OFFLINE PATH PLANNING OF COOPERATIVE MANIPULATORS USING CO-EVOLUTIONARY GENETIC ALGORITHM

by

M.S.Ajmal Deen Ali[1], N. Ramesh Babu[1] and Koshy Varghese[2]

[1]*Dept. of Mechanical Engg., I.I.T. Madras, India, email: nrbabu@iitm.ac.in*
[2]*Dept. of Civil Engg., I.I.T Madras,India, email: Koshy.Varghese@asu.edu*
(Currently on Sabbatical as Visiting Eminent Scholar at Del E. Webb
School of Construction Arizona State University, Tempe, AZ)

ABSTRACT: This paper presents a new path planning approach using Coevolutionary genetic algorithm (CGA) for automating the path planning of two Cooperative construction manipulators. A methodology based on the concept of configuration space (C-space) technique in conjunction with the Coevolutionary genetic search is used for generating the path. The paper proposes a method for finding the minimum distance and collision free path using CGA. It uses an interference detection algorithm that runs in parallel with CGA to check collision between obstacle and cooperative cranes during path planning. The effectiveness of CGA is compared with other search approaches like A* and Genetic Algorithms GA.

KEYWORDS: Cooperative manipulators, Path planning, Coevolutionary genetic algorithm

1. INTRODUCTION

Cooperative manipulators are being widely employed in construction and assembly for handling medium to heavy objects. Path planning of cooperative manipulators is quite different from path planning of a single manipulator. When the object to be lifted is too heavy or large for a single manipulator, options such as specially assembled equipment such as jacking systems and cooperative use of multiple medium capacity manipulators can be a suitable alternative.

Path planning techniques are classified based on the criteria such as dimension of space, mobility of manipulator and obstacles, representation of space and nature of information gathering [1-6]. Trajectory planning of two manipulators, cooperating to manipulate the same object, was solved using genetic algorithm (GA) [7]. Evolutionary algorithm has been used to find time-optimal trajectories of two coordinating manipulators sharing the same work space [8]. Customized genetic operators such as analogous crossover, which suits certain problem situations, was also designed [9]. Recent efforts include the development of natural systems that can capture key features such as Co-evolution and Life Time Fitness Evaluation, for solving path-planning issues [10].

Earlier attempts for path planning include Hill climbing search, A* search and Genetic Algorithm (GA) search [11, 12]. These approaches have certain disadvantages like excessive computational time, memory requirement and non-optimal paths. Recent works utilize the concepts of co-evolutionary genetic algorithm (CGA). Two populations constantly interact and co evolve in case of CGA.

A 2-D path-planning problem using CGA was attempted [10]. This paper presents a new approach to solve the 3-D path-planning problem of cooperative manipulators using CGA.

2. PROBLEM STATEMENT

This section presents the details of a path-planning problem involving two spatial cooperative manipulators, each of 3 DOF with hinged base. Two important considerations of cooperative manipulators path planning are:

(a) *Ensuring cooperation between the two manipulators during lifting.*

(b) *Handling the computational complexity of the problem based on the DOF of the cooperative manipulator system and its movement in obstacle clustered environment.*

a) Ensuring cooperation between the two manipulators during lifting

Cooperation between the two manipulators is ensured by (i) Nature of movement of cooperative manipulators. (ii) Spatial distance between the hook ends of the manipulators during cooperative lift. (iii) Altitude difference between the hook ends of the manipulators and (iv) Hoist limit evaluation for both manipulators.

Nature of movement represents the movement between the two arms, which can be either synchronous or asynchronous. A synchronous movement refers to the identical movement of different joints of cooperative cranes between two steps. In case of asynchronous movement, the movement between the two successive steps will not be the same.

The spatial distance between the boom tips of the manipulators is kept within object length ± 2 units. When the spatial distance between the boom tips of both manipulators is different from the object length, the sloping of load line occurs. This can be computed by (Figure 1).

$$\beta = \tan^{-1} \left[\frac{D}{HL} \right] \qquad (1)$$

where 'HL' - hoisting length of the hook and 'D' - Off-lead i.e. displacement of the hook from boom tip. Let 'W' be the vertical component of the load transferred from the object to the manipulator and 'W_a' the actual load acting along the load line of the crane. 'W_a' is calculated by

$$W_a = W / \cos\beta \qquad (2)$$

$(W - W_a)$ is the additional load transferred to the crane due to the slope 'β' of the load line. This results in additional load transformation to the manipulators.

If the altitude difference (H1-H2), as shown in figure 2, exists, different loads will be acting on the hook ends. This results in more payload acting on a particular manipulator. In order to limit the extra load on the crane, an altitude difference of '3' units is considered.

Hoist limit evaluation represents the safety to be considered in hoisting the rope in either direction i.e. up or down from the ground level. Minimum limit is considered as '0' unit i.e. at ground level and maximum position is boom tip position, considered above ground level, which keeps on varying depending on the luffing angle.

b) Handling the computational complexity of the problem based on the DOF of the cooperative manipulator system and its movement in obstacle clustered environment

Collision of the cooperative manipulator system in the obstacle-clustered environment can occur due to (i) manipulator-1 colliding with obstacles, (ii) manipulator-2 colliding with obstacles, (iii) object colliding with obstacles and (iv) manipulator-1 colliding with manipulator-2.

The feasible movement of the manipulators in the obstacle-clustered environment is ensured by means of interference detection algorithm that assesses the different configurations of the manipulator for collision. The collision checking is done by means of interference checking algorithm that performs interference check between line and plane segments. For this purpose, the obstacle is represented by many plane segments and cooperative manipulator system is represented by many line segments.

A test problem considers two spatial cooperative manipulators performing asynchronous movement. Both manipulators are identical in size and shape. Table 1 shows the configuration of different arms. The upper and lower bound movement for different arms are shown in Table 2.

3. SEARCH METHODOLOGY

This section covers the details of modeling and implementation for 2x3 cooperative manipulator system using A*, GA and CGA.

3.1 A search in open C-space*

A* search is a free search in the open C-space with on-line feasibility check [12]. The main advantage of this search is that it has the ability to go back to any node which was visited earlier. The step size values of swinging, luffing and hoisting for generating neighbors are 5, 5 and 1.

3.2 GA search in open C-space

GA search is a free search in the open C-space with on-line feasibility check [11]. The population consists of 250 strings. Each string in GA represents movements of the manipulator from pick to place location. A typical string with fifteen intermediate configurations between pick and place location is shown in figure 3.

3.3 CGA

In this approach, two populations known as solution population and test population continuously interact with each other to evolve an optimal solution.

a) Solution population: Path representation as a string

The solution population consists of 200 strings. Each string represents the movement of manipulator from pick to place location (Figure 3). The number of steps between pick and place location is taken as fifteen.

b) Test Population

The test population consists of test conditions known a priori (Figure 1 and Figure 2). It consists of (1) spatial distance between the hook ends of the manipulators i.e. object length ± 2 i.e. 15+2 and 15-2, (2) altitude difference between the hook ends of the manipulator i.e. (H1-H2) = '3' units, (3) hoist limit evaluation varies according to boom luffing angle and (4) collision of the cooperative manipulator system with the environment that consists of (i) manipulator-1 colliding with obstacles, (ii) manipulator-2 colliding with obstacles, (iii) object colliding with obstacles and (iv) manipulator-1 colliding with manipulator-2.

c) Encountering

In CGA encountering is the process in which one string from solution population and randomly chosen test conditions from test population interact with each other to produce a better offspring. It consists of three stages.

STAGE 1

Fitness is found for all the individuals in both the populations by subjecting them to encounter '3' randomly selected individuals from the other population. A solution receives a payoff of one if it satisfies the test. Otherwise, it receives a zero. The opposite is true for test. It gets a payoff of one if the solution encountered does not satisfy a test. Each individual - test or solution - has a history, which stores the payoff resulting from such an encounter. The fitness of an individual in solution population is estimated by

$$F_s = P\ (x)\ [1+(1/H_s)] \qquad (3)$$

where P (x) is the objective function and H_s is the total payoff for that individual. The objective function is defined as the sum of square of absolute differences of identical joint angles between successive configurations for all the joints of the manipulators as the cooperative manipulator system moves from pick to place location. It is estimated by

$$P(x) = \sum_{i=1}^{n-1} \sum_{j=1}^{m} | \theta_{i+1, j} - \theta_{i, j} |^2 \qquad (4)$$

where 'n' represents the number of configuration sets and 'm' represents the number of joint parameters required to define a unique position of the cooperative manipulator system and '$\theta_{i, j}$' is the value of joint angle of j^{th} link in i^{th} configuration set. The fitness of an individual in the test population is estimated by

$$F = H_t. \qquad (5)$$

where H_t is the total payoff for that individual. An individual according to their fitness value i.e. minimum distance and payoff for first population and only payoff for second population, is arranged in descending order in their respective population

STAGE 2

In this stage, one fittest string from the first population and three randomly selected test conditions from the test population are subjected to encounter. The selection of this string and test conditions are biased towards highly ranked individuals i.e. the fittest individuals are more likely to be selected. The result of such an encounter is '1' if any test is satisfied (or) '0' in case of violation. The fitness (or) maximum pay-off for this string is calculated. According to its fitness value, it is ranked in descending order in the first population. A similar operation is performed on the test population also. Since both the populations are sorted based on their fitness values, an individual might move up and down in its population as a result of the update of its fitness.

STAGE 3

In this stage, the conventional GA process is followed. Two strings from the first population are selected based on their fitness. An offspring is created by the process of adaptive crossover. Adaptive crossover is illustrated in figure 4, i.e. the same intermediate configurations on both parent 1 and parent 2 will be checked for their payoff value. The particular configurations, with highest payoff, will be inserted in to the same configuration in the offspring. This cycle is repeated for the remaining configurations until all the configurations in the offspring are filled.

Mutation is applied in an adaptive manner with a probability of '0.1'. Adaptive mutation is implemented in order to reduce the angular displacement between adjacent configurations as well as to bring the cooperation between manipulator 1 and manipulator 2. All the joint angle positions are subjected to this probability. If they are satisfied, they will undergo adaptive mutation as shown in Figure 5. For example, the swing position as marked by arrow '1" is to be subjected to adaptive mutation (assumed to be the j^{th} position) then the j-6^{th} (arrow 2) and j+3^{rd} position (arrow 3), both are swing positions, will be considered. A random number will be generated between the swing positions considered. Similarly if luffing position is considered as j^{th} position, then the j-6^{th} and j+3^{rd} positions i.e. luff position, will be considered for mutation. A random number will be generated between these points. The value generated is inserted in the j^{th} position. This process is applied to all the joint angle positions, which satisfy the mutation probability condition.

Figure 6 shows the methodology adopted for finding a feasible collision free string with CGA. The fitness of offspring is estimated as the sum of payoff received from encounter with three-selected test conditions. The offspring is then inserted into the solution population based on their fitness values. To accommodate this new offspring, the lower fitness value string in the solution population is deleted. The procedure adopted in STAGE 2 and STAGE 3 is continued until the fitness of string in the solution population remains almost the same in ten consecutive generations.

4. RESULTS AND DISCUSSIONS

The manipulators shown in the section 2 are considered for cooperative path planning with the proposed CGA approach. In order to asses the effectiveness of this approach, two different approaches such as A* and GA for path planning were also considered and these approaches were developed using C++ programming language and implemented on the same platform, i.e. 333 MHz Pentium II processor PC with 128 MB RAM with Windows NT operating system. The computation time for finding the feasible path from pick to place location using different approaches like A*, GA and CGA, depends on the lift path and position of the pick and place location in the obstacle clustered environment i.e. lifting an object vertically up may be simpler to compute than another path. For path planning of cooperative manipulators, C-space approach was adopted for representing the position of the manipulators. Performance of A*, GA and CGA for path planning of 2x3 cooperative manipulators is assessed and the results are presented in Table 3.

*4.1. A*search*

The minimum incremental movements of different joints of manipulator are shown in Table 4. A* search could determine the feasible path from pick to place location in 30 intermediate steps. The time taken for finding the feasible path is 320 minutes. The minimum distance, in terms of linear movements, from pick to place location is 73 units. If A* is allowed to search the C-space exhaustively i.e. with 1 degree increment, the time taken will be more than 2000 minutes. The disadvantage with this step angle increment is that pick and place angle has to lie within the increment angle of search; otherwise the search will never give a feasible solution.

4.2. GA search

Total number of joint angles to represent a unique configuration is 6; Range for different joint angles swing: 0-360 degrees: luffing: 30-80 degrees: hoist: 0-39 units; Number of configuration sets representing a string excluding pick and place location: 15.

For generating the path with GA, the number of intermediate configurations between pick and place location is fixed as fifteen. The time for computation of feasible path with GA is 183 minutes. The minimum distance in terms of linear movements from pick to place location is 79 units. GA computes the collision for all the configurations in the population, due to which a considerable time was spent.

4.3. CGA search

Total number of joint angles to represent a unique configuration: 6; the arm configurations and its lower and upper bound values are shown in Table 3 and Table 4. Number of configuration sets representing a string excluding pick and place location: 15.

The computation time for finding a feasible path is 20 minutes. The minimum distance, in terms of linear movements, from pick to place location is 58 units. CGA subjects only two strings i.e. one fittest string and one offspring for collision computation in the successive generations except the initial fitness ranking generation, thus saving a large amount of time.

The pictorial view showing the object position at intermediate locations for A*, GA and CGA are shown in figure 7, figure 8 and figure 9. These discrete object positions are drawn by converting the intermediate configuration angles of the cooperative manipulator system in C-space to the Cartesian Space.

From the figure 7, it is observed that the discrete object positions are placed in a zigzag position in A*. This is due to the move taken by the search for adjacent feasible configuration when it encounters one. In the Figure 8, it is observed that discrete object positions computed with GA are not located at equal intervals. Since the entire population undergoes cross over and mutation at the same time, the possibility of a larger random joint angle movement exists. In the Figure 9, it is observed that discrete object positions computed with CGA are almost located at equal intervals. Two populations i.e. highly biased individuals from string population undergo adaptive cross over and parameter based mutation, which results in the possibility of smooth movement of joints with equal interval of displacement.

5. CONCLUSIONS

This paper presents a new approach using CGA for automated path planning of 2x3 cooperative manipulator system. The suitability of CGA for offline path planning in comparison with other techniques was demonstrated. From the results presented in this paper, the following conclusions can be made:

1. CGA in conjunction with C-space technique proves to be an effective approach to solve path planning problems of cooperative construction manipulators in complex environment

2. Search using CGA is found to be effective approach in terms of computation time, when compared to other search techniques like A* search and GA search.

3. CGA was found to be efficient in generating the shortest path from pick to place location, when compared to other search techniques like A* and GA.

The future work attempts to investigate the suitability of this approach for more complex cooperative manipulator applications like 2x4 i.e. two manipulators each with 4 DOF.

ACKNOWLEDGEMENTS

The authors wish to acknowledge the Department of Science and Technology (DST), New Delhi, India, for the financial support given to this research work vide Grant No. III. 5(23)/2000-ET.

6. REFERENCES

[1] Latombe, J.C. (1991) *Robot Motion Planning*, Kluwer Academic Publishers, Norwell, Massachusetts.

[2] Schwartz, J.T. and Sharir, M. (1988) *A Survey of Motion Planning and Related Geometric Algorithms*, Artificial Intelligence, 37, 157-169.

[3] Udupa, S. (1977) *Collision Detection and Avoidance in Computer Controlled Manipulators*, Ph.D. Dissertation, Department of Electrical Engineering, California Institute of Technology, California.

[4] Lozano-Perez, T. (1983) *Spatial Planning: A Configuration Space Approach*, IEEE Transactions on Computers, C-32(2), 108-120.

[5] Khatib, O. (1986). "*Real-Time Obstacle Avoidance for Manipulators and Mobile Robots*," International journal of Robotics Research, Vol. 5, no. 1, pp. 90-98

[6] Schilling, R. J. (1990). *Fundamentals of Robotics: Analysis and control*, Prentice-Hall, Englewood Cliffs, New Jersey

[7] Sun, S., Morris, A.S. and Zalzala, A.M.S. (1996) *Trajectory Planning of Multiple Coordinating Robots using Genetic Algorithms*, Robotica, 14, 227-234.

[8] Rana, A.S. and Zalzala, A.M.S. (1996) *Near Time Optimal Collision Free Motion Planning of Robotic Manipulators using an Evolutionary Algorithm*, Robotica, 14, 621-632.

[9] Davidor, Y. (1991) *Genetic Algorithms and Robotics: A Heuristic Study for Optimization,* World Scientific Publishing Company.

[10] Paredis, J., Westra, R., (1997) *Co-evolutionary Computation for path planning,* Proceedings 5th European Congress on Intelligent and Soft Computing (EUFIT 97), H.-J Zimmermann(ed.), Verlag Mainz, Aachen.

[11] Sivakumar, PL., Varghese, K. and Ramesh Babu, N. (1999) *Path Planning of Construction Manipulators using Genetic Algorithms*, Proceedings of the 16th International Symposium on Automation and Robotics in Construction, Madrid, Spain, 555-560.

[12] Sivakumar, PL., (2001) *Automated Path Planning of cooperative crane lifts using Genetic and heuristic searches*, M.S. Thesis, Department of civil engineering, Indian institute of technology Madras.

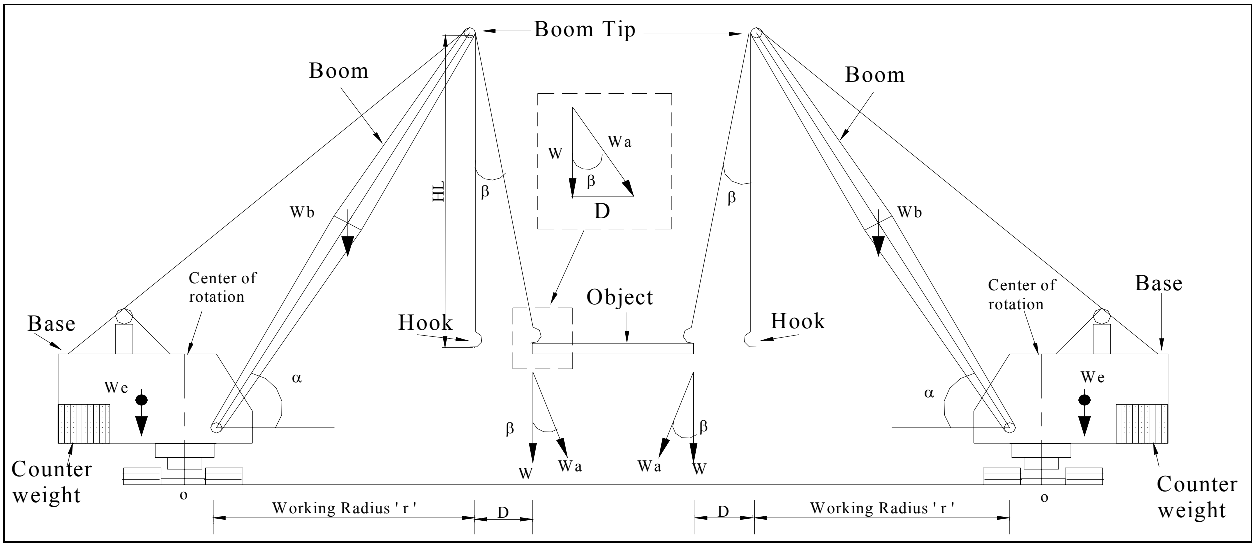

Figure 1 Increase in Load Transferred to the manipulators due to slope of the load line

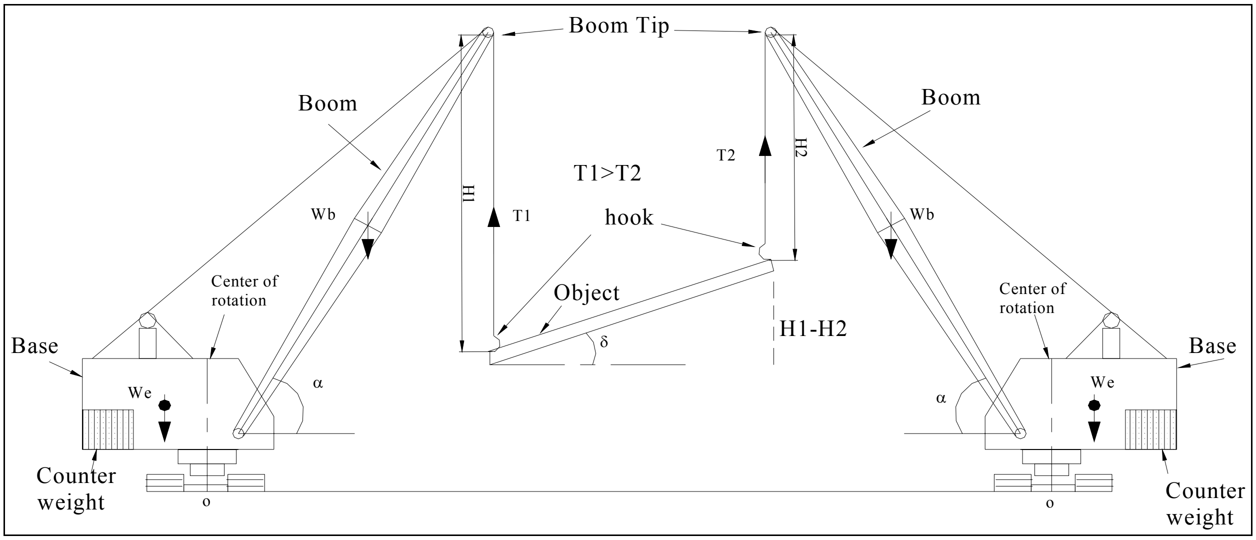

Figure 2 Difference in load shared by hook ends of the manipulators

Table 1. Manipulator arm configurations

Arm configurations	Length	Breadth	Height
Base	10	8	2.5
Boom	40	3	3
Hoisting Rope	39 unit	-	-

Table 2. Upper and lower bounds for the manipulator arms

Arm	Base	Boom	Hoisting Rope
Lower	0^0	30^0	0 unit
Upper	360^0	80^0	39 unit

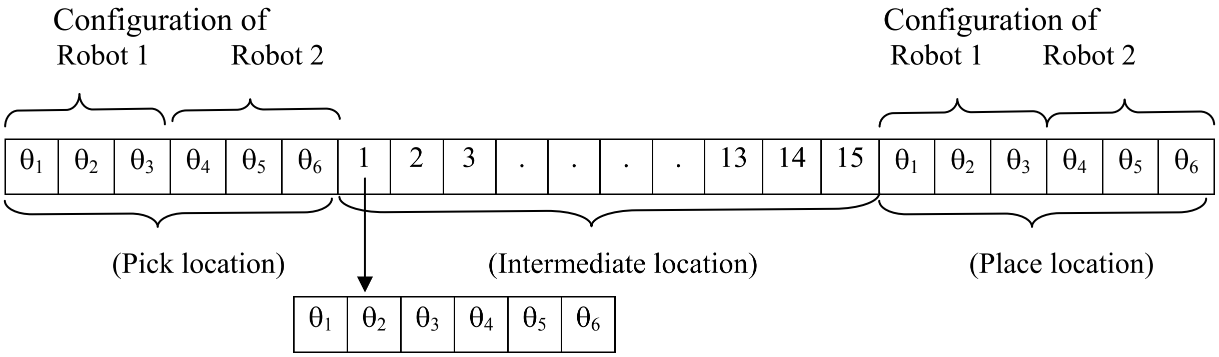

Figure 3.A typical string with fifteen intermediate configurations of 2x3 cooperative manipulators between pick and place location

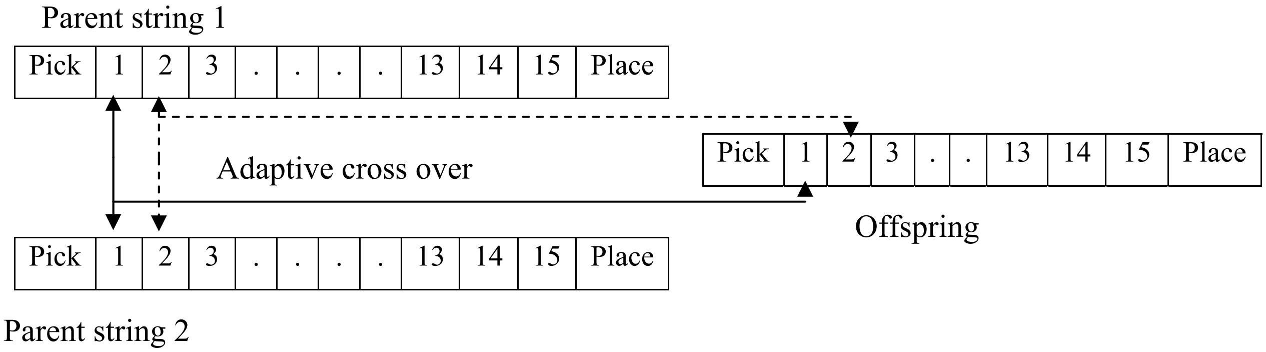

Figure 4. Adaptive cross over

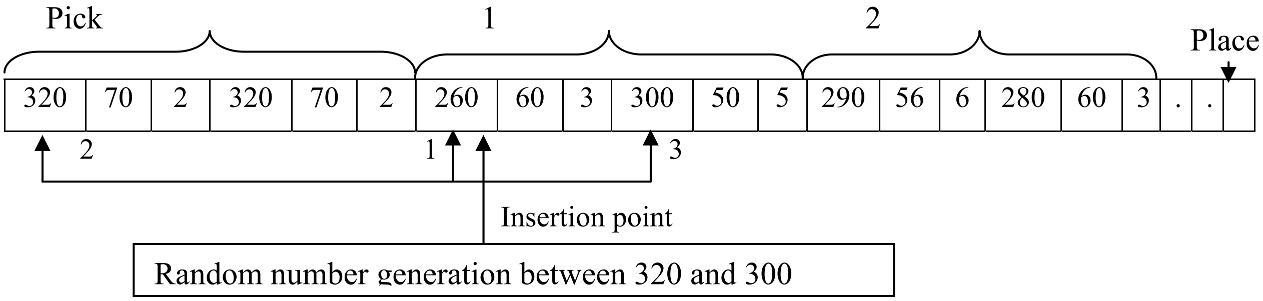

Figure 5. Adaptive mutation

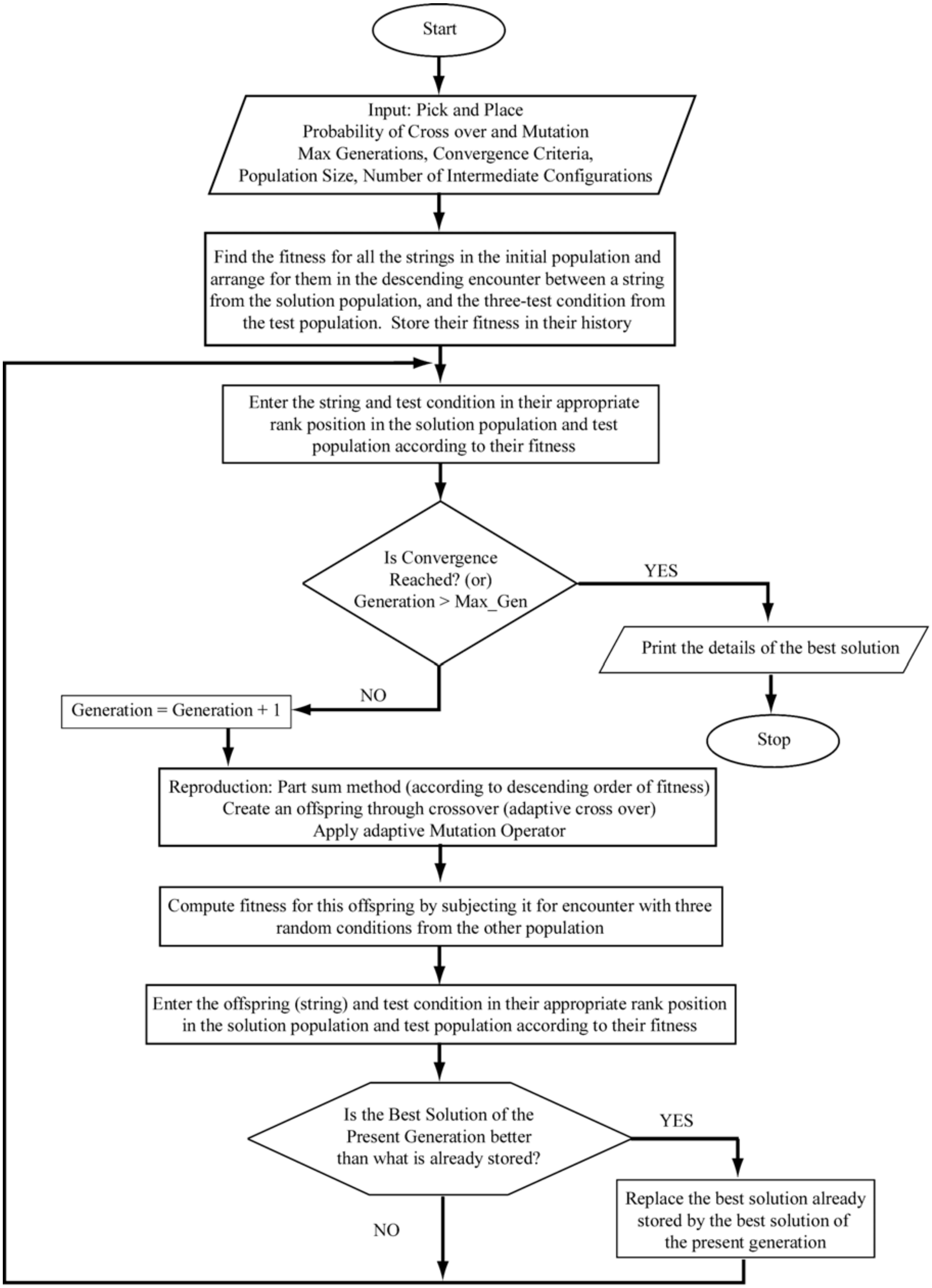
Start
Input: Pick and Place
Probability of Cross over and Mutation
Max Generations, Convergence Criteria,
Population Size, Number of Intermediate Configurations
Find the fitness for all the strings in the initial population and arrange for them in the descending encounter between a string from the solution population, and the three-test condition from the test population. Store their fitness in their history
Enter the string and test condition in their appropriate rank position in the solution population and test population according to their fitness
Is Convergence Reached? (or) Generation > Max_Gen
YES
Print the details of the best solution
Stop
NO
Generation = Generation + 1
Reproduction: Part sum method (according to descending order of fitness)
Create an offspring through crossover (adaptive cross over)
Apply adaptive Mutation Operator
Compute fitness for this offspring by subjecting it for encounter with three random conditions from the other population
Enter the offspring (string) and test condition in their appropriate rank position in the solution population and test population according to their fitness
Is the Best Solution of the Present Generation better than what is already stored?
YES
Replace the best solution already stored by the best solution of the present generation
NO

Figure 6: CGA methodology for finding feasible collision-free string

Table 3. Performance comparison of A*, GA and CGA for 2x3 cooperative manipulator path planning

Method	Earlier approach [11,12]		Current approach
	A*	GA	CGA
Pick location	[320,70,2][320,70,2]	[320,70,2][320,70,2]	[320,70,2][320,70,2]
Place location	[270,40,2][270,40,2]	[270,40,2][270,40,2]	[270,40,2][270,40,2]
Number of generation	-	540	750
CPU time (minutes)	320	183	20
Distance in terms of linear movements (units)	73	79	58

Table 4. Incremental angle between adjacent moves of manipulator

Arm	Base	Boom	Hoisting Rope
Lower	0^{0}	30^{0}	0 unit
Upper	360^{0}	80^{0}	39 unit
Increment	5^{0}	5^{0}	1 unit

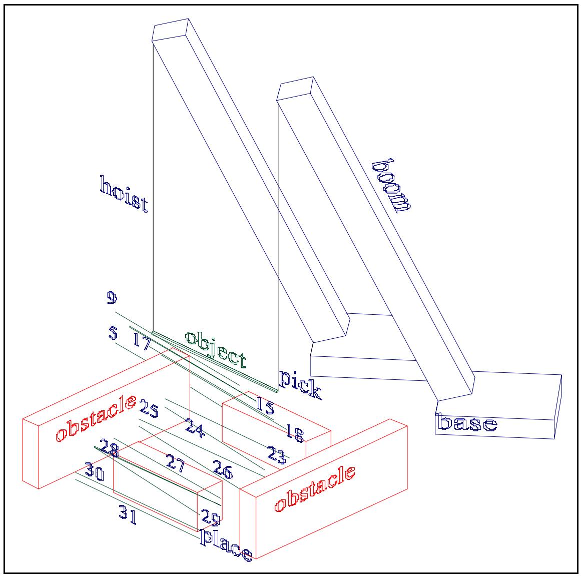

Figure 7. Pictorial view showing the path generated by A* for cooperative manipulators

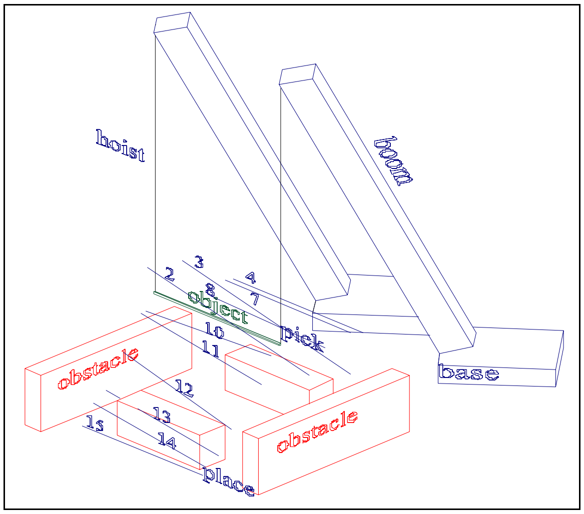

Figure 8. Pictorial view showing the path generated by GA for cooperative manipulators

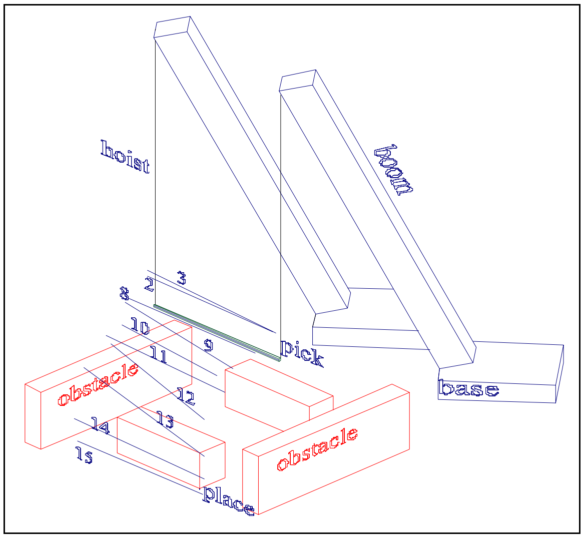

Figure 9. Pictorial view showing the path generated by CGA for cooperative manipulator