COMPUTATIONAL METHODS FOR COORDINATING MULTIPLE CONSTRUCTION CRANES

Shih-Chung Kang[1] Eduardo Miranda[2]

ABSTRACT

A method for pre-planning and coordinating multiple tower cranes in relatively narrow construction sites is presented. This computational method, referred to as *incremental coordination method*, is capable of considering the kinematics and geometrical constrains of cranes and plan detailed motions for collaborative work of two or more cranes. By following pre-computed motions, the cranes are then able to work closely together and collaborate to complete erection tasks safely. This method allows computers to plan and coordinate the crane activities incrementally from the start to the end of an erection process. This paper explains the details of *incremental coordinate method* and presents an example case, in which two construction cranes work together on a building project where working areas of the crane intersect, hence having the possibility of colliding with each other. The example illustrates the effectiveness of the *incremental coordination method* in planning and synchronizing all motions in the cranes to avoid all the conflicts between cranes and all obstacles in the construction site.

KEY WORDS

multiple cranes, robotics, motion planning, crane coordination, virtual construction, tower crane.

INTRODUCTION

Recently an increasing trend toward the simultaneous use of multiple cranes at a site has been observed. This is particularly true in high-rise construction in which the use of multiple cranes can translate into significant faster rates of construction. However, computational motion planning when two or more cranes are working simultaneously and share a portion of their work area is significantly more complex than the motion planning for a single crane project. There are two main aspects that contribute to the complexity of dealing with multiple cranes: (1) in addition to new obstacles being added by the crane for each element that is erected, the collision detection and motion planning algorithms must also consider new obstacles being created by other cranes on the site; (2) motion planning must consider multiple moving objects, whereas in the case of a singe crane the only moving object is the one being moved by the crane.

[1] Assistant Professor, Department Civil Engrg., Rm. 314 Civil Engrg. Bldg., No. 1, Roosevelt Rd., Sec. 4, National Taiwan Univ., Taipei, Taiwan, Phone +886 33664346, FAX +886 33664346, sckang@ntu.edu.tw

[2] Assistant Professor, Department of Civil and Envir. Engrg., Terman Engineering Center, Rm. 293, Stanford Univ., Stanford, CA 94305. Phone +1 6507234450, FAX +1 6507237514, emiranda@stanford.edu

When dealing with multiple cranes, motion planning involves coordination of their motions in order to avoid possible collisions of any of the cranes with all static objects in the project, as well as possible collisions between the moving cranes and the elements being transported. This paper presents the computational methods that were developed in order to coordinate erection activities of multiple cranes.

Since coordinating the activities of two or more moving cranes in general requires a very large number of calls to collision-checking functions, a particularly efficient computational method for detecting inter-crane collisions is needed. In this paper, the computational methods specifically developed for detecting possible collisions between cranes is explained.

EXISTING METHODS FOR MULTIPLE ROBOTS COORDINATION

In the field of robotics, many methods have been developed for coordinating the motions of multiple robots working closely together. For example, in the automobile industry it is very common that multiple robots are used simultaneously for assembling vehicles. A good summary of previous methods have been presented in various references (e.g., Kant and Zucker, 1986; Warren 1990; Alami et. al., 1995; and Sanchez and Latombe, 2002). In particular, Sanchez and Latombe (2002) have provided an excellent summary of some of the most important methods and have presented a new approach for coordinating multiple robots. These methods deal with the coordination of individual motions that two or more robots must execute in order to carry out a shared task. Specific computational tools are required to generate the paths of each robot, and also to plan the motions of each movable part of each robot to follow the pre-computed path in order to avoid collision with other moving parts of the same robots, with the obstacle(s) being assembled as well as with other robots.

In general we can categorize robot coordination methods into two groups: one for centralized methods and the other one for decoupled methods. Centralized methods require huge computational efforts. They are hard to be applied in practical cases. A decoupled planning method is a simpler and therefore more suitable approach for computer implementation. It composed of two phases: (1) individual planning phase; and (2) the velocity tuning phase. In the first phase, we plan the motions of each robot individually without considering the other robots in the working space. Path-planning methods, such as Probabilistic Roadmap Method (Latombe, 1991) and Rapidly-exploring random tree (Lavalle, 1998 and 2001) are usually introduced in this phase to eliminate possible collisions between each one of the robots and static obstacles in the environment. Since the first phase does not coordinate the motions of the various robots, collisions may occur between the robots. The second phase, velocity tuning phase, aims to remove the inter-robots collisions by tuning their relative velocities along the paths pre-computed in the first phase. This method essentially adds velocity controllers to all robots so that we can adjust robots' velocities to keep a safety distance between them.

In this research, a decoupled coordination method was developed, so the problem about velocity tuning needs to be dealt with. Velocity tuning consists of searching a coordination space. To clearly explain the concept of velocity tuning problem and coordination space here we use two robotic cranes as an example. Let $\tau 1$ and $\tau 2$ be the paths of two robotic cranes (Crane1 and Crane2) computed in the first phase. These two paths avoid possible collisions from static obstacles in working space, but still have possibility to conflict with the other

crane. By forcing the cranes to move along these paths, we reduce the number of degree-of-freedom of each crane into one. Hence, the composite configuration space becomes two dimensions. We denote this composite configuration space as a coordination space. More details about velocity tuning and coordination space have been summarized in multiple published papers, such as (O'Donnell and Lozano-Perez, 1989; Švestka and Overmars, 1998; Sanchez and Latombe, 2002).

INCREMENTAL COORDINATION METHOD

The major effort in this research is to develop a computational method specifically for coordinating multiple cranes on a construction site.

Achieving this goal requires considering more issues than in typical coordination problems in multiple robots. A typical environment for robots involves relatively less changing obstacles in the environment. Take a welding robot in vehicle manufacture for example, the list of obstacles before and after welding may change very little or even the change can be ignored. However, in a construction project, the number of structural elements is increasing as erection progresses. Cranes need to avoid collisions with all structural elements that have previously been erected and find a collision-free path for the each subsequent element to be erected. Namely, the obstacle list is increasing along with construction progress. Therefore, we have to modify existing decoupled coordination method so that we can deal with the coordination problems in a continuously-changing construction environment.

In this research, a new method, referred to as *incremental coordination method* was developed to plan motions for multiple cranes. Similarly to other decoupled coordination methods, this method also is composed of two phases: the first phase is to plan the motions of each crane without considering other cranes; the second phase is to coordinate the activities between cranes. Instead of planning the crane motions for an entire project, the method developed in this research plans the motions of individual crane during a period of time and coordinates the motions of cranes during that time period by using velocity tuning approach.

A two-crane example is used in the following paragraphs to illustrate the main concepts of the proposed method. This case will later be generalized for the cases involving more than two cranes. Assume we would like to synchronize the motions of two cranes, Crane1 and Crane2, in a construction site with the possibility of spatial conflicts. To coordinate the activities of the cranes, the first phase is to perform the motion planning for both Crane1 and Crane2 during a user defined time interval δ. This involves computing collision-free paths of all elements to be erected by Crane1 during a time interval δ, by considering all the elements to be erected by Crane2 during the same time interval δ as obstacles even in the case that these elements have not been erected. At this stage, motion planning avoids the cranes from colliding with elements in their final (erected) position, but does not consider the possibility of the collisions between cranes or the collisions between a crane and a structural element being erecting. The same procedure is performed in Crane2. It regards all the elements to be erected by Crane1 in the end of time interval δ and plan the erection paths for each elements to be erected by Crane2 during this time.

The possible collisions between cranes are then eliminated in the second phase by tuning the velocities of the cranes in the coordination space. In this phase, the motions in each crane required to erect pieces during the time interval δ are adjusted to avoid possible collisions during this time interval. Phase one and two are then repeated at each time interval δ until all the elements in the project have been erected. The time interval usually needs to be a short period of time, for example a time that is approximately equivalent to the average time required to erect three or four elements. This time interval does not need to remain constant and, for example, in the case of high rise construction, δ could increase as the erection progresses, in order to take into account the increase in average time to erect an element as the erection of the building progresses. The process of incremental decoupled method is summarized and illustrated in Figure 1.

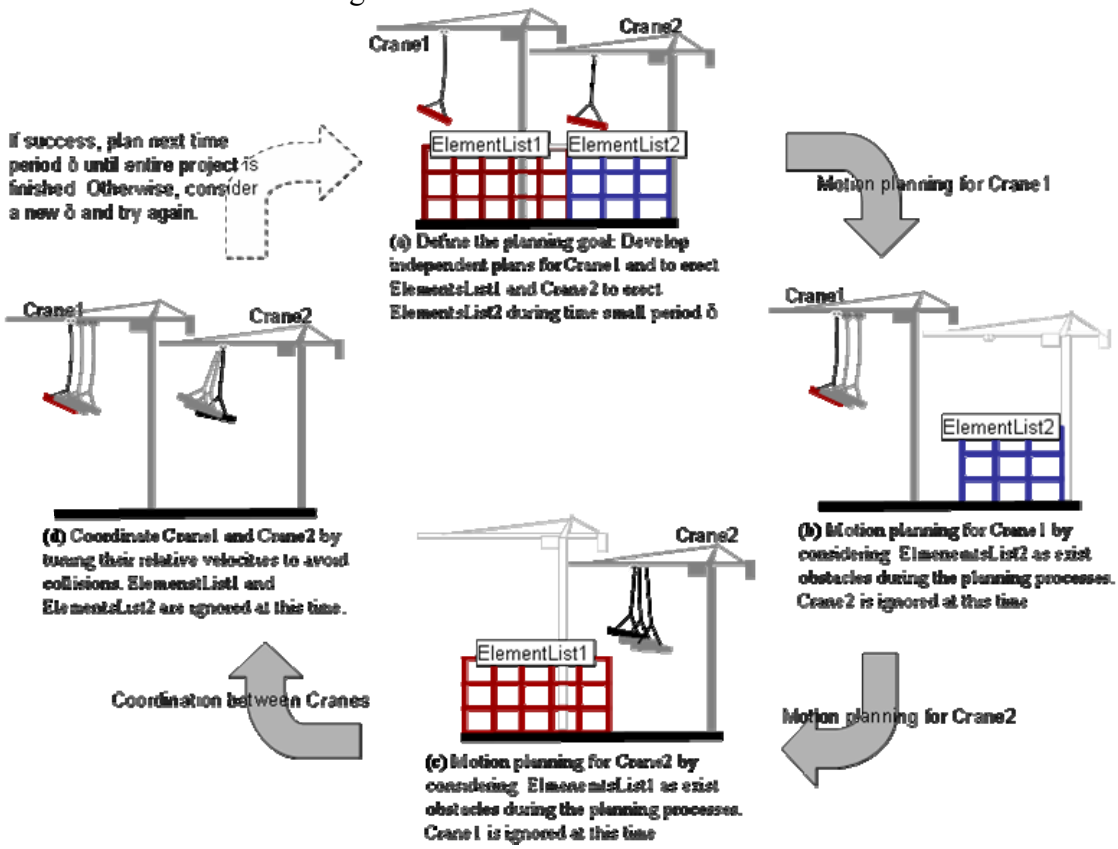

Figure 1: Procedure for Planning and Coordinating a Two-Crane Construction Project

The planning method shown in Figure 1 can be further generalized. Figure 2 presents a general algorithm of incremental decoupled method, which is capable of dealing with the coordination problem in a project involving two or more cranes. The basic idea of this method is to ignore the other cranes and only considering the obstacles in the environment first. Each crane performs its motion planning individually within a pre-defined incremental time. It then employs velocity tuning method to coordinate the activities between cranes. If we cannot find collision-free coordinated paths for all of the cranes, we may try to adjust

(usually shorten) the pre-incremental time and perform the procedure again. Because shorter time periods usually consider fewer obstacles during the calculation, we have better opportunity to successfully find the collision-free planning for all the cranes.

Algorithm *IncrementalDecoupledCoordinator (CraneList CList, ElementList EList):* Coordinate the cranes in crane list to erect the buildings in building list.

```
1:  t ← starting time
2:  REPEAT
3:    DO
4:      δ ← time increment
5:      FOR EACH crane Cr in CList
6:        obstacleList← All the elements except the elements will erected by Cr
7:        Plan the motion of Cr using obstacleList
8:      END FOR
9:    WHILE successfully coordinate cranes in CList during time t to t+ δ
10:   t ← t+ δ
11:  UNTIL all the buildings elements in EList is erected
```

Figure 2: Algorithm of Incremental Coordination Method

THE USE OF INCREMENTAL COORDINATION METHOD

An example case was implemented to demonstrate the use of computational methods to coordinate multiple cranes in a construction site. This example simulates the erection activities of two cranes, which work closely in a building project and partially overlap their working zones. Figure 3 illustrates the layout of the example project.

As shown in Figure 3, two tower cranes (named Crane 1 and Crane 2) are involved in this building project. Working zone of Crane 1 is the working area that Crane 1 can reach. It covers part of the area that the building structure will be erected and also covers the material supply location of Crane 1. Similarly, working zone of Crane 2 is the working area that Crane 2 can reach, covering partially the building structure and also the material supply location of Crane 2. In order to cover the entire building area where the structure will be erected, the working zones of the cranes need to be overlapped. While both cranes move within the overlap area in the same time, two cranes could collide. Therefore, when planning the erection processes for this example project, we can not only consider the erection tasks for individual cranes but also need to eliminate collisions between cranes.

A computer function, *Coordinator*(), was implemented to plan the erection tasks and coordinate the motions between cranes. The inputs of this function include the kinematics of the cranes, the sizes and shapes of each structural element and the three-dimensional model of the structure will be erected. Two computer objects, Crane 1 and Crane 2, were defined to present the kinematics information of the cranes, such as locations, dimensions and operational speeds. In the function, each structural element is also presented by a computer object to convey the geometrical information of the material supplies. Furthermore, construction model (building model) is presented by a computer object, in which the positions of all structural elements and the positions of connections were defined.

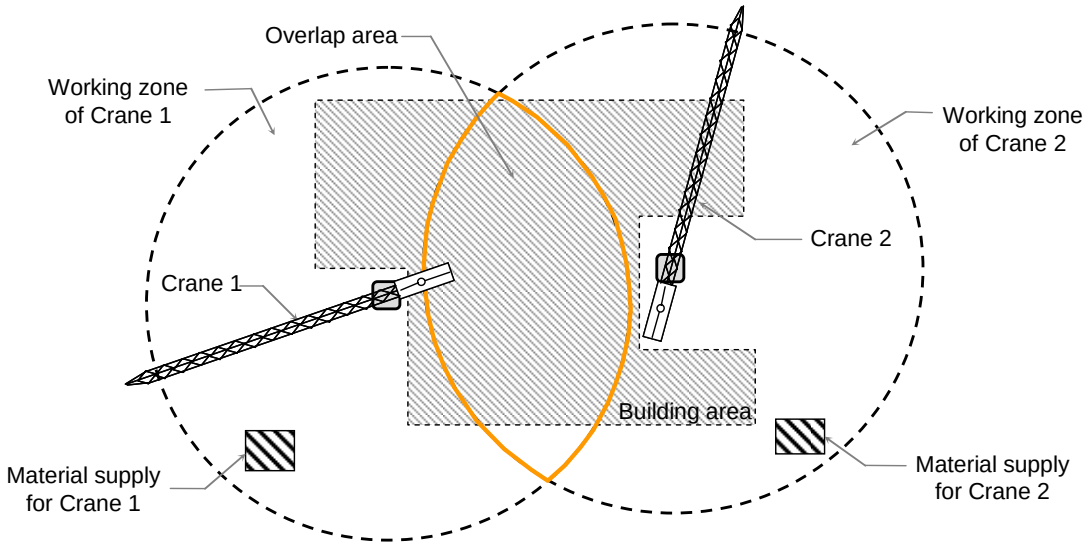

Figure 3: Site Layout of a Two-Cranes Example Project

Having the basic inputs (cranes, materials and construction models), the *Coordinator*() function follows the procedure shown in Figure 2 (i.e. incremental decoupled method) to plan and coordinate the activities between cranes. The main workflow in the *Coordinator*() function is: (1) to generate the obstacle list specifically for Crane 1 and plan its motions to complete given erection tasks without colliding with any obstacles in the list within a predefined incremental time; (2) to generate the obstacle list specifically for Crane 2 and plan its motions to complete given erection tasks without colliding with the obstacles in the list within the incremental time; (3) to save the time histories of the motions of the both cranes (including the time history of the movements of each part of the crane); (4) to coordinate the motions of the crane by applying velocity tuning method; (5) to generate a global time history table so that the time histories of individual cranes can be mapped to global times; and (6) visualize the cranes' motions by forcing both cranes to follow their time histories in the global time history table.

In the workflow described above, three additional major functions are implemented that are called by the *Coordinator*() function. The first one is *generateObstacleList*(), which is capable of dynamically generating a list of obstacles that represent the existing building (the structural elements have been erected) to reflect current construction progress. The second function is *motionPlanner*(), which is used to plan the motions of the given crane to complete the erection tasks and simultaneously avoid the obstacles in the obstacle list. *motionPlanner*() actually includes the path-finding methods described in Kang and Miranda (2004) , which are capable of computing possible collision-free paths for a crane to complete erection tasks efficiently. The third function, *craneCollisionDetector*(), was developed to detect the collisions between cranes. In order to detect possible collisions between the cranes, the cranes are approximated by multiple cylinders. The *craneCollisionDetector*() function essentially computes the distances between groups of cylinders. If there is any collision (clear distance less than a safety margin) between the cylinders in different groups, the cranes are regarded as in collision status.

VISUALIZATION OF THE ERECTION PROCESSES

Figure 4 shows the snap shots generated by running the *Coordinator*() function. They are a portion of the erection activities in the example project shown in Figure 3. These pictures illustrate a scenario in which both cranes erect the structural elements in the overlap area and collisions could occur between them. All inter-crane collisions have been eliminated by applying the computational methods implemented in the *Coordinator*() function. These snaps shots are taken every 10 seconds (i.e. choosing 1 from every 100 frames in the animation file that visualizes actual speed of the erection activities), so the 20 frames in Figure 4 correspond to 200 seconds in the actual erection processes. For a better presentation, the waiting times for securing the element to the hook and the waiting times for releasing the elements are not shown here.

From picture number 01 to 05, both cranes lift the structural elements from material supply locations and rotate their jibs to transport the elements toward their destinations in the building structure. From picture number 06 to 08, both cranes reach the position above their destinations and start lowering the structural elements. Although it seems the tips of two jibs overlap in these pictures, they are actually free from collisions in the three-dimensional space due to different set-ups between the cranes. In construction practice, the heights of the jibs are always set differently (different tower heights) to reduce potential conflictions between jibs. In this example case, the jib of Crane 2 (the crane in further side of the pictures) is higher than the jib of Crane 1 (the near side crane). Therefore, possible inter-cranes collisions will only occur between the cable of Crane 2 and the jib of Crane 1.

At picture number 9, Crane 2 has released the structural element and starts rotating toward its material supply location for preparing the next lift. From picture number 10 to number 16, since Crane 1 is still lowering the structural element, Crane 2 stops rotating to prevent its cable from hitting the jib of Crane 1. From picture number 17, Crane 1 has released the structural element and starts rotating the jib backward its material supply location. At this time, the jib of Crane 1 still blocks the way, to which the jib of Crane 2 supposes to rotate. From picture number 18 to number 20, Crane 1 has left the region that impedes Crane 2 from rotating, so Crane 2 rotates toward it material supply location, and simultaneously Crane 1 continue rotates toward its material supply location for preparing the next lift.

Figure 4 indicates that *Coordinator*() is capable of coordinating the motions of two cranes to complete the erection tasks without any collision. It also shows that using the *Coordinator*() function allows us generating detailed simulations and realistic animations, which provide engineers quantitative and graphical references to visualize and improve erection plans.

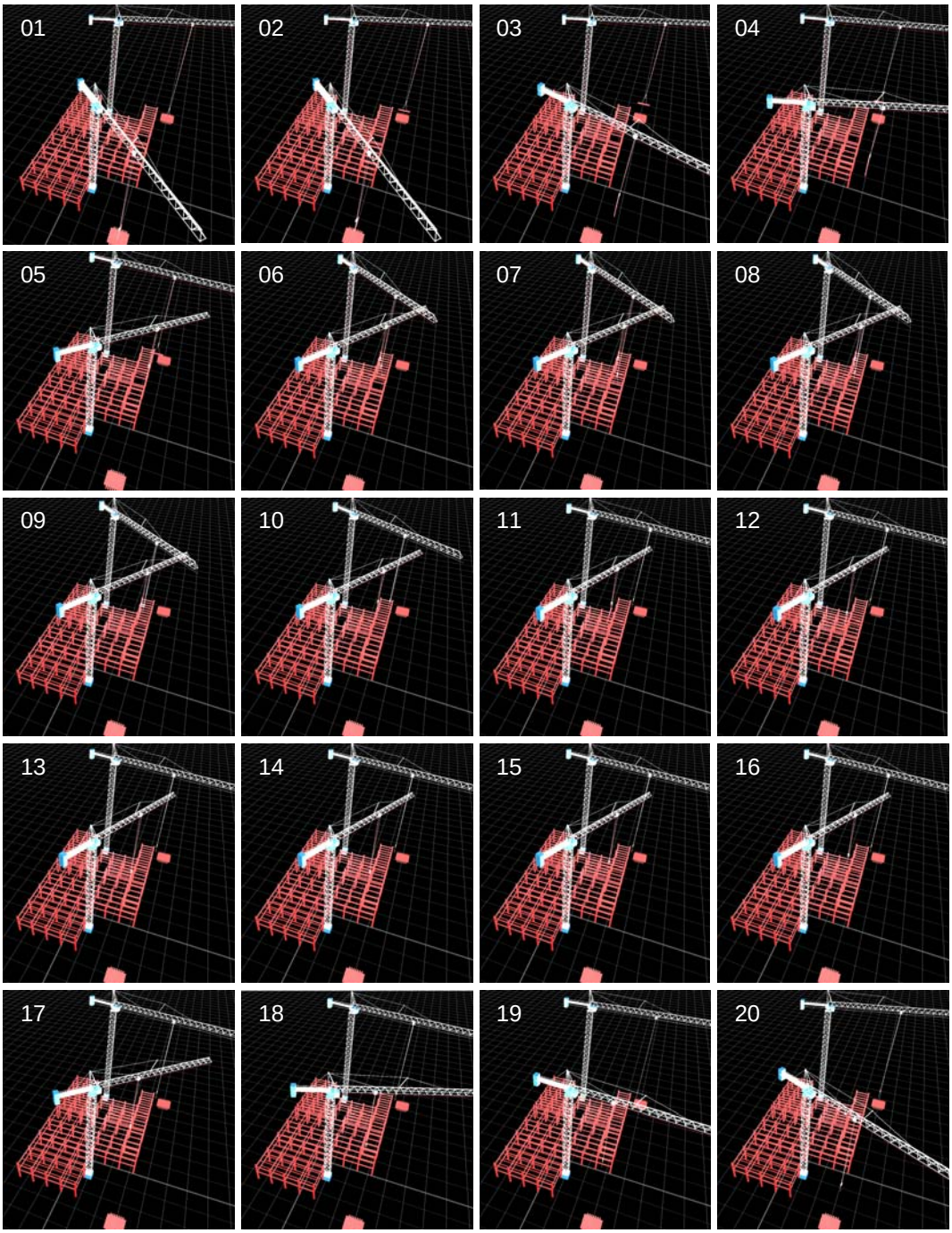

Figure 4: Snapshots of Two-Cranes Collaborating for Erecting a Steel Building.

CONCLUSIONS

A crane-specific coordination method was developed in this research. This paper described a computer algorithm developed in this research. It can be used to coordinate the motions of multiple cranes working simultaneously in a construction site. This algorithm can also preplan the motions of the cranes in computers and actively avoid all possible collisions. A computer program was also developed to demonstrate the use of the coordination method in an example project, in which two cranes are used simultaneously to erect a structure. The result indicates that the developed algorithm can effectively compute the collision-free paths for the cranes and complete assigned tasks safely.

The computer program developed in this research also provides visualizations of crane motions and erection paths in a virtual construction environment. The animations generated by the program can become a useful reference for engineers or project managers using before or during erection processes.

REFERENCES

Alami, R., Robert, F., Ingrand, F. F. and Suzuki, S. (1995). "Multi-robot Cooperation through Incremental Plan-merging," *Proceedings of IEEE International Conference on Robots and Automation*, Nagoya, Japan, May, pp. 2573-2578.

Kang, S. C. and Miranda, E. (2004b). "Automated Simulation of the Erection Activities in Virtual Construction," *Proceedings of Xth International Conference on Computing in Civil and Building Engineering (ICCCBE)*, Weimar, Germany, June.

Kant, K. G. and Zunker, S. W. (1986). "Toward Efficient Trajectory Planning: Path Velocity Decomposition," *International Journal of Robotics Research*, 5(3), pp. 72-89.

Latombe, J. (1991). Robot Motion Planning. Boston: Kluwer Academic Publishers.

LaValle, S. M (1998). "Rapidly-Exploring Random Trees: A New Tool for Path Planning," *TR 98-11,* Computer Science Department, Iowa State University.

LaValle, S. M. (2001). "Motion Strategy Library, Version 1.2.," Computer Science Department, University of Illinois. Retrieved September 20 2003, from http://ms1.cs.uiuc.edu/msl.

O'Donnell, P. A. and Lozano-Perez, T. (1989). "Deadlock-Free and Collision-free Coordination of Two Robot Manipulators," *Proceedings of IEEE International Conference in Robotics and Automation*, Scottsdale, AZ, pp. 484-489.

Sanchez, G. and Latombe, J. C. (2002). "Using a PRM Planner to Compare Centralized and Decoupled Planning for Multi-robot Systems," *Proceedings of IEEE International Conference on Robotics and Automation,* 2, Arlington, VA, pp. 2112-2119.

Sivakumar, P. L., Varghese, K. and Babu, N. R. (2003). "Automated Path Planning of Cooperative Crane Lifts Using Heuristic Search," *Journal of Computing in Civil Engineering*, 17(3), pp. 197-207.

Švestka, P., Overmars, M. H. (1998). "Coordinated Path Planning for Multiple Robots", *Robotics and Autonomous Systems*, 23(4), Cincinnati, Ohio, pp. 125-133.

Warren, C. W. (1990). "Multiple Robot Path Coordination Using Artificial Potential Fields," *Proceedings of IEEE International Conference on Robots and Automation*, pp. 500-505.