Lift Path Planning for Telescopic Crane Based-on Improved hRRT

Yuanshan Lin, Xin Wang, Di Wu, Xiukun Wang, and Shunde Gao

***Abstract*—The hRRT algorithms based on heuristics can solve many path planning problems, but its efficiency can still be improved. We deeply analyze the details of hRRT, and enhance it by using bi-directional tree and adopting kd-tree as secondary data structure to select nearest neighbors. At last, the improved hRRT is used for lift path planning for telescopic crane, and the efficiency of the improved hRRT is verified. The result of the experiment shows that the enhanced hRRT can dramatically shorten the plan time without reducing the quality of path.**

***Index Terms*—Rapidly-exploring random tree, hRRT, path planning, crane lifting.**

I. Introduction

Many path planning methods are proposed in the field of robotics, such as Grid Method, Roadmap Method, and Artificial Potential fields Functions. However, these deterministic algorithms can not cope with the "exponential explosion" problem, and they can only find the exact solution in low-dimensional space or under special conditions. To this end, some approaches based-sampling are proposed, such as Probabilistic Road Map (PRM)[1], Rapidly-exploring Random Tree (RRT)[2]. Unlike PRM, RRT does not require the preprocessing of building roadmap, but follows the state equations of control theory and generate new states increasingly under the controlling amount until reaching the target. Besides, RRT has been used to solve the problems that other deterministic planning methods fails to solve, especially in high dimensional space. RRT has been widespread concern since it was proposed. So far there are numerous extensions and variants of RRT. For example some researchers presented goal biased RRT[3], bidirectional RRT (shortened Bi-RRT)[4]-[6]. But the above and basic RRTs can not effectively control the path quality. Urmson Simmons put forward a RRT algorithm based-on heuristic searching (h-RRT)[7], which is to inspire the spanning tree with growing to the blank area which can produce optimal path. The evaluation function of path cost is establishment based on the size of Voronoi region. Sampling the space points based on path cost can choose the high-quality nodes to be extended, and optimize the final path. However, the planning time increases because it need make several attempts to get a good node and it is difficult to grow a node nearby the goal, especially in high dimension space.

In this paper, the performance of hRRT is improved by taking some strategies. The improvements are mainly the following two points: 1) changing the original algorithm to bi-direction extending; 2) using kd-tree as the secondary data structure to improve the efficiency of selecting node. Finally, we apply the improved hRRT, called BihRRT, to solve the lift path planning problem of a telescopic crane.

Manuscript received November 5, 2012; revised March 10, 2013.

Yuanshan Lin, Di Wu, and Xiukun Wang are with the School of Computer Science, Dalian University of Technology, Dalian, Liaoning, P.R. China (e-mail: Linyuanshan2008@126.com, Jsjwxk@dlut.edu.cn).

Xin Wang and Shunde Gao are with the School of Mechanical Engineering, Dalian University of Technology, Dalian, Liaoning, P.R. China (e-mail: Wangxbd21@163.com, Gaoshunde@163.com).

II. The Basic hRRT Algorithm

The hRRT is an improved RRT, which uses heuristic cost function to guide the expanded tree's growth. It takes the initial node as the root, and selects new nodes by random sampling to generate an expanded tree. When the leaf node of the random tree contains the target node or the tree enters the target area, the path is founded. The basic hRRT algorithm pseudo code is as follows:

```
GENERATE_RRT (x_init)
    T.init (x_init);
    While (x_goal not in T)
        { x_rand, x_near} SELECT_NODE( T );
        EXTEND ( T, x_rand, x_near);
    Return T;

SELECT_NODE(T)
    Do
      x_rand ← RANDOM_STATE();
      x_near ← NEAREST_NEIGHBOR(x,T);
      m_quality ← 1– (x_near cost – T.opt_cost) / (T.max_cost – T.opt_cost);
      m_quality ← min(m_quality, T.prob_floor);
      r ← RANDOM_VALUE();
    while (r > m_quality);
    return { x_rand, x_near };

EXTEND (T, x_rand, x_near)
    If (NEW_STATE (x_rand, x_near, x_new, u_new) then
    T.add_vertex (x_new);
    T.add_edge (x_new, x_new, u_new);
    CALCULATE_COST (x_new);
    T.max_cost←max (x_new.cost, T.max_cost);
     Return;
```

Firstly, function RANDOM_STATE( ) select a sampling point x_{rand} from the CS, which can by using biased sampling strategy to guide the expanded tree growing to target area. Secondly, function NEAREST_NEIGHBOR (x_{rand}, T) select a node (x_{near}) nearest to x_{rand} from the random tree T, and

evaluate the quality of the node using cost function and select the node to a certain probability. Finally, function NEW_STATE (T, x_{rand}, x_{near}) choose the best input u from x_{rand} to x_{near} and generate a new node. If this new node meets all the constraints, then join the expanded tree to complete an expansion. Repeat the above process until obtain a spanning tree containing the request path.

III. Improved hRRT Algorithm

A. Bi-Directional Extension

In general, hRRT grows in a single RRT. It works well for state spaces of low dimension, but it drawbacks emerges when it faces the path planning problem with high dimension. 1) hRRT tends to slowly explore in the unknown regions because it grows a single tree from initial configuration; 2) in high dimensions it becomes more difficult to randomly wander upon a state is close enough to the goal state for each of the state space variables. So in this paper we borrow the idea of bidirectional RRT [4], [6]. The two trees alternately extend from the initial and the goal states and get a path when the two trees encounter. The difference from the basic bidirectional RRT is that in our method one tree select the node of the other tree as sample in certain probability when it choose state. Thus, it improves the probability for the two trees encountering. The procedural framework is shown below:

```
GENERATE_RRT (x_init, x_goal, T_init, T_goal, K)
{
    T_init.Init(x_init);
    T_goal.Init(x_goal );
      while ( k < K )
       {
  if(EXTEND (T_init, NULL, true, x_new1 ) )
  {
    if(EXTEND (T_goal, x_new1, false, x_new2))
      if (GapSatisfied(x_new1, x_new2 ) )
                return Path( T_init, T_goal);
       }
       Swap( T_init, T_goal );
    }
}

EXTEND (T, x_target, forward, x_new)
{
    {x_target, x_near }← SELECT_NODE(T, x_target , forward);
    {u, x_new }←SelectInput(x_near, x_target, bSuccess, forward);
    if( bSuccess)
    {
       T.AddVertexandEdge( x_nearest, x_new, u);
      if( T == T_init )
           SP_goal. AddandUpdate( x_new);
      Else

         SP_init. AddandUpdate ( x_new);
    }
}
```

B. Use of Kd-Tree Data Structure

The difference with the basic hRRT is that: in the proposed RRT algorithm, one tree directly selects certain node of the other tree as sample and makes the nearest node as the growing node. It does not need to repeatedly sample and compute the quality assessing function of the nearest node. Thus, it improves the probability of the two trees enouncing and shortens the computing time. In addition, much time is spent in choosing node. The more nodes are in trees, the more time need to select good node. In this paper, we adopt kd-tree to organize the nodes in order to improve the efficiency. The pseudo code of node selecting is shown below:

```
SELECT_NODE (T, x_target , forward)
{
    if(x_target)
    {
        x_near←NEAREST_NEIGHBOR(x_target, T)
        return {x_target, x_near };
    }
    else
    {
        SP= forward? SP_init : SP_goal;
        Do
        {
            x_target ←ChooseTarget( T, SP );
            x_near←NEAREST_NEIGHBOR(x_target, T)
            m_quality←1-
            (x_near.cost-T.opt_cost)/(T.max_cost-T.opt_cos
            t)
            m_quality←min(m_quality, T.prob_floor);
            r←RANDOM_VALUE();
        } while(r> m_quality);
    }
    return {x_target, x_near };
}
```

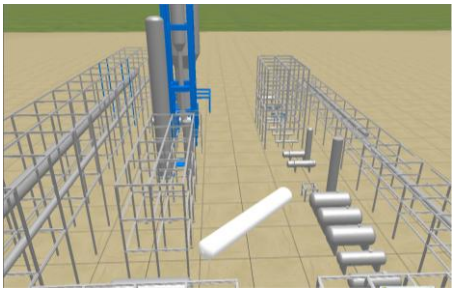
IV. Case Study

A. Lifting Task

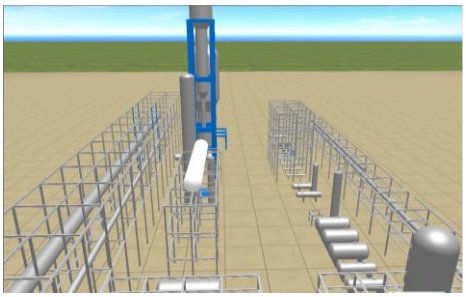
This case is a petrochemical refinery construction of an actual lifting project: the work environment is shown as Fig. 1 (a) and (b). The electric desalting tank lifted is 27m long, 3.8m wide, and 34 tons weight. At the beginning lifting, the equipment (electric desalting tank) lifted is placed at (-26.0, 0.0, -36.0), the angle between the centerline of equipment lifted and the X axis is 0.87 radian, our task is to choose a telescopic crane and transport the equipment lifted using it from the starting position to the installation position (-42.7, 18.4, -38.8) with the angle of the equipment centerline and X axis $\pi/2$ radian, as shown in Fig. 3. It is noted that the coordinates is the coordinates of the center of equipment.

B. Crane Selection

In this case, we choose a telescopic crane according to the length, diameter, weight parameters of the lifted equipment and the work environment. The crane selected is Liebherr LTM1500, and the parameters of the crane selected are shown in Table I.



(a) Initial pose of equipment lifted



(b) Installing pose of equipment lifted

Fig. 1. The initial and installing pose of equipment lifted

The configuration of telescopic crane is shown in Fig. 2, it has three basic freedoms, i.e., lifting movement, turntable slewing, arm luffing, and rope hoisting. In addition, considering collisions in the process of lifting, the motion of hook rotating is also required. All actions are shown in Fig. 2.

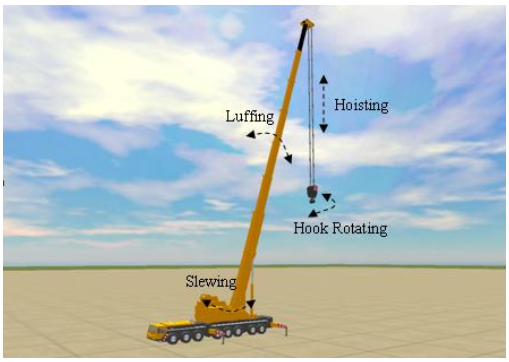


Fig. 2. Actions of the crane

TABLE I: CRANE PARAMETERS

Crane Parameters	Value
Crane Type	LTM1500
Configuration	S
Boom Length(m)	50
Counter Weight (t)	165
Center Weight(t)	43
Hook Weight(t)	5

C. Planning Path Using BihRRT

In the case, h-RRT algorithm is applied to plan an optimal lifting path without collisions.

Defining state space: That h-RRT algorithm planning path is in the state of space, so we first define the state space. The tuple $x = f(\alpha, \beta, \gamma, L)$ represents the state of the crane. Hence, The state-space *X* is a subsets of the 3-dimensional Euclidean space. Where, α is the slewing angle of the crane, within $[-\pi, \pi]$; β is the luffing angle of the boom, within $(\pi/4, \pi/2)$; γ is the rotation angle of the hook, within [0, $\pi/2$]; L is the length of the lifting rope, within [0, 100].

Distance Metric between two states: In this case, the length of the path is defined as Euclidean path length, which is the sum of the Euclidean distances between consecutive positions in the solution trajectory. So we define the distance metric between two states as the length of equipment trajectory generated by crane's input applied, as equation 1.

$$D(x_1, x_2) = |r_1(\alpha_2 - \alpha_1)| + |r_2(\beta_2 - \beta_1)| + |L_{wgt}(\gamma_2 - \gamma_1)| + |L_2 - L_1| \quad (1)$$

where, r_1 is the working radius of the telescopic crane, r_2 is the length of the crane's boomarm, and L_{wgt} is the length of the equipment lifted.

So the entire length of path is defined as $l = \sum_{i=0}^{n-1}(x_i, x_{i+1})$, where n is the number of the nodes in the path.

Applying control: The RRT Algorithm extends the tree by the input steer, so we firstly define the control inputs of the crane as equation 2. Each component represents the slewing, luffing, rotating of hook and lifting of crane, respectively.

$$\dot{x} = [\dot{\alpha}(t) \quad \dot{\beta}(t) \quad \dot{\gamma}(t) \quad \dot{L}(t)]^T \quad (2)$$

Assuming Δt equals to 0.01s and we define a set of inputs for the crane as following:

$$\dot{x}(t)\left(\begin{bmatrix}0.1\\0.0\\0.0\\0.0\end{bmatrix},\begin{bmatrix}-0.1\\0.0\\0.0\\0.0\end{bmatrix},\begin{bmatrix}0.0\\0.05\\0.0\\0.0\end{bmatrix},\begin{bmatrix}0.0\\-0.05\\0.0\\0.0\end{bmatrix},\begin{bmatrix}0.0\\0.0\\1.0\\0.0\end{bmatrix},\begin{bmatrix}0.0\\0.0\\-1.0\\0.0\end{bmatrix},\begin{bmatrix}0.0\\0.0\\0.0\\0.1\end{bmatrix},\begin{bmatrix}0.0\\0.0\\0.0\\-0.1\end{bmatrix}\right)$$

Using h-RRT Algorithm: As shown in Fig. 3, we find the suitable fixed position of the crane according to the initial, installing position of the equipment and crane work environment, and we let the crane locate at (-26.0, 0.0, -50.0). The initial state and the installing state of this problem are (2.16, 1.165, 0.0.59, 30) and (-2.967, 1.24, 0.87, 49) respectively.

Finally, the crane will do action corresponding to the input set, and complete the lifting task.

In this path planning, the experiment is performed on a PC of 2.0GHz Core2 processor, 2G of Memory, Windows XP operating system; C++ language programming; the probability of randomly selecting the goal node is set 0.05;using PQP to do collision detection.

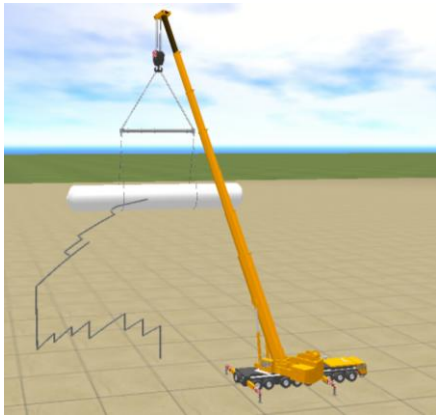
D. Experiment Result and Analysis

The procedure of the crane lifting path planning runs 100 times, the number of iterations in each run process is 10000 times, and the mean value of the parameter of the algorithm performance is shown as Table II. From the table, we can see that: the basic hRRT attains the path with the probability of 64%, while the BihRRT dramatically shorten the computing time without reducing the quality of the path.

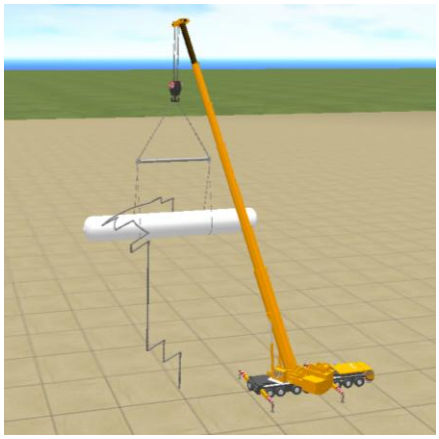
TABLE II: PERFORMANCE OF BASIC HRRT AND ENHANCE HRRT

Algorithms	hRRT	BihRRT
Success Number	64	100
Avg. PlanningTime (s)	52.5	6.1
Avg. Path Length(m)	204.9	191.6
Avg. Nodes in T	6975.1	2777.4
Avg. of C D Calls	68095.6	24972.8

The paths worked out by the basic hRRT using kd-tree and the enhanced hRRT are shown in Fig. 3. It can be seen that, the path produced by the enhanced hRRT is better than the basic one as well.



(a) Path attained by basic hRRT



(b) Path attained by BihRRT

Fig. 3. Paths returned by the two hRRTs

V. Conclusion

In order to improve the efficiency of the basic hRRT path planning algorithm, this paper improves the basic hRRT by using two optimization strategies: Bi-directional extension and use of kd-tree data structure. Based on the improved hRRT, called BihRRT, a lift path planning approach for telescopic crane is proposed, in which state space, metric measure between two states, control inputs are presented. The case simulation illustrates that the proposed approach can dramatically shorten the planning time compared with the basic hRRT, without reducing the quality of path.

Next, the more complex working conditions can be taken into considering and trying to applying this method in the cooperative lifting of two cranes.

Acknowledgment

The authors thank, Liping Zhang, Li Chen and Zhiyi Pan for proof reading and Yufeng Zheng for providing language help.

References

[1] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. on Robotics and Automation*, vol. 12, pp. 566-580, Aug . 1996.

[2] S. M. Lavalle, "Rapidly-Exploring Random Trees: A New Tool for Path Planning," Report No. TR 98-11, 1998.

[3] S. M. LaValle and J. J. Kuffner, "Rapidly-exploring random trees: Progress and prospects," in *Algorithmic and Computational Robotics: New Directions*, B. R. Donald, K. M. Lynch, and D. Rus, ed. A K Peters, Wellesley, MA, 2000, pp. 293-308.

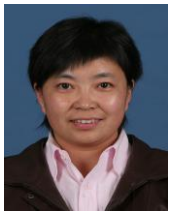
[4] S. M. L. Valle and J. J. Kuffner, "Randomized kinodynamic planning," *International Journal of Robotics Research*, vol. 20, pp. 378-400, May 2001.

[5] J. J. Kuffner and S. M. L. Valle, "RRT-connect: an efficient approach to single-query path planning," in *Proc. IEEE International Conference on Robotics and Automation, San Francisco, CA, USA*, 2000, pp. 995-1001.

[6] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," in *Proc. IEEE International Conference on Robotics and Automation. Proceedings*, vol. 1, pp. 473-479, 1999.

[7] C. Urmson and R. Simmons, "Approaches for heuristically biasing RRT growth," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 2, pp. 1178-1183, Oct. 2003.

Yuanshan Lin received the B.S. degree in Computer Science from the Harbin University of Science and Technology, Heilong Jiang, China in 2005 and M.S. degree in Computer Science from the Dalian University of Technology, Liaoning, China in 2008. He is currently working toward a Ph.D. degree in the School of Computer Science, Dalian University of Technology. His main research interests include topics like path planning, collision detection and 3D visualization.



Xin Wang received the B.S. degree in 1994, M.S. degree in 1997 and Ph.D. degree in 2000, all in the School of Mechanical Engineering, Dalian University of Technology. She is currently an Assistant Professor in School of Mechanical Engineering, Dalian University of Technology. Her main research interests include design theory of heavy equipment, CAD for Complex Structure and computing in mechanical engineering.

Di Wu received the B.S. degree in 1993, M.S. degree in 1996 and Ph.D. degree in 2000, all in the School of Computer Science, Dalian University of Technology. He is currently an Assistant Professor in School of Computer Science, Dalian University of Technology. His main research interests include wireless net, optimization method and application.

Xiukun Wang received the B.S. degree in Engineering Mechanics in 1970 and M.S. degree in Computer Science and Engineering in 1982, both in the Dalian University of Technology, Liaoning, China. She is currently a Professor and doctoral supervisor in School of Computer Science, Dalian University of Technology.

Shunde Gao received the B.S. degree in School of Mechanical Engineering from the Dalian University of Technology, Liaoning, China in 1985 and M.S. degree from the Asia International Open University (MACAU), Macau, China in 2001. He is currently a senior engineer with the rank of a professor. His main research interests include design theory for engineering machinery.