single tree. Furthermore, it is time consuming to trim the entire path for all new obstacles, which may affect the far later nodes on the path. Some of the detected obstacles can become non-obstacles at a later time because they continuously move in the space. Therefore, in the proposed algorithm for re-planning, only the next several nodes on the path are checked for collision to ensure immediate safety.

4. Methodology

Based on the discussion in Section 2, RRT algorithms have been selected as the basic algorithms used in this research for crane motion planning and dynamic re-planning in near real time. During the execution of the plan of a crane, dynamic obstacles are detected using sensors and potential collisions, if any, are detected. In that case, path re-planning is triggered to avoid the newly found obstacles. Safety is assured by adding buffers to the components of the cranes, for example the boom. Collision detection is done based on the buffered crane boom to ensure there is a safe distance between the boom and the obstacle. The size of the buffer depends on the speed of the crane, and the accuracy of the location sensors. However, calculating the size of the buffer is not the focus of our present paper. In the following subsections, the improvement of planning time and path quality is presented.

4.1. Proposed motion planning algorithm

4.1.1. Path smoothness

Due to the random nature of sampling-based algorithms and the greediness of the *Connect* function of RRT-Con-Con, the

Please cite this article in press as: C. Zhang, A. Hammad, Improving lifting motion planning and re-planning of cranes with consideration for safety and efficiency, Adv. Eng. Informat. (2012), doi:10.1016/j.aei.2012.01.003

jaggedness of the path is inevitable and some movements of the crane may be unnecessary. These problems make the RRT-Con-Con algorithm impractical for crane operation. Therefore, in this research, the smoothness of the path is used as a metric to evaluate the path. Smoothness can be improved by keeping the difference between the coordinates of two subsequent nodes at a minimum or relatively small. Ali et al. [2] have introduced a fitness function in a Genetic Algorithm for crane path planning. The fitness function calculates the total angular displacement of the robotic joints while the crane moves from a pick-up location to a place location. In the present research, the smoothness of a path is calculated by introducing a function $r(q_i, q_{i+1})$ representing the roughness of the movement from node q_i to the next node q_{i+1} on the path, as shown in the following equation.

$$r(q_i, q_{i+1}) = \sqrt{\sum_{j=1}^{m} \left(d^j_{q_{i+1}} - d^2_{q_i}\right)^2} \tag{2}$$

where m is the number of DoFs of the equipment; $d^j_{q_i}$ and $d^j_{q_{i+1}}$ are the normalized values of the movement along the jth DoF of node sq_i and q_{i+1}, respectively. The purpose of normalization is to consider the effects of all movements of the boom represented by polar coordinates that have different units and different ranges. For example, in the case study discussed in Section 4, the swing angle of the boom of the hydraulic crane is in the range of $[-180, 180]$ degrees and the boom extension is in the range of $[10.97, 33.53]$ meters. However, after normalization, both movements are in the unitless range of $[0, 1]$. The roughness of the path r_{path} is the sum of the roughness of movements between all couples of consecutive nodes on the path, as shown in Eq. (3).

$$r_{path} = \sum_{i=1}^{k-1} r(q_i, q_{i+1}) \tag{3}$$

where k is the number of nodes on the path. r_{path} should be minimized to improve the smoothness of the crane movement.

4.1.2. Cost function

Considering plan execution time and path smoothness, the overall cost of a path can be calculated using a cost function, as shown in Eq. (4), which is used to evaluate each path and to select a better path. A path with less execution time and lower roughness value is considered to be a better path.

$$C_{path} = \alpha \cdot w_t \cdot t_{path} + \beta \cdot w_r \cdot r_{path} \tag{4}$$

t_{path} is calculated based on the path and velocity along each DoF. For example, the velocities of extending the boom and swinging the boom are different; therefore, the time of executing a path should be calculated according to the equipment specifications.

The weights of execution time w_t and path roughness w_r can be based on a preference of shorter time or better smoothness. For example, if better smoothness is preferable, $w_t = 0.4$ and $w_r = 0.6$ can be selected. Other weights (α and β) are used to adjust the values of t_{path} and r_{path} to the same order of magnitude. For example, if the range of the values of t_{path} is within 100 to 1000, and the range of the values of r_{path} is within 1 to 10, $\alpha = 0.01$ and $\beta = 1$can be applied to t_{path} and r_{path}, respectively.

4.1.3. Selective node sampling

After randomly sampling nodes over the whole *C-space*, only nodes that are expected to result in a smoother path are selected as nodes that will be connected to the tree. The smoothness of a newly sampled node q is evaluated by summing of the roughness values from this node to the roots (q_{init} and q_{goal}) of the two trees T_{init} and T_{goal}.

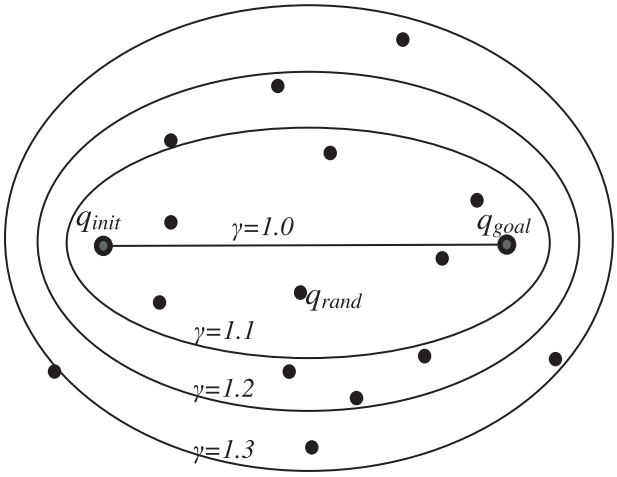


Fig. 2. Nodes with different roughness values.

$$R(q) = r(q, q_{init}) + r(q, q_{goal}) \tag{5}$$

A threshold is set by multiplying the roughness value between the initial and the goal nodes ($r(q_{init}, q_{goal})$) by an amplifier (γ). The first sampled node that meets the following inequality is selected as the node to be connected to the tree.

$$R(q) \leqslant \gamma \cdot r(q_{init}, q_{goal}) \tag{6}$$

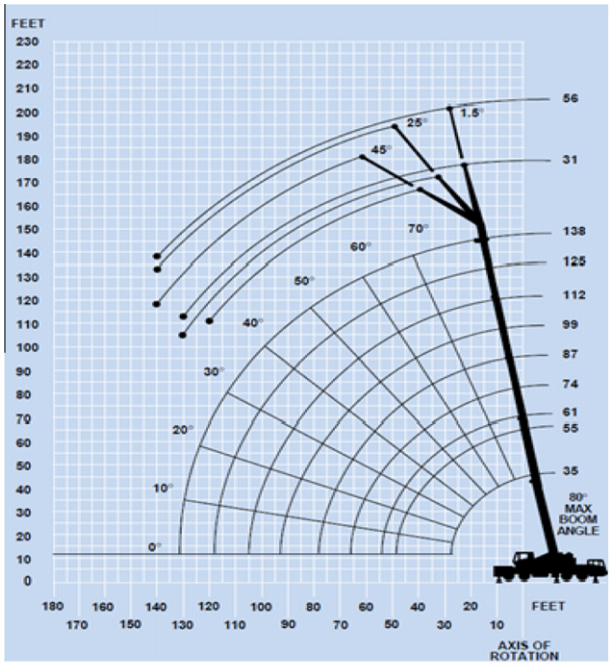
However, to restrain the calculation time, if the algorithm failed to find a node satisfying this condition after k trials, the node with the minimum roughness value is selected. Fig. 2 shows a 2D schematic example of nodes with different roughness values. Each γ value defines an area where the nodes inside a smaller area are expected to contribute to a better solution. According to this criterion, the smaller the γ value, the better the smoothness of the node.

4.1.4. Engineering constraints

For cranes, the engineering constraints are mainly created by the working range and by the load charts in order to avoid tip-over problems. Fig. 3 shows examples of a working range graph and of a load chart. The working range shows the minimum and maximum boom angles according to the length of the boom and to the size of the counterweight. Load charts give the lifting capacity based on the boom length, the boom angel to the ground and the size of the counterweight. For example, for a Grove crane TSM870 [12], if the lift object is 15,000 lbs (6.8 metric tons)and the counterweight is 18,000 lbs (8.2 metric tons), the ranges of the three DoFs for this lifting task are extracted from the load chart of the crane (Fig. 3) and shown in Table 1 as the following: (1) boom length: 36–110 ft (10.97–33.53 m); (2) luffing angle: 23–80 degrees; (3) swing angle: −180 to 180 degrees. Furthermore, the range of luffing angles varies according to the boom length; therefore, node sampling should be constrained within these ranges. When connecting the sampled node to the tree, the intermediate nodes are also checked to make sure that they meet these engineering constraints. The *Connect* and *Extend* functions discussed in Sections 3.1 and 3.2 are modified to check these constraints for the intermediate nodes on the connection edge between the sampled node and the tree.

When a random node is generated in the feasible *C-space*, the position of each component of the crane in the work space is defined, and then a collision detection algorithm is used to detect whether there is any collision between the crane and the obstacles on site. Current collision detection methods applied in robotics and computer graphics are generally more complex than necessary for construction purposes and are relatively difficult to implement

Please cite this article in press as: C. Zhang, A. Hammad, Improving lifting motion planning and re-planning of cranes with consideration for safety and efficiency, Adv. Eng. Informat. (2012), doi:10.1016/j.aei.2012.01.003



(a)

(Feet)	36	50	*60	70
10	+140,000 (68)	109,500 (75)	84,200 (78)	**56,450 (80)
12	110,500 (64)	104,500 (72.5)	79,850 (76)	56,450 (78.5)
15	96,800 (58.5)	91,400 (69)	73,900 (73)	56,450 (76)
20	78,750 (47)	75,300 (62)	59,600 (67.5)	56,450 (71.5)
25	59,800 (32.5)	59,750 (55)	50,000 (62.5)	48,900 (67)
30		47,300 (47)	42,300 (56.5)	41,900 (62.5)
35		38,550 (37.5)	36,950 (50)	36,400 (57.5)
40		28,450 (24.5)	28,450 (43)	29,700 (52)
45			23,400 (34.5)	24,650 (46.5)
50			19,450 (23)	20,700 (39.5)

(b)

Fig. 3. Examples of working range and load chart of a crane (Grove [12]). (a) Working range. (b) Load chart

Table 1
Ranges of crane boom length and luffing angles.

Boom length (ft)	Luffing angle limits (degree)
36	32.5–68
50	24.5–75
60	23–78
70	32–80
80	37.5–78.5
90	46.5–80
100	53–79
110	54.5–80

efficiently [16]. Therefore, bounding boxes of the components of the crane are considered enough for collision detection with objects in the environment. Buffers added to the bounding boxes can ensure the safety on site.

4.1.5. Details of the proposed motion planning algorithm

The proposed RRT-Con-Con-Mod is a modified version of RRT-Con-Con, which uses selective node sampling and considers the engineering constraints introduced above. Fig. 4 shows the flowchart of the RRT-Con-Con-Mod algorithm. Fig. 5 shows the pseudo code of the same algorithm. It should be noted that not all the details of the algorithm are shown in the flowchart for better readability. First of all, the load charts and the working range data are read after the parameters of the lift task are defined including the lift weight, the counterweight of the crane, and the initial and goal configurations of the crane. Based on the required capacity, the working range of the crane is defined. A node q_{rand} is sampled randomly within the range of the engineering constraints. Then the roughness value of q_{rand} is calculated based on Eq. (5). If this value is less than the threshold value $\gamma \cdot r\,(q_{init}, q_{goal})$, q_{rand} is selected as the node that one tree should grow towards. If the value is bigger than the threshold, this value is compared with the smallest roughness value of other sampled nodes. Then, after several trials (maximum k trials), the node with the smallest roughness value is returned and connected to the tree. To consider the trade-off of calculation time and path quality, k is set to 20. Next, the modified *Connect* function (*Connect-Mod*) is called to try to connect one tree with q_{rand}. The *Connect-Mod* function calls the *Extend-Mod* function repeatedly and gradually extends the tree towards q_{rand} under the condition that each extension is collision-free and the end node (q_{new}) satisfies the engineering constraints. If the tree reaches q_{rand}, which means T_a is extended to q_{new}, the *Connect-Mod* function is called again to try to connect the other tree (T_b) with q_{new}. If after several extensions of the tree, T_b reaches q_{new}, which means the two trees are connected, then the path between the initial and the goal states is returned. In the cases that the returned status of *Connect-Mod* is *Trapped* when connecting T_a or T_b with the node q_{new}, the two trees are swapped and a new round of sampling and connecting starts until the maximum number of node-sampling trials m is reached. The default value of m is 1000. When the number of sampled nodes reaches this value without finding any path, the algorithm will stop searching and return *Failed*. The value of m is related to the complexity of the environment (i.e. more complex environment may require larger m value).

Due to the randomness of the RRT-Con-Con-Mod algorithm, it is difficult to guarantee that better solutions can be found using a specific γ value with one trial using one specific random seed number. A large number of tests have been carried out by changing the random seed number and the γ value. It was found that the paths generated by using the same random seed number are the same when γ value is large, e.g., $\gamma = 2.0$ in the first case study introduced in Section 4. These paths are the same as the paths generated by the RRT-Con-Con algorithm. It was also found that when the γ value is gradually reduced, the roughness of the path in the majority of the cases decreases to some extent and then starts to increase again at a certain value of γ. This γ value varies depending on the value of the random seed number. This behavior occurs because small γ values can result in over-constraining the dual-tree generation mechanism. Consequently, in order to obtain the best feasible path, an anytime algorithm is applied to investigate the improvement of the proposed RRT-Con-Con-Mod

Please cite this article in press as: C. Zhang, A. Hammad, Improving lifting motion planning and re-planning of cranes with consideration for safety and efficiency, Adv. Eng. Informat. (2012), doi:10.1016/j.aei.2012.01.003

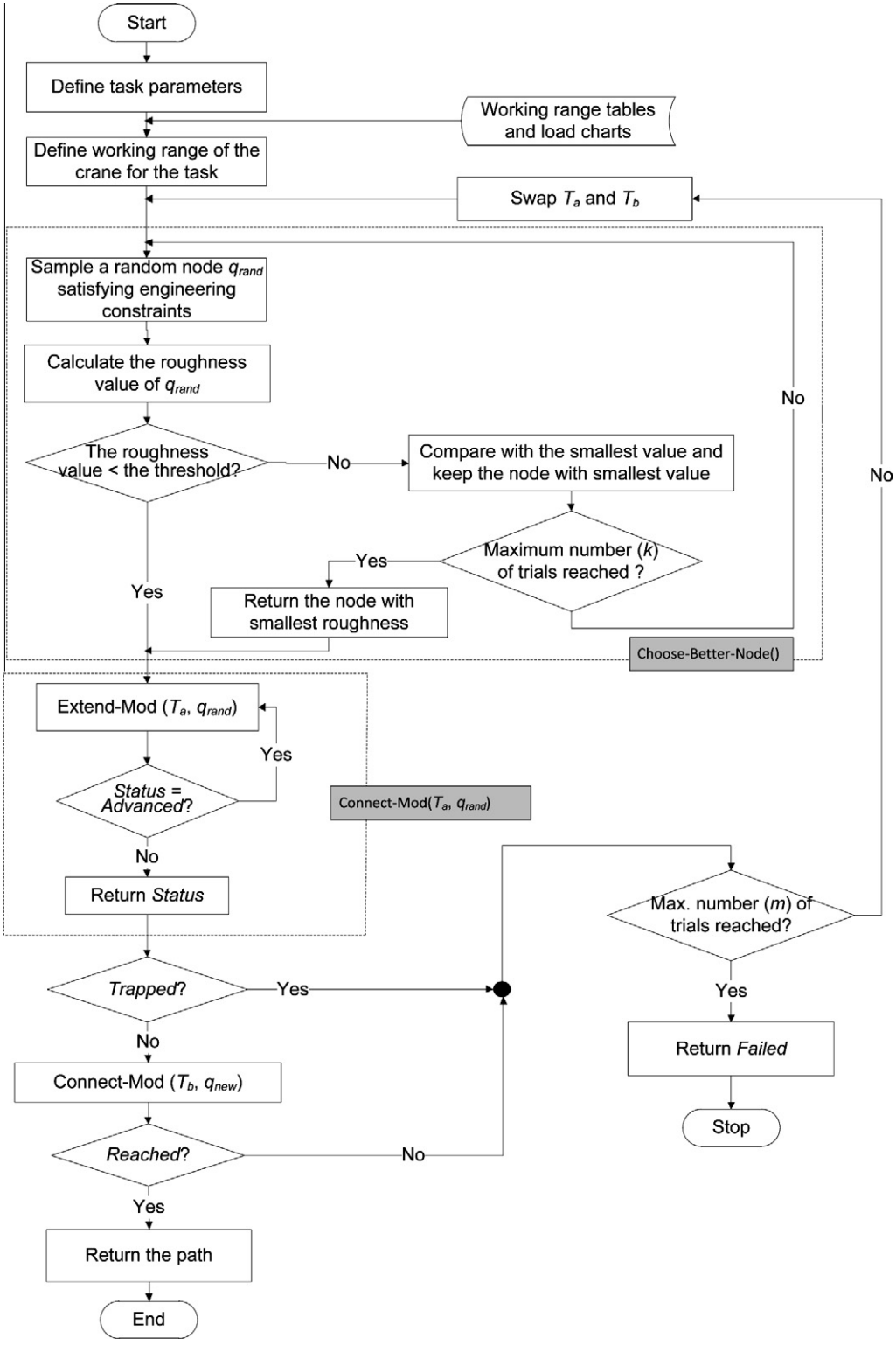


Fig. 4. Flowchart of motion planning algorithm RRT-Con-Con-Mod.

algorithm by changing the random seed number and the γ value. Fig. 6 shows the flowchart of this algorithm. Instead of reading the load charts and the working range data every time when calling RRT-Con-Con-Mod, they are read only once after defining the task parameters. Based on this information, the working range of the task is defined. A path P_{best} is initialized by calling RRT-Con-Con to find a feasible path. By reducing the γ value, nodes with lower roughness value are selected and connected to the tree; in this way, a better path is expected. After several loops of reducing γ, no more improvement can be realized with the same random seed number. If there is enough time to search for a better path, the random seed number is changed to find other feasible paths, and the loop of improving the smoothness of the path is repeated until time is over.

Please cite this article in press as: C. Zhang, A. Hammad, Improving lifting motion planning and re-planning of cranes with consideration for safety and efficiency, Adv. Eng. Informat. (2012), doi:10.1016/j.aei.2012.01.003

```
RRT-Con-Con-Mod(q_init, q_goal)
 1. T_a.add(q_init); T_b.add(q_goal);
 2. for i = 1 to m do
 3.       q_rand = Choose-Better-Node();
 4.       if not (Connect-Mod(T_a, q_rand) == Trapped)
 5.             if (Connect-Mod(T_b, q_new) == Reached)
 6.                   return Path(T_a, T_b);
 7.       Swap(T_a, T_b);
 8. return Failed;

Choose-Better-Node()
 9.  R_o = r(q_init, q_goal);
10.  R_less = ∞;
11.  q_better = null;
12.  for i = 1 to k
13.       q = Random-Node-Satisfy-Eng-Const();
14.       R = r(q, q_init ) + r(q, q_goal);
15.       if R ≤ γ · R_o
16.             return q;
17.       else
18.             if R ≤ R_less
19.                   q_better = q;
20.  return q_better;

Connect-Mod(T, q)
21.  repeat
22.       Status = Extend-Mod(T, q);
23.  until not (Status == Advanced)
24.  return Status;

Extend-Mod(T, q)
25.  q_near = Nearest-Neighbor(q, T);
26.  q_new = New-Node(T, q_near, q);
27.  if (Collision-Free(q_new) == true and Eng-Const-Satisfied(q_new) == true)
28.       T.add(q_new);
29.       if q_new ≈ q
30.             return Reached;
31.          else
32.             return Advanced;
33.  return Trapped;
```

Fig. 5. Pseudo code of motion planning algorithm RRT-Con-Con-Mod.

4.2. Proposed motion re-planning algorithm

Planning motions for navigating the crane on the actual construction site is more challenging than planning motions off-line. The main challenges come from incomplete or imperfect information, limited deliberation time, and the dynamic environment [10]. Models of the building under construction should be updated based on the project progress monitoring, which can be either retrieved from building information model (BIM) [14] or using 3D scanners [5]. However, the update rate needed for the construction processes is much less than the update rate of equipment path planning. Thus, updated 3D models of the building can be still considered as static obstacles by satisfying this assumption and by updating the building models between successful equipment tasks. Therefore, in the present research, during the plan execution stage, a Real-Time Location System (RTLS) composed of several Ultra Wideband (UWB) sensors is used to monitor the construction site, and multiple tags are attached to different components of objects, e.g., the boom and outriggers of a crane, as explained in Zhang [36]. Consequently, the poses of an obstacle can be calculated in near real-time and are used for collision detection. Details about the UWB system can be found in Zhang et al. [38]. At each configuration state on the path of the crane, the information about the dynamic objects is updated continuously, and collision detection is applied. If the next movement of the crane is not collision-free, the crane stops and re-planning is triggered to find a new collision-free path.

Please cite this article in press as: C. Zhang, A. Hammad, Improving lifting motion planning and re-planning of cranes with consideration for safety and efficiency, Adv. Eng. Informat. (2012), doi:10.1016/j.aei.2012.01.003

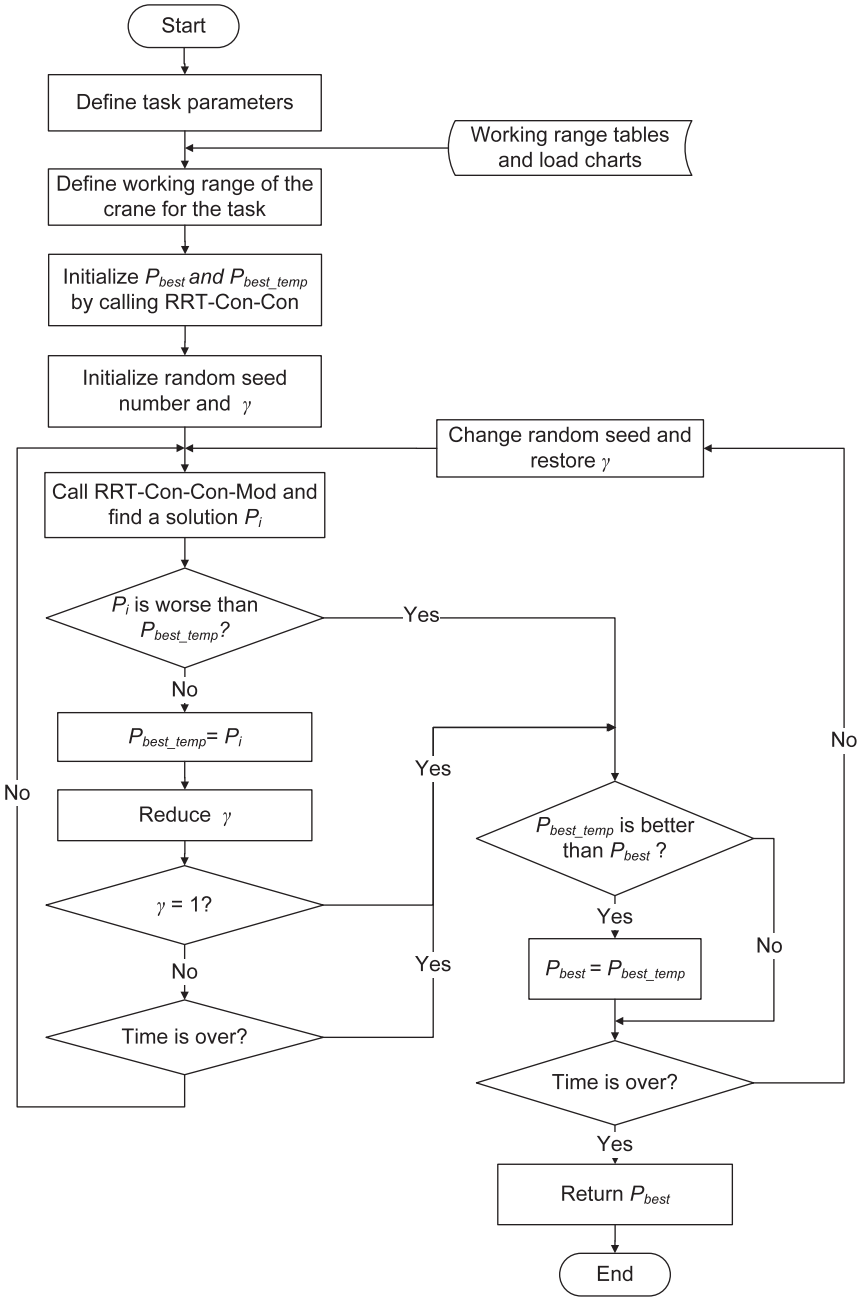


Fig. 6. Flowchart of anytime motion planning algorithm.

As discussed in Section 3.4, an efficient re-planning algorithm should be able to plan optimal traverses in near real time by incrementally repairing paths of the equipment as new information is discovered. Reusing a part of the previously generated plan is a good option to reduce the time of re-planning. Since RRT-Con-Con-Mod has been proposed in the present research for rapid motion planning for cranes, a dynamic version of this algorithm is proposed to solve the re-planning problem.

Fig. 7 shows the flowchart of the DRRT-Con-Con-Mod algorithm. Fig. 8 shows the pseudo code of the same algorithm. It should be noted that not all the details of the algorithm are shown in the flowchart for better readability. RRT-Con-Con-Mod is used twice: first to generate the initial plan (Lines 1–3 in Fig. 8) and, then, to repair the path if necessary (from Line 15). The path is repaired by re-growing two trees rooted at the current node q_{crane} (current configuration of the crane) and the first collision-free node $q_{newgoal}$ on the remaining path (Line 14) while detecting collisions for the remaining path based on updated environment information.

The main function starts by growing two trees from the initial q_{init} configuration and the goal q_{goal} configuration taking into account the engineering constraints and the roughness value. If the two trees connect successfully, a path P is obtained and is executed using a loop until the equipment reaches the goal (Lines 4–25). Before the next movement, environment information is updated (Line 8) to check whether the next movement is collision-free or not. If there is no obstacle, the crane is moved to the next configuration (Lines 10 and 11); otherwise, the remaining path $P_{remaining}$ is evaluated to remove the affected nodes on it (Line 13). Once a collision-free node is found, this node is recorded as a temporary goal node $q_{newgoal}$, and together with the crane's current configuration q_{crane} as an initial node, two new trees are grown to connect these two

Please cite this article in press as: C. Zhang, A. Hammad, Improving lifting motion planning and re-planning of cranes with consideration for safety and efficiency, Adv. Eng. Informat. (2012), doi:10.1016/j.aei.2012.01.003

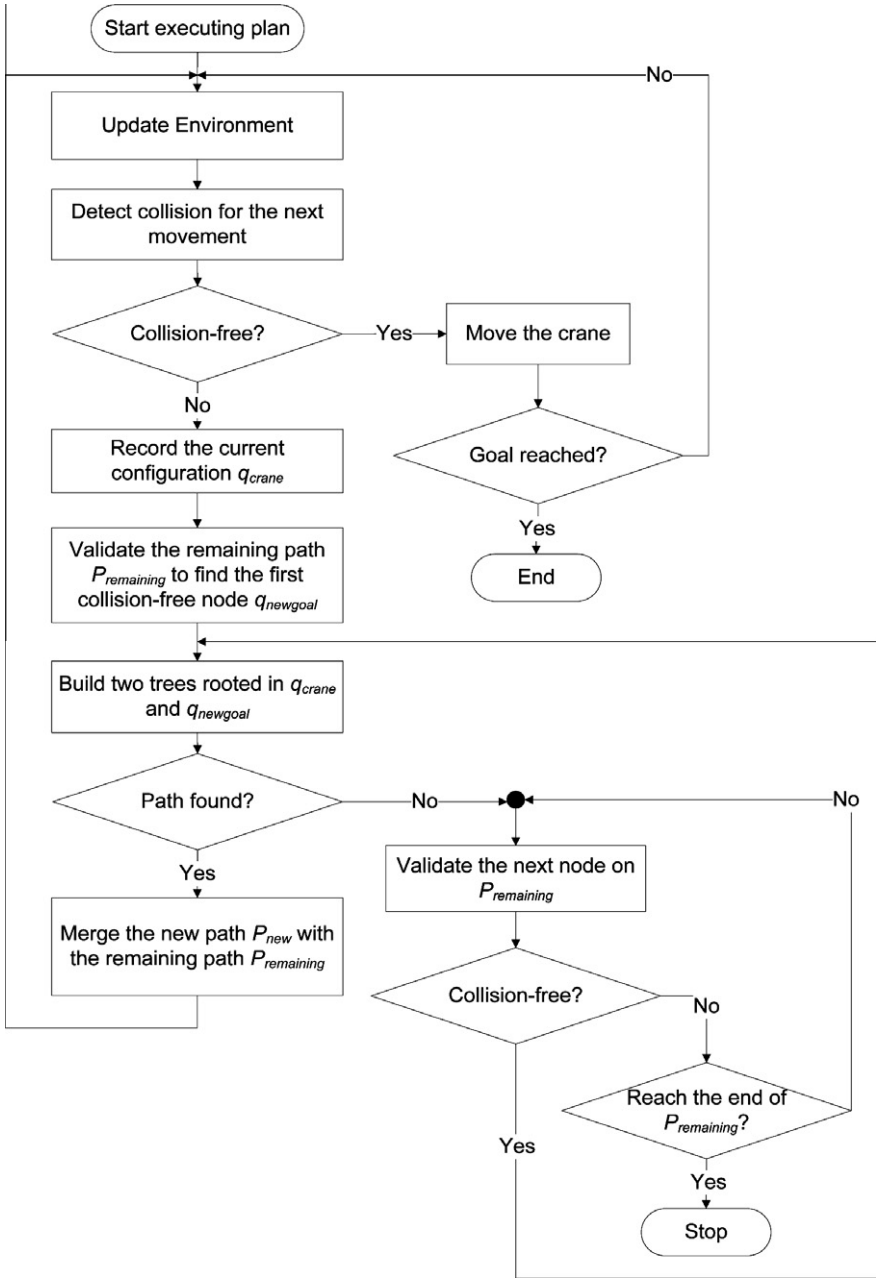


Fig. 7. Flowchart of re-planning algorithm DRRT-Con-Con-Mod.

configurations. If the connection succeeds, the new partial path P_{new} is merged with the remaining path to form a new path (Line 24). If the connection fails, the following nodes on the remaining path are evaluated and attempts are made until successful connection is made between that node and the current configuration node.

The differences between the proposed dynamic motion planning algorithm and the DRRT algorithm [11] discussed in Section 3.4 are the following: (1) a dual-tree algorithm is used in the present research to find a solution. This algorithm is faster than the one-tree DRRT algorithm; (2) the proposed algorithm ensures immediate safety and ignores obstacles that are far away from the current node on the path since these obstacles may move before the crane reaches that part of the path. Thus, the proposed algorithm reduces the calculation time compared with the DRRT algorithm since it does not trim the entire tree. Due to the greediness of RRT-Con-Con-Mod, regenerating a small part of the path is expected to be very quick. To the best of our knowledge, no research has considered dual-tree RRT algorithms for dynamic motion planning.

5. Case studies

A simulation environment is built in Autodesk Softimage [3], where a scene is created with two identical hydraulic cranes (TMS870/TTS870) [12]. CAD models of the cranes are imported into Softimage. A hierarchy of components and kinematics is created in a 3D environment. Four DoFs are considered in the current work, which are boom swing, boom luffing, boom extension, and hook lifting. In addition, a model of a steel structure with 596 elements is created in the simulation environment. Static objects are defined by grouping them under a specific model with the name *obstacles*. The algorithm then considers all objects in this model as static obstacles and performs collision checking during the planning

Please cite this article in press as: C. Zhang, A. Hammad, Improving lifting motion planning and re-planning of cranes with consideration for safety and efficiency, Adv. Eng. Informat. (2012), doi:10.1016/j.aei.2012.01.003

```
DRRT-Con-Con-Mod(q_init, q_goal)
1. P = RRT-Con-Con-Mod (q_init, q_goal);
2. if P = Failed
3.     return Failed; //There is no feasible path
4. while (q_crane ≠ q_goal)    //Start to execute the path
5.     q_next = Read-Next-Node(P);
6.     while (q_crane ≠ q_next)
7.         q_inter = Interpolate(q_crane, q_next); //Calculate intermediate node
8.         Update-Environment();
9.         if Collision-Free(q_inter) == True
10.            Move-Crane(q_inter);
11.            q_crane= q_inter;
12.        else
13.            P_remaining = Validate-Nodes(P_remaining);
14.            q_newgoal = First-Node(P_remaining);
15.            P_new = RRT-Con-Con-Mod(q_crane, q_newgoal);
16.            while P_new == Failed
17.                if q_newgoal = q_goal
18.                    return Failed;
19.                else
20.                    P_remaining.Remove(q_newgoal);
21.                    P_remaining = Validate-Nodes (P_remaining);
22.                    q_newgoal = First-Node(P_remaining);
23.                    P_new = RRT-Con-Con-Mod (q_crane, q_newgoal);
24.            P = Merge-Path(P_new, P_remaining);
25.            q_next = Read-Next-Node(P);

Validate-Nodes (path)
26. repeat for all the nodes q on path
27.     if Collision-Free(q) == False
28.         path.Remove(q);
29.     else
30.         return path;
31. return Stop;
```

Fig. 8. Dynamic re-planning algorithm DRRT-Con-Con-Mod.

stage. Dynamic objects are defined by applying motion information to 3D objects [1].

The ranges of the dimensions of the *C-space* vary according to the size of the counterweight and the size of the lift weight. The example shown in Section 4.1.4 is used in the present case study. The length of the cable could vary between 0 and the distance from the tip of the boom to the ground. In the present research, the area under the boom, defined by the plane of the center line of the boom and the cable, is considered for collision detection to prevent the case of having the booms of two cranes crossing each other [1]. The initial and goal configurations of the crane are shown in Fig. 9. It should be noticed that the fourth DoF (hook up and down) does not contribute to the smoothness of the path.

5.1. Case study-1: motion planning

To evaluate the proposed motion planning algorithm, two major criteria are considered: calculation time and path quality. Calculation time should be as short as possible in near real-time applications.The calculation times are based on the CPU time for an Intel Pentium dual-core processor with 1.83 GHz. Path quality is evaluated in terms of smoothness and plan execution time, as discussed in Section 4.1.2.

Fifty-one random seed numbers (0–50) were used to compare the planning time for each algorithm. Table 2 lists the times spent for finding a feasible path using RRT, RRT-Con-Con and RRT-Con-Con-Mod. It was found that RRT took more time to find a path, whereas the dual-tree algorithms found a path much faster due to the greediness of the dual-tree structure and the *Connect* function. The RRT-Con-Con-Mod spent slightly more time than RRT-Con-Con to check the threshold of the roughness value of the nodes. In addition, the average path roughness value is very similar for the three algorithms.

The same random seed numbers were used sequentially in the anytime algorithm to evaluate the improvement that can be achieved using RRT-Con-Con-Mod. Table 3 shows the path smoothness improvement using different threshold values. The γ value was initialized with the value of 1.7 and was reduced by a step of 0.1 until 1.1. No time limit was set in order to evaluate the performance of the anytime algorithm. The program stopped

Please cite this article in press as: C. Zhang, A. Hammad, Improving lifting motion planning and re-planning of cranes with consideration for safety and efficiency, Adv. Eng. Informat. (2012), doi:10.1016/j.aei.2012.01.003

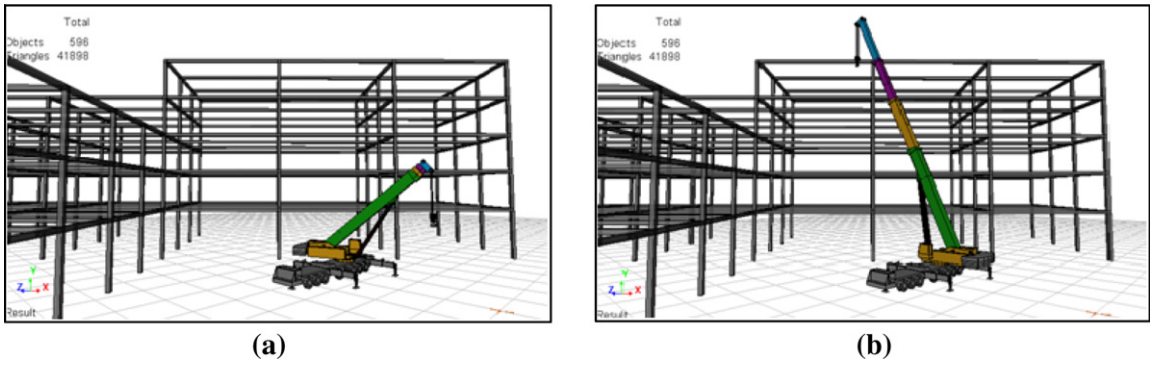


Fig. 9. Initial and goal configurations of the crane. (a) Initial configuration. (b) Goal configuration.

Table 2
Comparison of calculation time of three algorithms.

Algorithm	Average calculation time (s)	Shortest time (s)	Longest time (s)	Std. dev. (s)
RRT	16.95	3.56	39.53	8.302
RRT-Con-Con	1.59	0.61	3.16	0.594
RRT-Con-Con-Mod	1.91	0.61	4.11	0.718

after finishing the calculations using all of the 51 random seed numbers. The results were sorted according to roughness values. It was found that in the top 14 cases, where the paths are smoother, there was no improvement when applying the threshold. By contrast, in the remaining 37 cases where the path smoothness has a potential to be improved, there are 24 cases that the path smoothness has been improved. In these 24 cases, an average improvement of 11.51% better smoothness occurred compared with the paths found using RRT-Con-Con. The best solution among these cases was 29.89% better than the path generated by RRT-Con-Con. Fig. 10 shows the smoothness improvement during the time taken for running 51 cases (seed numbers) with 320 run times (447.5 s). An initial path is found with a roughness value of 3.025. After 9 run times (16.31 s) by reducing γ value and changing the random seeds, a better path is found with a roughness value of 2.563. Then at run time 89 (146.75 s), a better path is found with a roughness value of 2.372. Table 4 shows the roughness values and the execution times of these three paths (P_1, P_2, P_3) and the improvement percentages of P_3 compared to P_1. The execution time is calculated by assuming the speeds of the movements of the boom as 3.75°/s for swinging, 1.04°/s for luffing, and 0.124 m/s for extending the boom [26]. It can be seen that the smoothness and the execution time are improved by 21.59% and 16.86%, respectively. Using $\alpha = \beta = 1$ and $w_t = w_s = 0.5$ in Eq. (4), the costs of the three paths are calculated and shown in the same table. The improvement of the path cost is 18.32%. As explained in Section 4.1.2, other weights can be used in Eq. (4).

Table 3
Path smoothness improvement using different threshold values.

Threshold multiplier (γ) used in RRT-Con-Con-Mod	1.6	1.5	1.4	1.3	1.2	1.1	
Number of improved cases	2	3	5	4	7	3	Total number: 24
Averaged improvement (%)	9.603	6.729	10.565	6.769	11.927	16.183	11.51

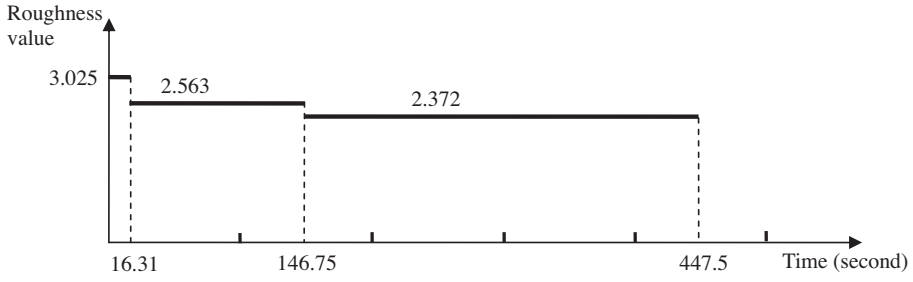


Fig. 10. Smoothness improvement.

Table 4
Roughness values, execution times, and costs of three paths.

Path	P_1	P_2	P_3	Improvement of P_3 compared to P_1 (%)
Path cost values				
Roughness value	3.025	2.563	2.372	21.59
Execution time (min)	6.82	6.54	5.67	16.86
Cost of the path	4.923	4.55	4.021	18.32

Please cite this article in press as: C. Zhang, A. Hammad, Improving lifting motion planning and re-planning of cranes with consideration for safety and efficiency, Adv. Eng. Informat. (2012), doi:10.1016/j.aei.2012.01.003

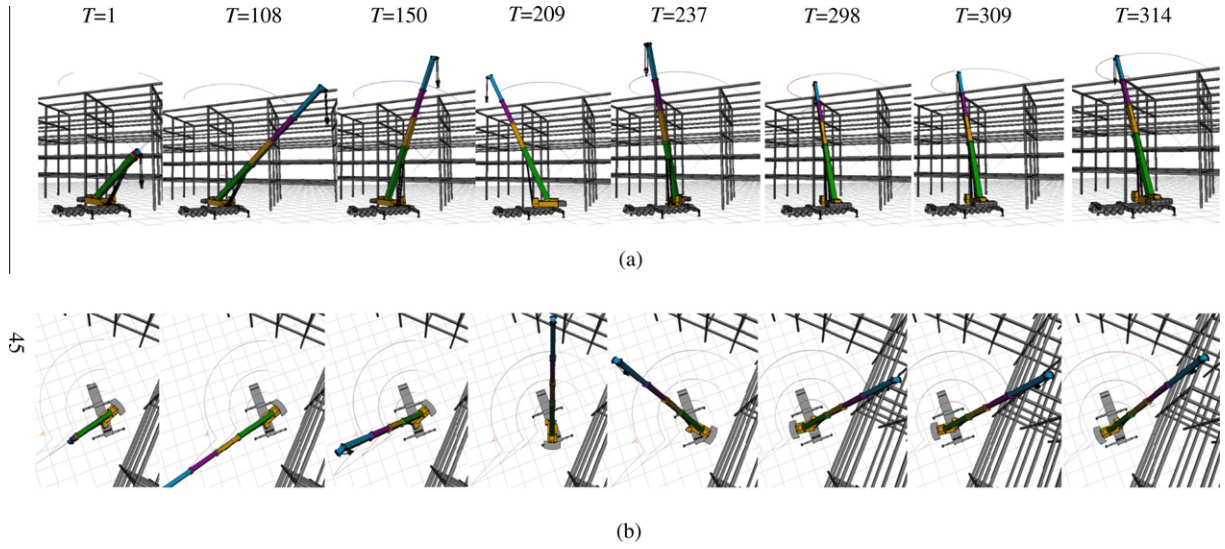


Fig. 11. The initial path (P_1). (a) 3D view of motion plan. (b) Top view of motion plan.

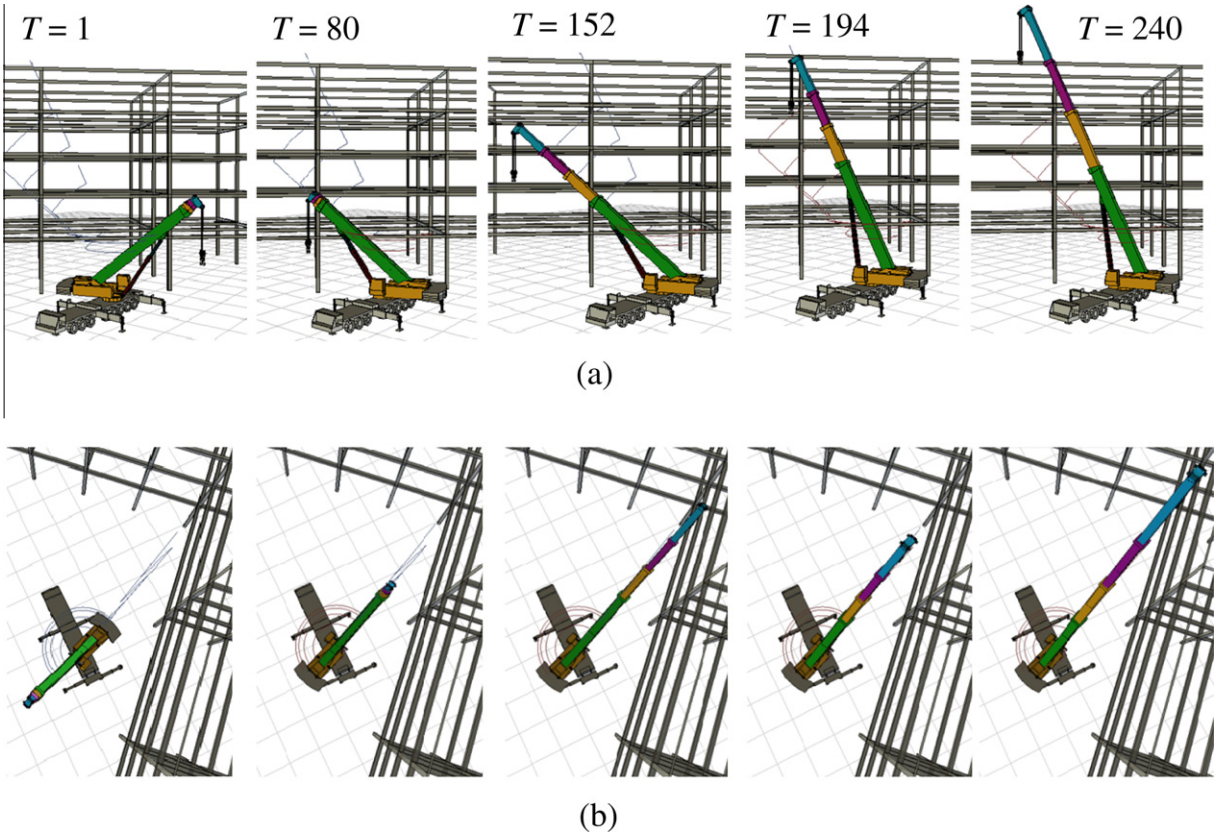


Fig. 12. The improved path (P_3). (a) 3D view of motion path. (b) Top view of motion path.

Fig. 11 shows eight snapshots of path P_1 at several frames of the simulation. This path includes several backward movements due to the randomness of the algorithm. T represents the time frame used in the simulation where 32 frames equal to 1 s. Between frame 1 and 108, the crane extends its boom, and then raises its boom until frame 150. After that, the crane starts rotating the boom clockwise until frame 209. However, a counterclockwise rotation occurs between frames 209 and 237. Next, the crane continues rotation clockwise until frame 298 and raises the boom until frame 309. Then, again, the crane rotates the boom counterclockwise to reach the goal. Because of the counterclockwise rotation, the roughness value is large and redundant movements occur.

After applying the anytime algorithm, the improved path (P_3) is shown in Fig. 12. The roughness value of this path is 2.372, which is the best path found by using RRT-Con-Con-Mod. The motion path starts by rotating the boom clockwise; and, then, at frame 80, the boom is extended. After that, at frame 152, the boom is raised until frame 194, and finally the boom is extended to reach the goal.

Please cite this article in press as: C. Zhang, A. Hammad, Improving lifting motion planning and re-planning of cranes with consideration for safety and efficiency, Adv. Eng. Informat. (2012), doi:10.1016/j.aei.2012.01.003

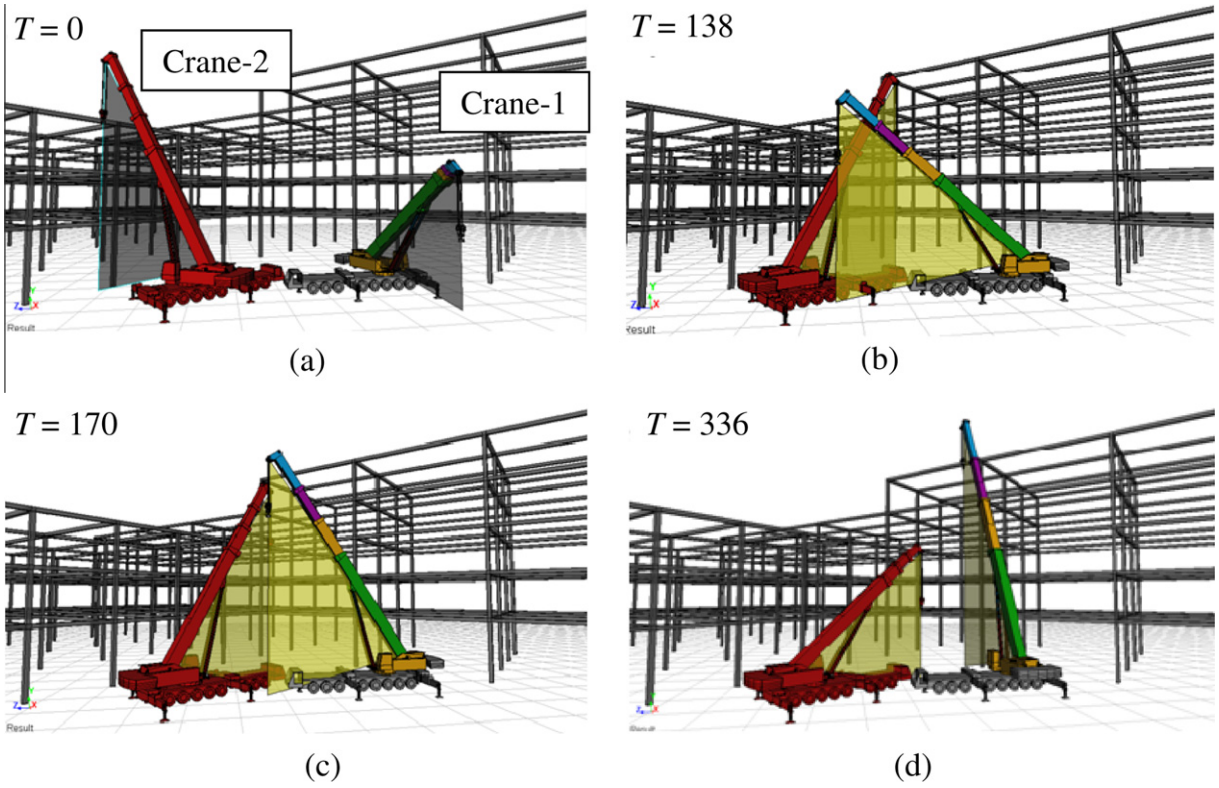


Fig. 13. Motion re-planning in the case of two cranes working in the same area. (a) Initial configuration. (b) Potential collision detected. (c) Collision avoided. (d) Goal configuration.

Table 5
Comparison of re-planning times.

Algorithm	Average calculation time (s)	Shortest time (s)	Longest time (s)	Std. dev. (s)
DRRT	5.98	0.17	26.97	9.63
DRRT-Con-Con-Mod	1.47	0.11	5.27	1.40

5.2. Case study-2: motion re-planning

The present case study focuses on testing the motion re-planning algorithm. In this case,as shown in Fig. 13, a second crane (Crane-2) is located in the same area of Crane-1 used in Case Study-1. It is assumed that Crane-2 has a higher priority than Crane-1 based on the decision of the site manager or some predefined priority rules [36]. While swinging its boom 90° clockwise, Crane-2 may become an obstacle for Crane-1. When a potential collision is detected as shown in Fig. 13b, the high-priority crane (Crane-2) was considered as an obstacle for the low-priority crane (Crane-1). Re-planning is triggered to re-plan the path for Crane-1 by raising its boom to avoid collision as shown in Fig. 13c and d.

A comparison is carried out between the proposed dynamic algorithm DRRT-Con-Con-Mod and the basic DRRT algorithm based on the concept of Ferguson [10]. The same random seed numbers are used for both algorithms to generate initial motion plans for Crane-1. Due to the randomness of the algorithms, the potential collision of the two cranes does not always occur. However, in some cases, more than one re-planning occured during the task when Crane-2 was detected several times as an obstacle. In these cases, the calculation time of each re-planning was taken into account in calculating the average re-planning time. The average re-planning time of the DRRT-Con-Con-Mod algorithm was 1.47 s compared with 5.98 s for DRRT, resulting in a reduction in re-planning time by a factor of four. Furthermore, DRRT cannot guarantee that a feasible path will be found in each re-planning case. In the 22 cases in which re-planning occured, there were 10 cases in which the DRRT failed to find a new path. Whereas in all the re-planning cases (23 cases) of DRRT-Con-Con-Mod, the algorithm successfully found a new path. Table 5 shows the comparison of re-planning times produced by DRRT and DRRT-Con-Con-Mod.

6. Concluding remarks and future work

The present research has proposed an innovative approach to improve safety and efficiency using advanced motion planning and re-planning algorithms for crane operations. In this paper, four criteria, which are efficiency, optimality, reusability, and safety, have been defined for the selection of the appropriate motion planning algorithm for cranes. Based on these criteria, RRT algorithms have been selected as the basic algorithms used in this research due to their quick calculation time and their ability of dealing with high dimensional problems. Detailed review of RRT algorithms has been carried out and several variations have been investigated, such as dual-tree algorithms and algorithms for path quality improvement. Based on this review, we have developed a new algorithm called RRT-Connect-Connect-Modified (RRT-Con-Con-Mod) for crane motion planning. The main characteristics of this algorithm are the following: (1) It is a dual-tree RRT algorithm, which generates two trees from the initial and goal configurations; (2) A cost function is used to evaluate the quality of the path by taking into account the smoothness of the path and the time taken to execute the path; (3) Engineering constraints are considered to generate safe paths for the cranes and to avoid tip over due to overloading.

Please cite this article in press as: C. Zhang, A. Hammad, Improving lifting motion planning and re-planning of cranes with consideration for safety and efficiency, Adv. Eng. Informat. (2012), doi:10.1016/j.aei.2012.01.003

In the case of re-planning, a dynamic re-planning algorithm has been proposed to efficiently repair the path when the environment is updated. Compared with the DRRT algorithm proposed by Ferguson [10], the proposed dynamic algorithm maintains a focus on the path instead of the entire tree by regenerating a partial path to replace the part that is not collision-free due to new obstacles. The advantage of the proposed algorithm is that the time for trimming the entire tree is eliminated and an immediate collision-free movement is ensured for the crane, thereby reducing safety risks. Regenerating the trees and finding a feasible partial plan are carried out quickly thanks to the greediness of the dual-tree structure and the *Connect-Mod* function.

Comparisons have been made between RRT, RRT-Con-Con and RRT-Con-Con-Mod for motion planning, and between DRRT and DRRT-Con-Con-Mod for re-planning. The results show that the path smoothness is improved by applying the anytime algorithm while gradually narrowing the area for node sampling based on the roughness value. An average improvement of 11.51% better smoothness has been obtained compared with the paths found by using RRT-Con-Con. The calculation time of RRT-Con-Con-Mod was much shorter than that of the RRT algorithm. The best path found using the anytime algorithm shows an improvement in smoothness and execution time of 21.59% and 16.86%, respectively. The cost of the path is consequently reduced by 18.32%. As for re-planning, a reduction in re-planning time by a factor of four is achieved by using the proposed dynamic algorithm. Furthermore, the DRRT-Con-Con-Mod algorithm was always able to find a new path during re-planning.

Our future work will focus on the testing in real projects using sensors for environment information updating.

Acknowledgements

This research is supported by grants from the Natural Sciences and Engineering Research Council of Canada (NSERC) and the *Institut de recherche Robert-Sauvé en santé et en sécurité du travail* (IRSST). The help of Mr. Homam Al-Bahnasi in building the simulation environment is appreciated.

References

[1] H. AlBahnassi, Real-time motion planning and simulation of cranes in construction, Master thesis submitted to the Department of Building, Civil and Environmental Engineering at Concordia University, 2009.
[2] M.S.A.D. Ali, N.R. Babu, K. Varghese, Collision free path planning of cooperative crane manipulators using genetic algorithm, Journal of Computing in Civil Engineering, ASCE 19 (2) (2005) 182–193.
[3] Autodesk Softimage, 2010. Available from: <usa.autodesk.com/adsk/servlet/pc/index?id=13571168&siteID=123112> (June 25, 2010).
[4] J. Berg, D. Ferguson, J. Kuffner, Anytime path planning and replanning in dynamic environments, in: Proceedings of the 2006 IEEE International Conference on Robotics and Automation, May 2006, Orlando, Florida, US, 2006, pp. 2366–2371.
[5] F. Bosche, Automated recognition of 3D CAD model objects in laser scans and calculation of as-built dimensions for dimensional compliance control in construction, Advanced Engineering Informatics 24 (1) (2010) 107–118.
[6] D. Brandt, Comparison of A* and RRT-Connect motion planning techniques for self-reconfiguration planning, in: Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, October 2006, Beijing, China.
[7] J.R. Bruce, Robot Motion Planning, Computer Science Department Carnegie Mellon University, 2004. Available from: <www4.cs.umanitoba.ca/~jacky/Robotics/Papers/Bruce-ERRTMotionPlanningSlides.pdf> (June 25, 2010).
[8] H. Choset, K.M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L.E. Kavraki, S. Thrun, Principles of Robot Motion – Theory, Algorithms, and Implementations, The MIT Press, Cambridge, 2005.
[9] Cranimax, 2010. Available from: <www.cranimax.com> (June 25, 2010).
[10] D. Ferguson, Single Agent and Multi-agent Path Planning in Unknown and Dynamic Environments, Ph.D. dissertation submitted to the Robotics Institute Carnegie Mellon University, Pittsburgh, Pennsylvania, 2006.
[11] D. Ferguson, N. Kalra, A. Stentz, Replanning with RRTs,in: Proceedings of the 2006 IEEE International Conference on Robotics and Automation, May 2006, Orlando, Florida, US, 2006, pp. 1243–1248.
[12] Grove Crane, TMS870/TTS870 Product Guide, Manitowoc Crane Group, 2008.
[13] M. Garber, M.C. Lin, Constraint-based motion planning for virtual prototyping, in: Proceedings of ACM Symposium on Solid Modeling and Applications, Saarbrucken, Germany, 2002.
[14] R. Howard, B.C. Bjork, Building information modelling-experts' views on standardisation and industry deployment, Advanced Engineering Informatics 22 (2) (2008) 271–280.
[15] V.R. Kamat, J.C. Martinez, Visualizing simulated construction operations in 3D, Journal of Computing in Civil Engineering, ASCE 15 (4) (2001) 329–337.
[16] S.C. Kang, E. Miranda, Planning and visualization for automated robotic crane erection processes in construction, Automation in Construction 15 (2006) 398–414.
[17] S.C. Kang, E. Miranda, Numerical methods to simulate and visualize detailed crane activities, Computer-Aided Civil and Infrastructure Engineering (24) (2009) 169–185.
[18] S.C. Kang, H.L. Chi, E. Miranda, Three-dimensional simulation and visualization of crane assisted construction erection process, Journal of Computing in Civil Engineering, ASCE 23 (6) (2009) 363–371.
[19] S. Karaman, E. Frazzoli, Incremental sampling-based algorithms for optimal motion planning, in: Proceedings of the 2010 Robotics: Science and Systems Conference, June, 2010, Zaragoza, Spain. Available from: <www.robotics-proceedings.org/rss06/p34.pdf> (August 9, 2010).
[20] S.K. Kim, J.S. Russell, K.J. Koo, Construction robot path-planning for earthwork operations, Journal of Computing in Civil Engineering, ASCE 17 (2) (2003) 97–104.
[21] J.J. Kuffner, S.M. LaValle, RRT-connect: an efficient approach to single-query path planning, in: Proceedings of IEEE International Conference on, Robotics and Autonomous Systems, 2000, pp. 995–1001.
[22] S.M. LaValle, Rapidly-exploring random trees: a new tool for path planning, Computer Science Dept., Iowa State University, 1998. Available from: <msl.cs.uiuc.edu/~lavalle/papers/Lav98c.pdf> (June 25, 2010).
[23] S.M. LaValle, Planning Algorithm, Cambridge University Press, New York, 2006.
[24] S.M. LaValle, J.J. Kuffner, Rapidly-exploring random trees: Progress and prospects, in: Algorithmic and Computational Robotics: New Directions, A K Peters, Wellesley, MA, 2001.
[25] T. Lozano-Pérez, M. Wesley, An algorithm for planning collision-free paths among polyhedral obstacles, Communications of the ACM 22 (10) (1979) 560–570.
[26] Manitowoc, Manitowoc products – GMK6300, 2010. Available from: <http://www.manitowoccranes.com/MCG_GRO/Products/UK/GMK6300.asp> (June 25, 2010).
[27] NIOSH, The National Institute for Occupational Safety and Health, Goal 1: Reduce the major risks associated with traumatic injuries and fatalities in construction, 2007. Available from: <www.cdc.gov/niosh/nas/construction> (June 25, 2010).
[28] H.R. Reddy, K. Varghese, Automated path planning for mobile crane lifts, Computer-aided Civil and Infrastructure Engineering 17 (2002) 439–448.
[29] P.L. Sivakumar, K. Varghese, N.R. Babu, Automated path planning of cooperative crane lifts using heuristic search, Journal of Computing in Civil Engineering, ASCE 17 (3) (2003) 197–207.
[30] A.R. Soltani, H. Tawfik, J.Y. Goulermas, T. Fernando, Path planning in construction sites: performance evaluation of the Dijkstra, A*, and GA search algorithms, Advanced Engineering Informatics 16 (4) (2002) 291–303.
[31] M.W. Spong, F.L. Lewis, C.T. Abdallah, Robot Control – Dynamics, Motion Planning, and Analysis, IEEE Press, IEEE Control Systems Society, 1992.
[32] A. Stentz, The focused D* algorithm for real-time replanning,in: Proceedings of the International Joint Conference on Artificial Intelligence, August 1995, Montreal, Canada.
[33] H.P. Tserng, B. Ran, J.S. Russell, Interactive path planning for multi-equipment landfill operations, Automation in Construction 10 (2000) 155–168.
[34] C. Urmson, R. Simmons, Approaches for heuristically biasing RRT growth, in: Proceedings of the IEEE International Conference on Intelligent Robots and Systems IROS, October 2003, Las Vegas, Nevada, US, pp. 1178–1183.
[35] K. Varghese, P. Dharwadkar, J. Wolfhope, J.T. O'Connor, A heavy lift planning system for crane lifts, Microcomputers in Civil Engineering 12 (1) (1997) 31–42.
[36] C. Zhang, Improving crane safety by agent-based dynamic motion planning using UWB real-time location system, Ph.D. dissertation submitted to the Building, Civil and Environmental Engineering Department of Concordia University, 2010.
[37] C. Zhang, A. Hammad, H. AlBahnassi, Collaborative multi-agent systems for construction equipment based on real-time field data capturing, ITcon Special Issue Next Generation Construction IT: Technology Foresight, Future Studies, Roadmapping, and Scenario Planning 14 (2009) 204–228. Available from: <www.itcon.org/2009/16>.
[38] C. Zhang, A. Hammad, S. Rodriguez, Crane pose estimation using uwb real-time location system, Journal of Computing in Civil Engineering, ASCE, accepted for publication.
[39] C. Zhang, A. Hammad, J. Bentahar, Multi-agent-based approach for real-time collision avoidance and path re-planning on construction sites, in: The 28th International Symposium on Automation and Robotics in Construction, Seoul, Korea, 2011.
[40] C. Zhang, A. Hammad, T.M. Zayed, G. Wainer, H. Pang, Cell-based representation and analysis of spatial resources in construction simulation, Journal of Automation in Construction 16 (4) (2007) 436–448.
[41] S. Zilberstein, S. Russell, Approximate Reasoning using Anytime Algorithms, Imprecise and Approximate Computation, Kluwer Academic Publishers, 1995.

Please cite this article in press as: C. Zhang, A. Hammad, Improving lifting motion planning and re-planning of cranes with consideration for safety and efficiency, Adv. Eng. Informat. (2012), doi:10.1016/j.aei.2012.01.003