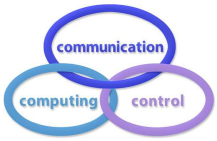
INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
Online ISSN 1841-9844, ISSN-L 1841-9836, Volume: 19, Issue: 3, Month: June, Year: 2024
Article Number: 6379, https://doi.org/10.15837/ijccc.2024.3.6379




CCC Publications


# Quicker Path planning of a collaborative dual-arm robot using Modified BP-RRT* algorithm

Josin Hippolitus A, Senthilnathan R

**Josin Hippolitus A***
Department of Mechatronics Engineering
SRM Institute of Science and Technology
Kattankulathur, Chennai, India
Corresponding author: josinhippolitus@gmail.com

**Senthilnathan R**
Department of Mechatronics Engineering
SRM Institute of Science and Technology
Kattankulathur, Chennai, India
senthilr4@srmist.edu.in

**Abstract**

Path-planning of an industrial robot is an important task to reduce the overall operation time. In industrial tasks, path planning is executed with lead-through programming, where in most cases the robot faces singulated object configurations. Cluttered environments demand path-planning algorithms, which are sensor driven, rather than pre-programmed. Path-planning algorithms, like RRT, and RRT* and their variants have inherent problems like the duration of a search and the creation of several node samples which may lead to longer path lengths. Back Propagation-Rapidly exploring Random Tree* (BP-RRT*) algorithm was a leap forward when an obstacle is enveloped with a sphere. Due to the spherical envelope of the obstacle, this method evaluates the connection between the path and obstacle in space with a spherical envelope using the triangular function and identifies the non-collision path in 3D space. This predicts the best non-collision path in the 3D workspace. The current state-of-the-art of BP-RRT* is limited to single-arm robots. A collaborative dual-arm robot faces more problems in path planning than a single-arm robot like inter-collision of manipulator arms apart from avoiding obstacles. A Modified BP-RRT* algorithm is proposed for the dual-arm collaborative robot has a pre-stage partition of grids that makes the computation faster, efficient, and collision-free compared to the traditional path planning algorithms namely RRT, RRT*, Improved RRT* and BP-RRT*. The algorithm is implemented in simulation as well as in physical implementation for ABB YuMi dual-arm collaborative robot and the typical length of the path of the proposed modified BP-RRT* method has reduced by 53.8% from the traditional RRT method, 6.95% from the RRT* method, 7.77% from improved RRT* method and 6.83% from the BP-RRT* method. Also, the average time to grasp has reduced by 17.84%, the typical duration for search has decreased by 33.45%, the number of node samples created has reduced by 14.79% from BP-RRT* algorithm.

**Keywords:** Path planning, Dual-arm collaborative robot, Modified BP-RRT* algorithm.

# 1 Introduction

## 1.1 Background

In this ever-evolving era of science, the application of manipulator robotics is growing past the known environments and known objects and their geometry. In singulated object configurations, only a single object is placed in the scene where path planning is limited to calculating the inverse kinematics of the manipulator. Robots developed for a singulated object configuration come with the disadvantage of maintaining the workspace clutter-free [7]. But in real-time use cases, the workspace may have more than one object to handle as well as obstacles to avoid. The user would differentiate the object of interest as well as objects of disinterest. The objects of disinterest are termed obstacles. Based on the geometry of the objects of interest [3], they can be grasped by a single arm or both arms. When both arms are engaged in collaborative grasping, there is a chance of slow search as well as grasp. Each arm acts as two different robots hence, resulting in independent computation and execution without hitting any obstacle and changing the configuration. This is where path-planning algorithms find their significance.

## 1.2 Problem description

Single-arm robots are the common robots of interest in path-planning for any object-handling scenario. The main reason behind this is that those researchers focused on geometrically smaller objects. Any larger objects which cannot be handled by a single-arm may result in an unsuccessful grasp [28]. That's when dual-arm grasping finds its significance. Also, in path-planning many researchers focused on singulated object configurations [27], where only one object is present in the act without any obstacle. With no obstacles, there is no need for an obstacle search. Hence, the path planning of the manipulator is computationally inexpensive and requires a comparatively modular algorithm to calculate the inverse kinematics of the manipulator in action. However, a structured-clutter environment will have challenges in searching the obstacle, avoiding the obstacles and grasping the target object all in a shorter period. Hence, advanced algorithms are used for path planning to avoid hitting the obstacles for two main reasons: a) One is to avoid creating unstructured cluttered environment due to the disturbance of obstacles, and b) To avoid the robot from stopping. Because, if the arm of the collaborative robot hits any obstacle, the arm would stop moving as a safety strategy.

In a dual-arm collaborative robot, the reason for slow search may be due to the obstacle present in the extended workspace and inefficient planning in the workspace. The most peculiar challenge in a dual-arm collaborative robot is its shared workspace. The left arm and right arm share some part of the workspace. This may lead to accidental collision and the robot may stop its operation. The mapped workspace of an abb YuMi collaborative robot is shown in Fig 1 (a). The abb YuMi has a 7-axis configuration, reachability of 0.559 m and a position repeatability of 0.02 mm[26]. That's why a powerful algorithm finds its significance in the path planning of a dual-arm collaborative robot.

The novel modified BP-RRT* algorithm performs an efficient dual-arm path planning. The algorithm leverages the ambidextrous part-handling ability of the robot to exhibit superiority in path planning in a structured clutter with obstacles of various sizes and shapes.

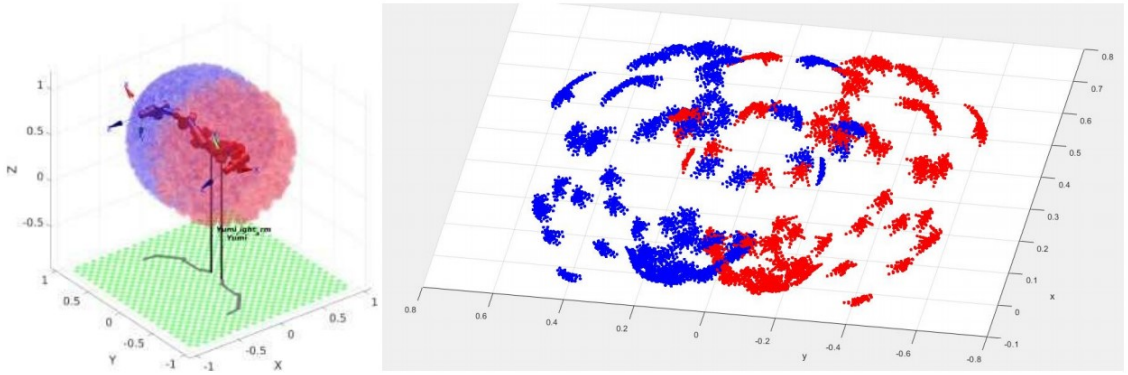
Consider a workspace W, where W$\subseteq \mathbb{R}^n$ with a dimension n, where $n \in \mathbb{N}$ and n$\geq$2. The obstacle region in the workspace is $W_{obs} \subset$W and the free region where the manipulator has the liberty to move freely without hitting any obstacle is $W_{free}$, where $W_{free} = W_{obs}$, $W_{free}$ has the set of all state points in the workspace.

W1 is the workspace of the left arm and W2 is the workspace of the right arm. W1 intersection W2 is the critical area of shared workspace where inter-arm collision might happen.

The major advantage of a dual-arm collaborative robot is its extended workspace and its higher manipulability index shown in figure 1 (a) and (b). If a single-arm robot is used, the workspace used is either W1 or W2. Whereas, when we use a dual-arm robot, the workspace would be W1+W2.

1. When the target object is being placed in $W1 \in W2$ and $W2 \in W1$, and the mass of the target object is within limits, then the respective arm L or R can be used.

2. Both arms are used under two conditions:



(a) (b)

Figure 1: (a) Workspace of the simulated YuMi robot (Hippolitus [26]), (b) Zones for higher manipulability index.

(a) When the payload of the target object is over the limit

(b) When the geometry of the object may lead to an unsuccessful grasp

3. If both arms have to be employed based on the geometry and the mass of the object in the shared workspace (W1 ∩ W2), the goal location should also be planned in the shared workspace area (W1 ∩ W2).

Hence, efficient planning in the workspace with a novel algorithm can improve the speed of search and grasp in a collaborative dual-arm robot.

### 1.3 Related works and their analysis

Traditionally path-planning methods are categorized into two: Search-based [24] and sampling-based [5] planning methods.

Algorithms like A*[6][10][17] and Dijkstras [11][20][25] comes under search-based path planning. The major disadvantages of search-based path planning are that they are computationally very expensive and hence their operation process time is very high. Since both of these disadvantages can reduce the efficiency of the system, sampling-based path-planning methods [13] are opted for.

RRT (Rapidly-expanding Random Trees) [14] is a common algorithm based on sampling-based path planning. RRT also has a few disadvantages they are very random, very slow in search and unoptimized path planning. There are multiple adaptations of the RRT algorithm that try to rectify the disadvantages of the RRT algorithm one after another ultimately to improve the convergence rate and search potential.

The high randomness of the final paths created by the RRT algorithm [14] has been solved by the RRT* algorithm [12]. But, RRT* creates a new challenge of a long operation time, which is again solved using MOD-RRT*[19]. The slow search speed nature of the RRT algorithm is rectified using an Neural Rapidly exploring Random Tree* (NRRT*) algorithm [22].

Even though NRRT*[22] addresses the challenge of the speed of search being very slack, they are fundamentally for 2D workspaces. But for a dual-arm robot manipulator path planning, the workspace is a 3D environment. This research challenge is being addressed in this paper.

BP-RRT*[8] addressed the problem of the high typical duration of a search, a greater number of node samples created, and longer path length. BP-RRT* overcame all three challenges with a minimum typical duration of a search, a lesser average amount of sample nodes created and a shorter path length. They used a stage search in a multi-obstacle search space for a single-arm robot. Also, the algorithm was implemented in a ROS-Moveit as well as a Baxter robot through ROS.

### 1.4 Contribution of this paper

This research seeks to overcome the challenges of high search time, a greater number of node samples created and a longer path length using a sampling-based algorithm. The BP-RRT* algorithm [8] has achieved a massive feat in reducing the high search time offered by RRT and a greater number of node samples created in the RRT* algorithm. It also managed to reduce the path length significantly. However, the BP-RRT* algorithm focused only on a single-arm robot manipulator path planning. When a path is planned for a dual-arm collaborative robot, the planning has its challenges to face. The two arms of the robot act as individual robots. So, the path planning also takes a considerable amount of time.

When we implement two independent BP-RRT* algorithm for each arm, the shared workspace is not considered, which provides a massive opportunity as well as challenge for this research. Extended workspace is the opportunity and optimizing the path planning with faster computation is the challenge. The extended workspace for the dual-arm robot and zones of higher manipulability index are shown in figure 1.

The Modified BP-RRT* algorithm divides the workspace into 54 grids [21] before classifying these grids into regions. This would ease the computation for faster processing. These grids are then sampled for discretization of obstacles and to determine the sampling probability for reducing unnecessary sampling nodes. In the end, simulation experiments and physical experimental analysis with an ABB YuMi dual-arm robot are performed to show the efficiency and robustness of the algorithm.

This work focuses on improving the overall efficiency of path planning in a static environment by planning a path smartly to process and execute an optimized path in a shorter period.

## 2 BP RRT*

BP-RRT* algorithm is based on a backpropagation neural network model and RRT* algorithm. This algorithm works in any multi-obstacle space, where the obstacles are of different geometry, the sampling space is split to determine the sampling probability of nodes depending upon the density of the obstacles. So that staged local search can be performed. There are four parameters which would help in making the algorithm agile for various single arm path planning scenarios which are sampling space, number of obstacles and their size along with step size. In the BP-RRT* algorithm, the workspace is split into 27 sampling grids. Due to its agile nature, the algorithm outperformed other general search algorithms like RRT, RRT*, P-RRT, and N-RRT. However, the major challenge is when it is implemented in a dual-arm collaborative robot due to reasons mentioned in the previous sections.

### 2.1 RRT* algorithm

The root node is indicated by the starting node, $X_{init}$, in the conventional RRT algorithm [14]. Using the Euclidean distance concerning primary node $X_{rand}$, which is created by the random function in the free zone, selects the nearest node $X_{near}$ from the random tree. The current iteration ends and the next one begins if there is an obstacle on the path between $X_{near}$ and $X_{rand}$. To create a new node called $X_{new}$, the fixed Step Size is expanded by a function new_state in the line direction connecting $X_{near}$ with $X_{rand}$. Fig. 1 shows this process of node expansion.

There are two ways that the RRT*[12] method differs from the RRT [14] algorithm. The ideal parent of an $X_{new}$ isn't always $X_{near}$ once it's been formed. Once the best parent node for the $X_{new}$ has been identified, we need to figure out if it will be less expensive to go from the $X_{new}$ to the $X_{init}$ and then from the $X_{new}$ to the locations surrounding the $X_{new}$. figure 2 shows the overall optimization procedure.

### 2.2 Probability-based Improved RRT*

BP-RRT*[8] is proposed as a solution to the sluggish pathfinding, low efficiency in 3D space, and poor node sampling under multi-obstacle situations concerns of the classic RRT* algorithm.

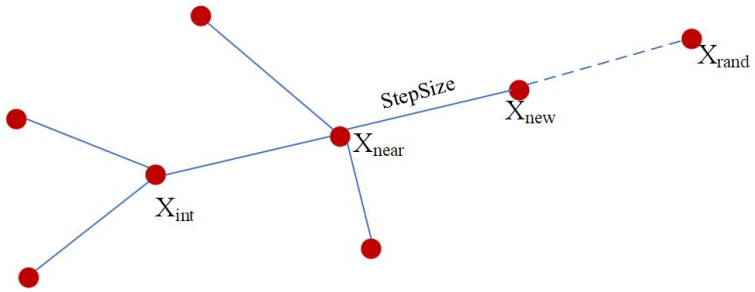



Figure 2: RRT algorithm extension diagram

First, 3D spatial obstacle collisions are detected by the algorithm. Subsequently, the node sampling probability computation is carried out to minimize superfluous sample nodes. To transform the global search into stage local search, stage local search sampling is finally carried out. This trains the BP neural network model, forecasts the number of nodes sampled in each stage of local search, directs the algorithm to the following stage so that the search is automatically completed, and increases pathfinding efficiency.

### 2.2.1 Sensing colliding obstacles

The first task in any collision sensing research is to differentiate the objects of interest from the so-called obstacles which are of no interest to grasp. This research focuses on creating an obstacle ball envelope to identify the obstacle and further use it in the algorithm to identify the best non-collision path in the workspace.

**Obstacle ball envelope** Identifying the obstacles and mapping them is a challenging exercise in a workspace especially when the obstacles are placed very close to each other. This follows the very conventional way of hand-picking any unimportant object in any given environment, which is the workspace in this case, wrapping the obstacle with a ball structure and identifying the centre of that ball structure O1(x2,y2,z2), which is shown in figure 3. The step involves finding the Euclidean distance from the sampling node P1(x1,y1,z1) to the centre of the ball structure in the given workspace W and is given by the expression equation 1.

$$\rho = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \tag{1}$$

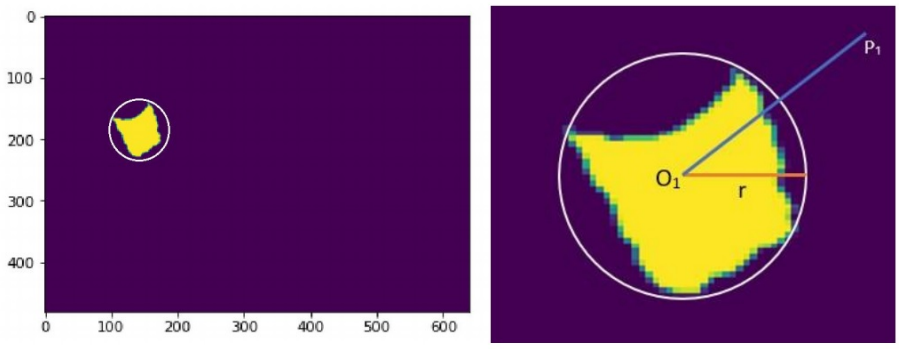



Figure 3: Obstacle sphere envelope with sampling node P1 and Obstacle envelope centre O1 and radius r

**Determination of Non-collision paths in 3D space** Now that the proximity of sampled nodes with the obstacles is found, non-collision paths can be developed in the workspace with the condition that the distance between the sampled nodes' P1 and the centroid of the ball O1 is always greater than r, the radius of the ball structure, which is enveloping the obstacle. This is shown in the figure

4. Trigonometric functions to determine if the created path $\overrightarrow{P_1P_2}$ coincides with the enveloping sphere are used to compute the affinity of the path from the obstacle.

### 2.2.2 Probabilistic node sampling

As the collision-free path is identified, a few more challenges arise which are seen in figure 4. They are namely: a) Unnecessary sampling nodes, b) inefficient sampling, and c) high calculation time. To overcome these challenges, a two-step process is adopted as seen below. The outcome of this process finds the sampling probability as they divide the sampling workspace, and discretize distance-weight function as well as obstacles.

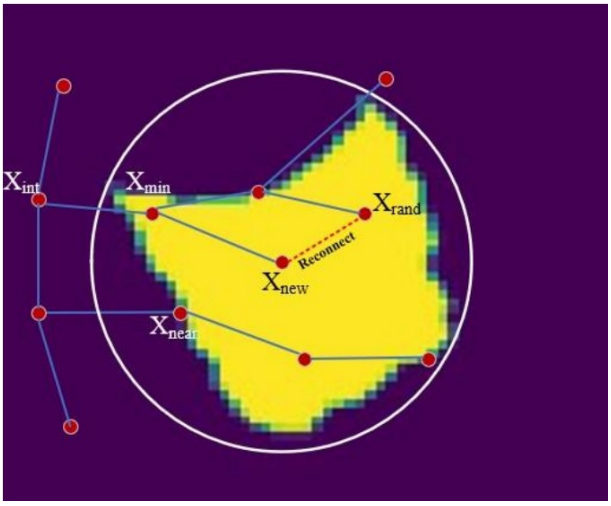



Figure 4: Depiction of RRT* algorithm with new parent node and Reconnect (Chen [4])

**Calculation of free volume** The sampling nodes created by RRT* are very much random and the required sampling nodes to derive the final non-collision path are also very high. Also, RRT algorithms are applied in 2D spaces, but a robotic workspace is a three-dimensional workspace, which is shown in figure 5. Hence, they become computationally expensive altogether to implement without any strategy or improvisation. Dividing the workspace into samples as sampling spaces is a good strategy to organize the workspace for any search-based strategy.

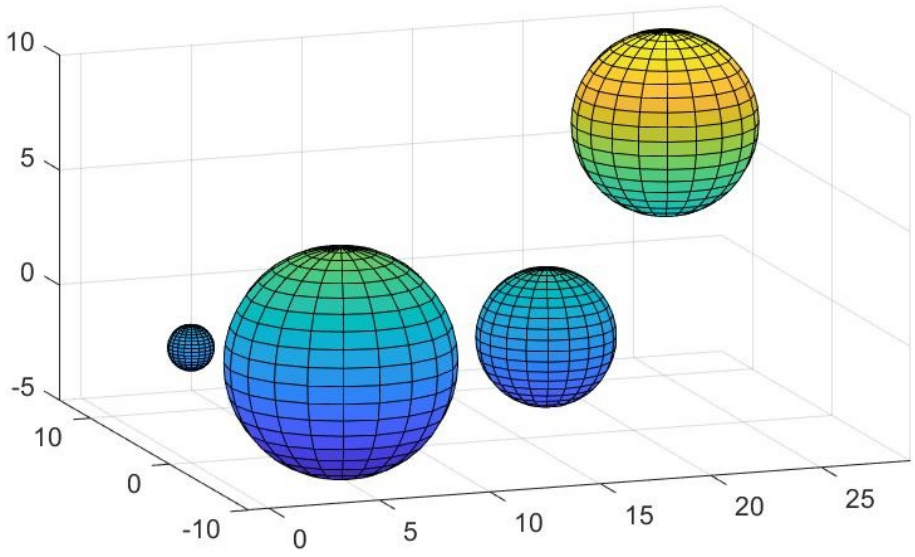



Figure 5: Simulation of obstacle in the configured workspace

In Raster method [9], a workspace is divided into 27 small sampling spaces (3*3*3). There are three vertical layers and each layer has 9 sub-layers called divisions, which totals 27 divisions. The starting location is 1 and the target location is 27. The obstacle can be present in multiple divisions. Once the obstacle is placed rightly this way, a ConvexHull algorithm [2] function helps in finding the volume of the obstacle [1].

Once the obstacles are discretized as shown in figure 6, the free volume through which the manipulator can move around can be determined using the expression below.

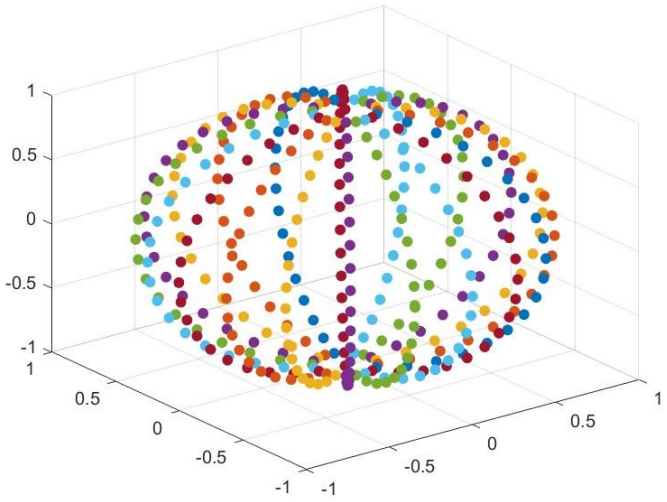



Figure 6: Discretization of Obstacle sphere

$$V_{free}^{i} = V_{x}^{i} - V_{obs}^{i} \tag{2}$$

Where, $V_{free}^{i}$ = Free volume $V_{obs}^{i}$= Volume of obstacle $V_{x}^{i}$= Volume of divided search space

**Calculation of distance weight** The distance-weight coefficient di is determined based on the distance from the divided target [8] location and is expressed in equation 2 as a normal distribution.

$$d_i = \frac{1}{4\sqrt{2\pi}} \times e^{-(\frac{n_i^2}{32})} \tag{3}$$

As we have determined the distance weight coefficient using equation 3 and free volume , deriving the node sampling probability weight coefficient $k_i$ has become an easy task. As $k_i$ is the distance weight coefficient times the free volume .

**Calculation of node sampling probability weight coefficient** The node sampling weight coefficient is distance weight times the free volume, they can be determined by equation 4

$$k_i = V_{free}^{i} \times d_i \tag{4}$$

**Calculation of node sampling probabilities** The configuration space is randomly set with obstacles in a spherical envelope. The radius and centroids of the spherical obstacle envelopes are tabulated in Table 1.

The node-sampling probability function[8] for every grid can be calculated by the equation 5

$$S_i = \frac{k_i}{\sum_{i=1}^{18} k_i} \quad i = 1, 2, 3, ...18 \tag{5}$$

Table 1: Obstacle geometry and location

| Obstacle envelope | Centroid (in cm) | Radius (in cm) |
|---|---|---|
| Obstacle envelope 1 | (0,0,0) | 1 |
| Obstacle envelope 2 | (5,-5,0) | 5 |
| Obstacle envelope 3 | (15,-3,0) | 3 |
| Obstacle envelope 4 | (25,10,6) | 4 |

### 2.2.3 Local search based on NN

**Search sampling** As per the location of the obstacle, the collective workspace is split into three regions, left arm region, right arm region and shared workspace region. They are further split into three stages: stage 1, stage 2 and stage 3. Once the obstacle region is identified, sampling will be performed from stage 1 to stage 2 to stage 3. The proposed Modified BP-RRT* algorithm will provide only 18 grids for the local search to sample compared to 27 grids provided by the BP-RRT* algorithm. This reduction in the local search space[18] will significantly reduce the typical duration of a search as well as the number of node samples created along with a much shorter path length.

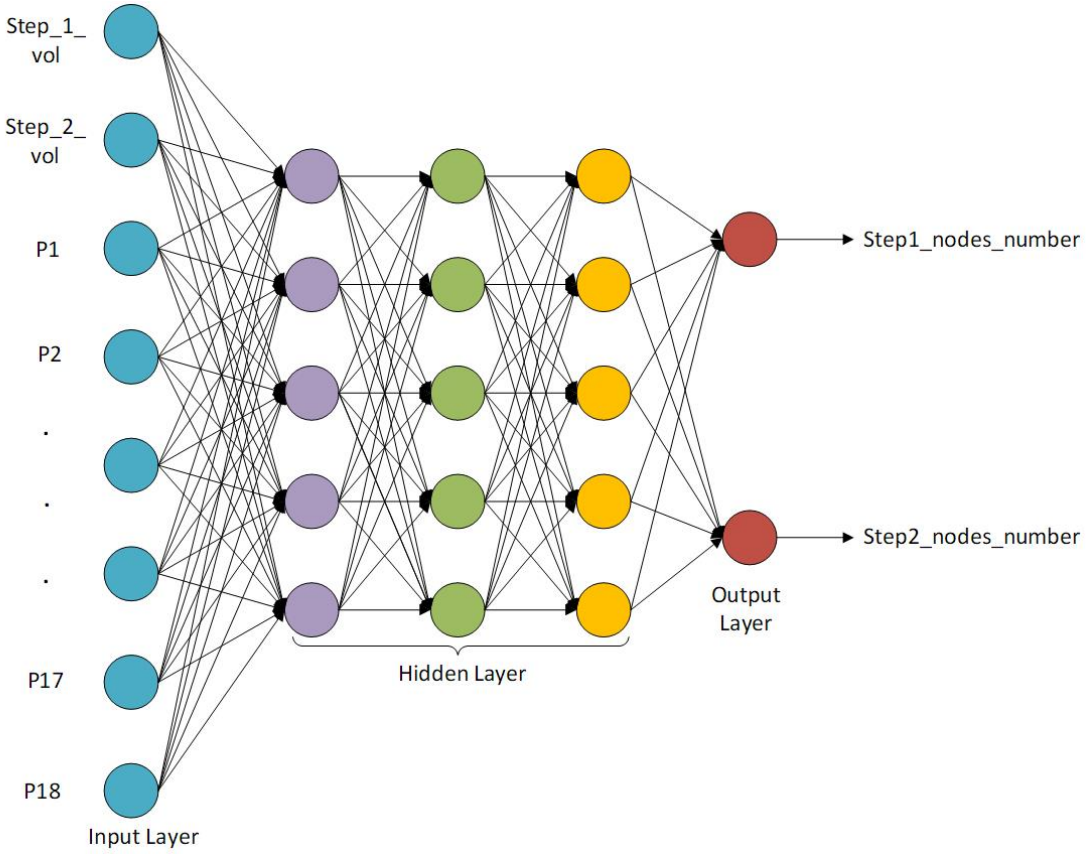



Figure 7: BP neural network structure diagram

**Sampling probability** The probability of node sampling for a search space can be calculated by the equation 5. Based on the divided sample space, the sampling probability for stage 2 can be determined from equation 6.

$$S^i_{stage2} = \frac{S_i}{\sum_i S_i} if x^i \quad instage2 \tag{6}$$

Upon satisfaction of the given condition, the process will gear up to stage 3. In stage 3 the sampling probability can be determined by equation 7.

$$S^i_{stage3} = \frac{S_i}{\sum_i S_i} if x^i \quad instage3 \tag{7}$$

**Backpropagation-based stage-wise prediction of the number of sampling nodes** In a random map, the total number of sampling nodes has to be calculated in each stage, namely stage1 and stage2. In a practical system, the need for arriving at total sampling numbers is critical. An algorithm based on ANN as shown in figure 7, helped in getting to that number quickly and precisely using backpropagation[15] so that the number of samples generated could be less effective than RRT* along with a shorter path length. This is done in a step-by-step manner as follows:

(i) Generating a dataset:

a. Create a random map

b. Run it through the probability-based RRT* algorithm

c. Input:

a) The volume of the space in the regions of Stage1 as well as Stage2,

b) Sampling probability of nodes in each space after division, $S_i$

d. Output:

a) Stage1_nodes_number, and

b) Stage2_nodes_number.

(ii) Back Propagation NN:

a. 20 input parameters

b. 20 inner layer dimensions

c. 2 output parameters

d. 2 output layer dimensions

e. The total number of layers including inner and outer layers is 5

(iii) Loss function:

Loss function is a way to assess how effectively your algorithm represents the data in your dataset. A bigger value will be produced by your loss function if all of your forecasts are incorrect. It will produce a lower number if they're quite decent. Mean square error is used as the loss function, which is mentioned in equation 8. Where, $Y_i$ is the accurate value and $\widehat{Y}_i$ is the predicted value with an output layer dimension of n.

$$\text{Mean Square Error} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - (\widehat{Y}_i)^2 \tag{8}$$

## 2.3 Implementation of the modified BP-RRT* algorithm

The steps for implementation of the modified BP-RRT* algorithm are as follows.

**Step 1:** The start_point, target_point, obstacles, step_size and other parameters are taken as input

**Step 2:** The map space which is the workspace is split into 54 small sampling grids. These small grids are classified into three regions. So, each region will have 18 small sampling grids. The grid(division) in which the target point is located is identified.

**Step 3:** The nearest starting point to the target point is identified.

**Step 4:** In this step, the decision on which arm to be used for the operation will be determined.

Case(i): The target is located in the left arm region if it is in the grid layers 1-6, 19-24, 36-42.

Case(ii): The target is located in the right arm region if it is in the grid layers 13-18, 31-26, or 49-54.

Case(iii): The target is in the shared workspace region if it is in the grid layers 7-12, 25-30, or 43-48.

**Step 5:** Discretize the obstacle surface in the corresponding grid layers and calculate the obstacle volume in each sampling space.

**Step 6:** The sampling probability of the nodes, Si in each divided sampling space is determined.

**Step 7:** Backpropagation NN data prediction trained model is fed with Si of each space and $V_{free}$ in stage 1 as well as stage 2.

**Step 8:** Calculation of total nodes sampled in each stage

**Step 9:** Backpropagation NN-based stage step sampling of predicted data. In this step, the number of sampled nodes from stage1 is used as a threshold. The sampling process is given below

(i) The sampling process starts with finding the nearest node by identifying the nearest neighbour $X_{near}$ using the Euclidean distance method and a new node would be created with a step distance $X_{new}$.

(ii) For the new $X_{near}$, a collision test has to be conducted. If it fails, a new $X_{near}$ has to be identified and the process goes on till it finds a non-colliding node.

(iii) Once a non-colliding node is found, the parent node is reselected and rewired and the adopted node, n is incremented.

(iv) If n exceeds the number of nodes in Stage1, the first condition will fail and the system will navigate to Stage2.

(v) Within a uniformly distributed [0,1], a random number [16] is identified to follow the same sampling process from (i) to (iv)

(vi) If n exceeds the number of nodes in stage2, the condition will fail and the system navigates to stage 3.

(vii) The same process in (v) is followed for stage3 sampling and the same process would continue from (i) to (iv) until a goal point is reached to end the process.

**Step 10:** Based on the predicted data, the optimal path is plotted. This process continues for any of the three cases in step 4, which is the novelty of our modified algorithm for a dual-arm collaborative robot.

The algorithm for the process is explained in flowchart in the figure 8 as well as a pseudocode algorithm below for detailed and precise understanding. The Stage step sampling of predicted data based on the Backpropagation NN algorithm is shown in figure 9.

## 3 Experiments and analysis

Experimental validation has been performed using simulation as well as physical experiment analysis for the proposed modified BP-RRT* algorithm for a dual-arm collaborative robot. The simulation is performed with initial node coordinates set to [0,0,0] and final node coordinates set to [10,10,10], the step size is maintained at $\rho = 0.2$. The computational requirements for simulation are tabulated in Table 2.

This study employs an ABB YuMi dual-arm collaborative robot for simulation and physical experiments and primarily focuses on implementing the modified RRT-BP algorithm and compares with the results of search success rate, typical duration of a search, number of node samples, and path length

**Algorithm 1** Modified BP-RRT* Algorithm

**Inputs:** start_point, target_point, obstacles, step_size, and other parameters

**Outputs:** Optimal path from start_point to target_point

**procedure** DIVIDE_WORKSPACE_INTO_SMALL_SAMPLING_GRIDS
    Split workspace into 54 small sampling grids
    Classify grids into three regions with 18 small grids each
    Identify the grid containing the target point
**end procedure**
**procedure** IDENTIFY_NEAREST_STARTING_POINT
    Identify the nearest starting point to the target point
**end procedure**
**procedure** DETERMINE_ARM_SELECTION
    Decision on which arm to be used for the operation
    Case(i): Target in the left arm region
    Case(ii): Target in the right arm region
    Case(iii): Target in the shared workspace region
**end procedure**
**procedure** DISCRETIZE_OBSTACLE_SURFACE
    Discretize obstacle surface in the corresponding grid layers
    Calculate obstacle volume in each sampling space
**end procedure**
**procedure** DETERMINE_SAMPLING_PROBABILITY
    Determine sampling probability of nodes $S_i$ in each divided sampling space
**end procedure**
**procedure** USE_BACKPROPAGATION_NN_FOR_DATA_PREDICTION
    Feed Backpropagation NN with $S_i$ of each space and $V_{free}$ in stages 1 and 2
**end procedure**
**procedure** CALCULATE_TOTAL_SAMPLED_NODES
    Calculate total nodes sampled in each stage
**end procedure**
**procedure** BACKPROPAGATION_NN-BASED_STAGE_STEP_SAMPLING
    Initialize n = 0
    Repeat until a goal point is reached:
    i. Find nearest node $X_{near}$ using Euclidean distance method
    ii. Create new node with step distance $X_{new}$
    iii. Conduct collision test for $X_{new}$
    iv. If collision fails, find new $X_{near}$ until non-colliding node is found
    v. Reselect and rewire parent node, increment adopted node n
    vi. If n exceeds nodes in Stage 1, navigate to Stage 2
    vii. Uniformly distributed [0,1], random number [16], repeat (i) to (iv)
    viii. If n exceeds nodes in Stage 2, navigate to Stage 3
    ix. Uniformly distributed [0,1], random number [16], repeat (i) to (iv)
**end procedure**
**procedure** PLOT_OPTIMAL_PATH
    Based on the predicted data, plot the optimal path
**end procedure**

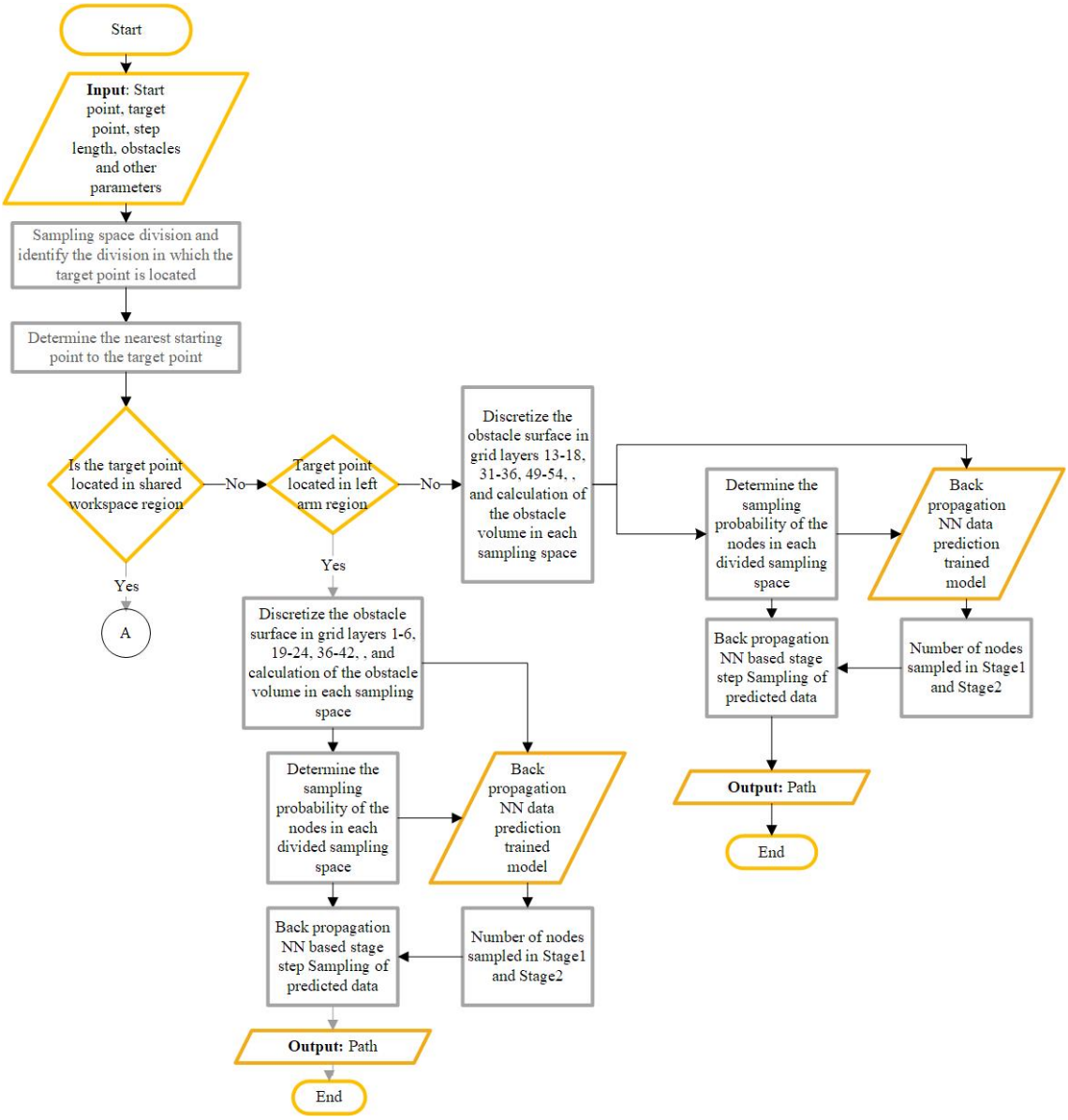



Figure 8: Algorithm flowchart

### 3.1 Simulation experiment analysis

As a modified algorithm was developed, two hundred experiments were performed in every group with other algorithms like RRT, RRT*, Improved P-RRT*[23], and BP-RRT*. The RRT algorithm

Table 2: Computational requirements

| | |
|---|---|
| Operating System | Ubuntu 18.04 LTS |
| CPU | AMD Architecture, 3.8 GHz |
| RAM | 16 GB RAM |
| GPU | NVIDIA GeForce GTX 1080 Ti |

creates more node samples and hence the path length is also very long. Whereas the RRT* has optimized the number of node samples created and hence, they have a much shorter path length which is around 33% shorter. But it has increased the typical duration of a search. This problem was solved by Improved P-RRT*[23] which has a much lesser typical duration of a search and slightly lesser path length than RRT*. However, the average amount of sample nodes has increased slightly. This alone could reduce the performance of the algorithm. The Backpropagation-based RRT* overcame all three challenges with a minimum typical duration of a search, a lesser average amount of sample nodes created and a shorter path length. But the above parameters can be reduced even more and

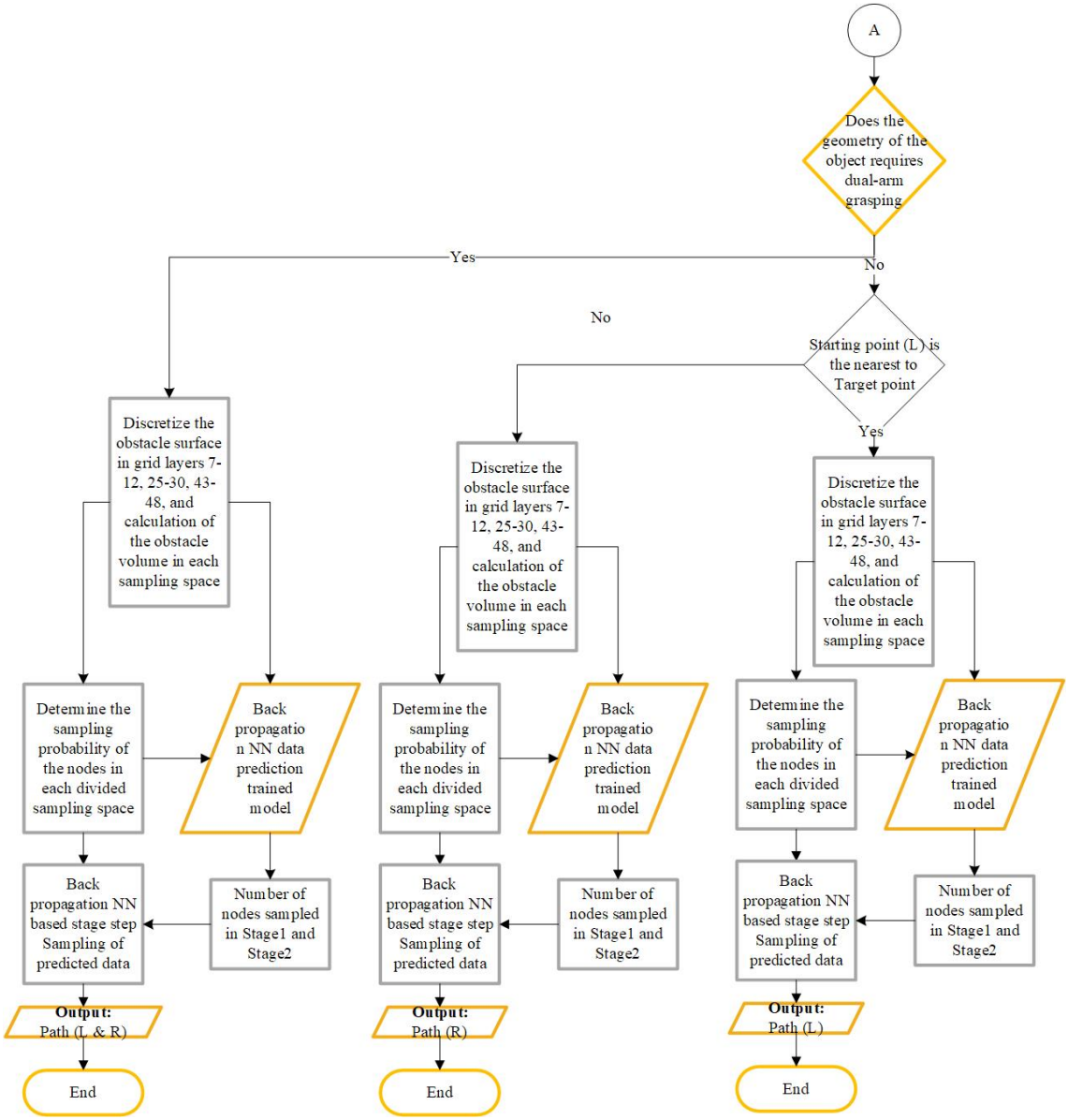



Figure 9: Algorithm flowchart (Extension)

still 100% search success rate can be achieved. The proposed Modified BP-based RRT* for dual arm has reduced the typical duration of a search, average amount of sample nodes and path length and still maintains a 100% search success rate.

ABB YuMi is simulated in MATLAB for moving around the obstacle to approach the target_1 and moving around the target_2. This is shown in figure 11. The movement of the joints are also plotted from the simulation which is shown in figure 12.

### 3.2 Comparison test

The modified BP-RRT* algorithm is implemented along with RRT, RRT*, Improved P-RRT* and BP-RRT* to check the performance of the novel BP-RRT* algorithm over search time, grasp time and success rate of search. The results of them in the same obstacle environment are tabulated in Table 3.

The average search time to grasp using Modified BP-RRT* has been significantly reduced. This is because of the region split based on the region-wise search in grids for the presence of obstacles well before the sampling is introduced in the algorithm. This has also reduced the length of the path. This is much more evident in figure 13 and figure 14.

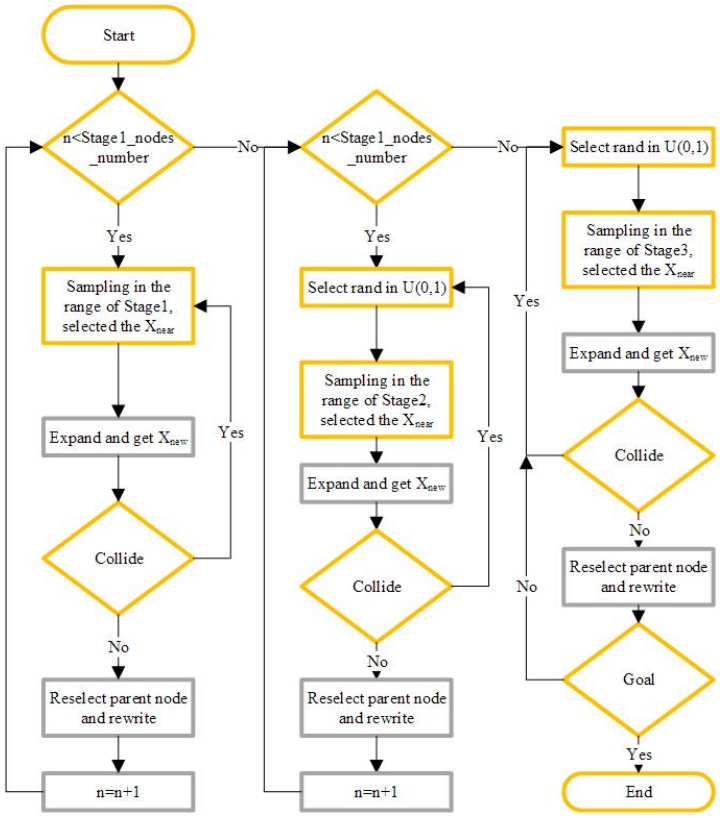



Figure 10: Stage step sampling of predicted data based on the Backpropagation NN algorithm

Table 3: Comparative analysis of algorithm performance

| Algorithm | Typical Length of the path | Average search and grasp time per second | Typical duration of a search per second | Success Rate (search) |
|---|---|---|---|---|
| RRT | 26.01 | 30.56 | 12.30 | 100% |
| RRT* | 18.15 | 27.96 | 10.75 | 100% |
| Improved P-RRT* | 18.29 | 22.87 | 6.77 | 100% |
| BP-RRT* | 18.13 | 20.85 | 5.56 | 100% |
| **Modified BP-RRT*** | **16.97** | **17.13** | **4.34** | **100%** |

### 3.3 Physical experiment

To check the performance of the algorithm practically, the Modified BP-RRT* algorithm is first simulated in MATLAB 2023a and tested which is shown in the figure 11 and later implemented in the ABB YuMi robot which is shown in the figure 19. All five algorithms were implemented for twenty simulation experiments and the results are averaged and tabulated in the table. The results exhibit the superiority of the Modified

BP-RRT* algorithm over others in simulation as well as physical experiments. The objects were placed in the workspace along with obstacles. The robot successfully handled the objects as per the algorithm. Wherein the objects placed on the left were handled by the left arm, the objects placed on the right were handled by the right arm, and the objects placed in the shared workspace were handled by either of the arms for a smaller geometry and both arms for larger geometry. The load handling capacity of the abb YuMi robot used is 0.5 Kg for each arm. Since, the depth camera mounted above the robot can measure only the geometry, objects of larger geometry are handled by dual-arm operations only.

Results from the simulation experiments exhibit that the Modified BP-RRT* algorithm surpasses the performance of other conventional algorithms and improves planning efficiency to a greater extent. They also proved to be computationally cheaper than the BP-RRT* algorithm as the number of sampling grids has been reduced to a lower level. Comparative analysis of the average time to grasp with their respective time to search in percentage for all RRT, RRT*, Improved P-RRT*, BP-RRT*,

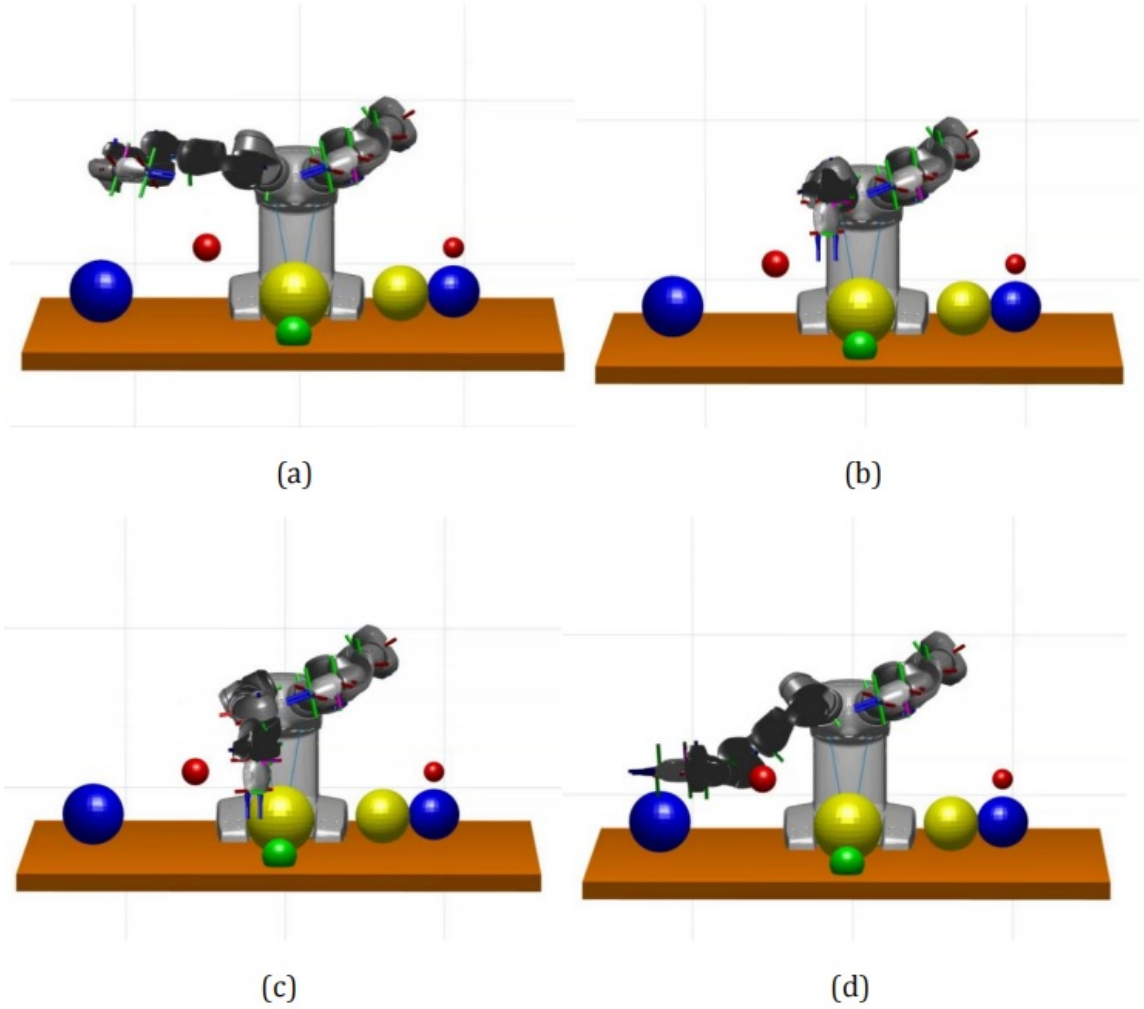



Figure 11: Simulated ABB YuMi in MATLAB (a) Initial position, (b) Moving around the obstacle to approach the target_1, (c) Moving around the obstacle to approach the target_2, (d) Goal position.

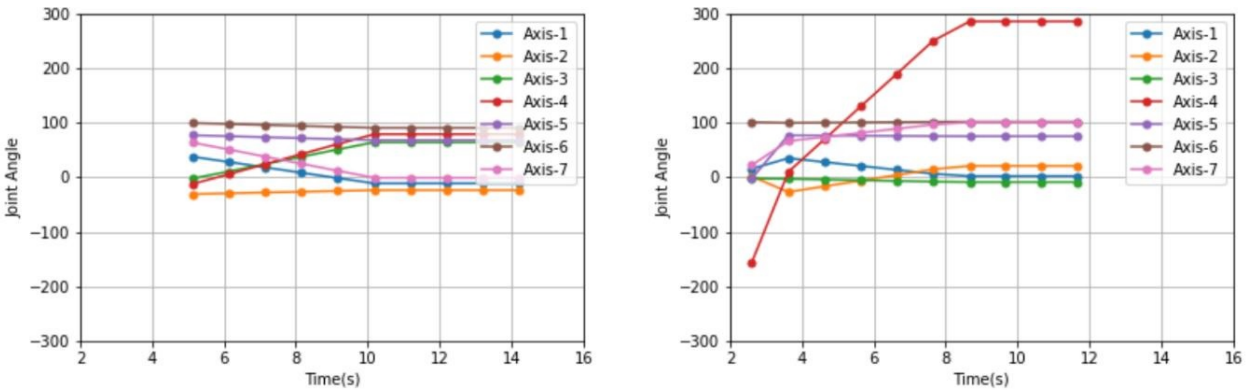



Figure 12: Planned path execution (a) Moving around the obstacle to approach the target_1, (b) Moving around the obstacle to approach the target_2

and Modified BP-RRT* is shown in figure 17. A comparative study to check the performance of the Modified BP-RRT* algorithm is done and it is shown in figure 15, figure 16, figure 17 and figure 18.

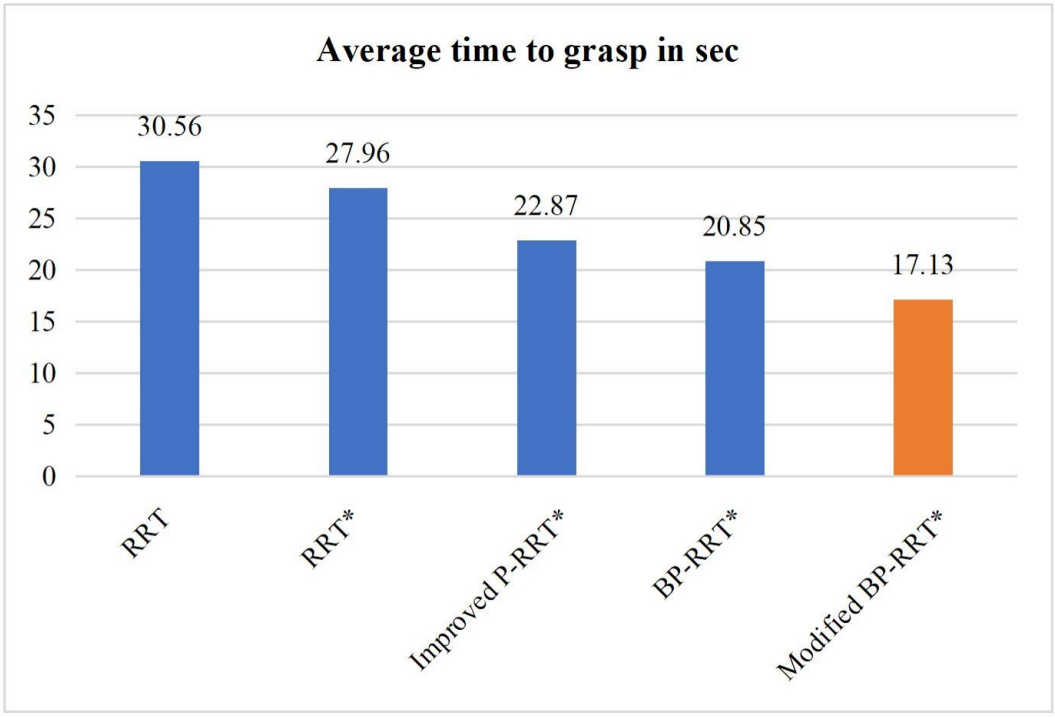



Figure 13: Comparative analysis of average grasp time – Modified BP-RRT* with RRT, RRT*, Improved P-RRT* and BP-RRT*

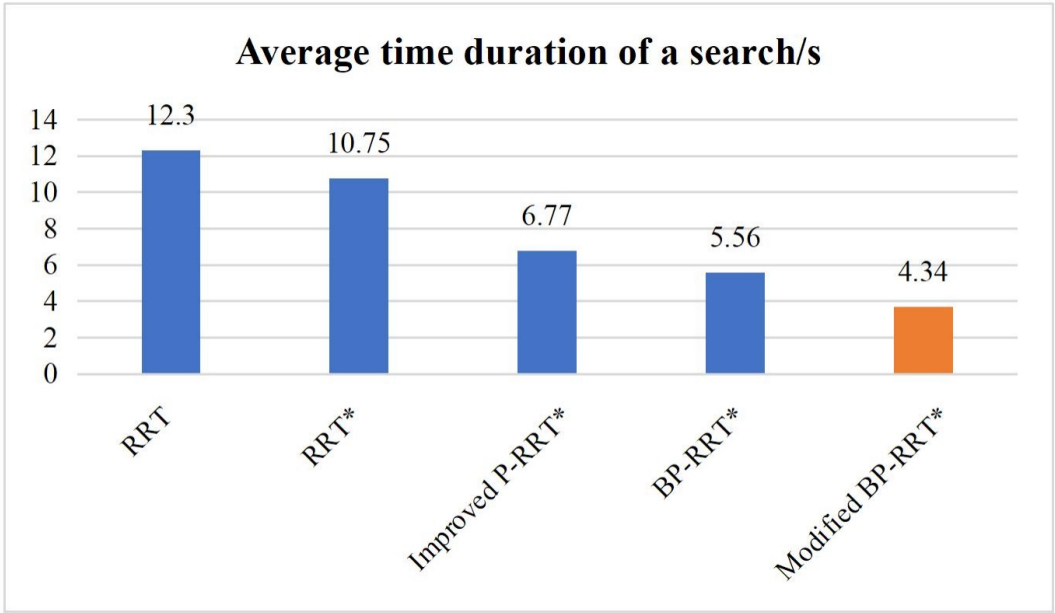



Figure 14: Comparative analysis of average time duration of a search – Modified BP-RRT* with RRT, RRT*, Improved P-RRT* and BP-RRT*

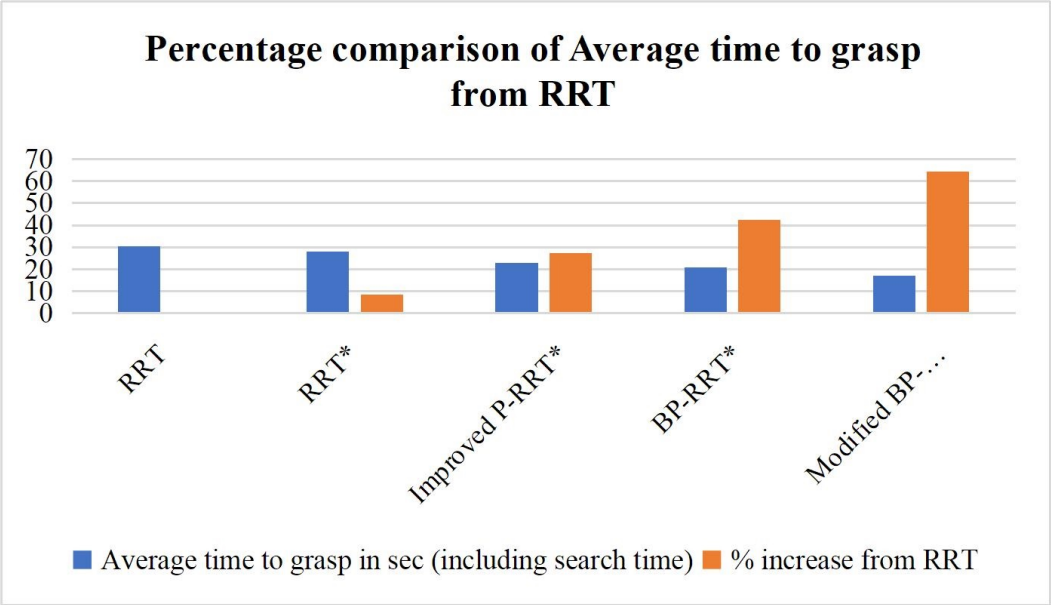



Figure 15: Comparative analysis of average time duration of a search in percentage – RRT with Modified BP-RRT*, RRT*, Improved P-RRT* and BP-RRT*

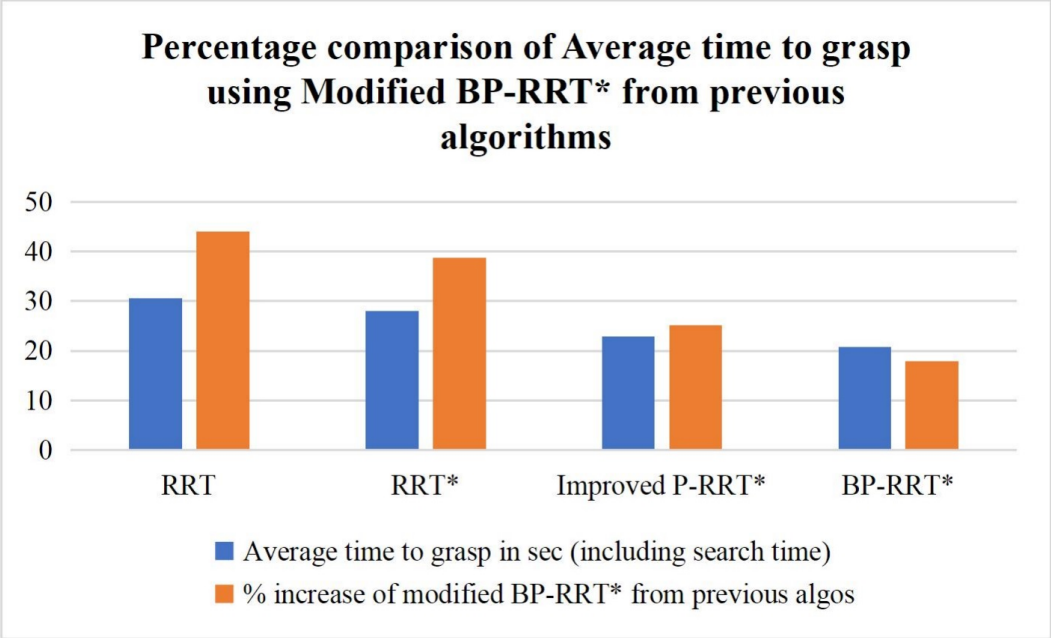



Figure 16: Comparative analysis of average time duration of a search in percentage – Modified BP-RRT* with RRT, RRT*, Improved P-RRT* and BP-RRT*

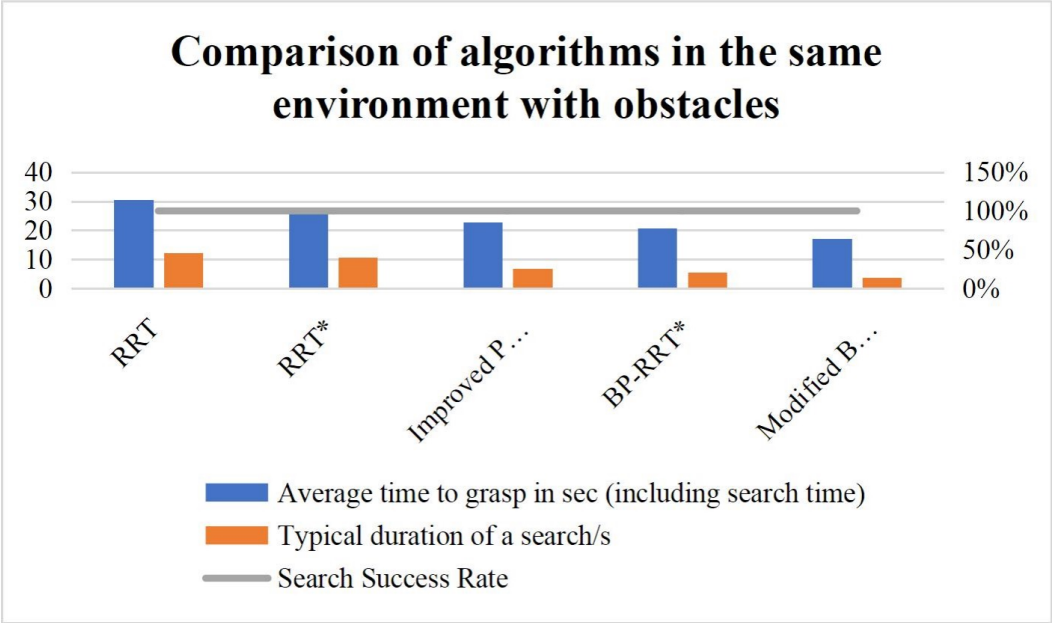



Figure 17: Comparative analysis of average time to grasp with their respective duration of search in percentage –RRT, RRT*, Improved P-RRT*, BP-RRT*, and Modified BP-RRT*

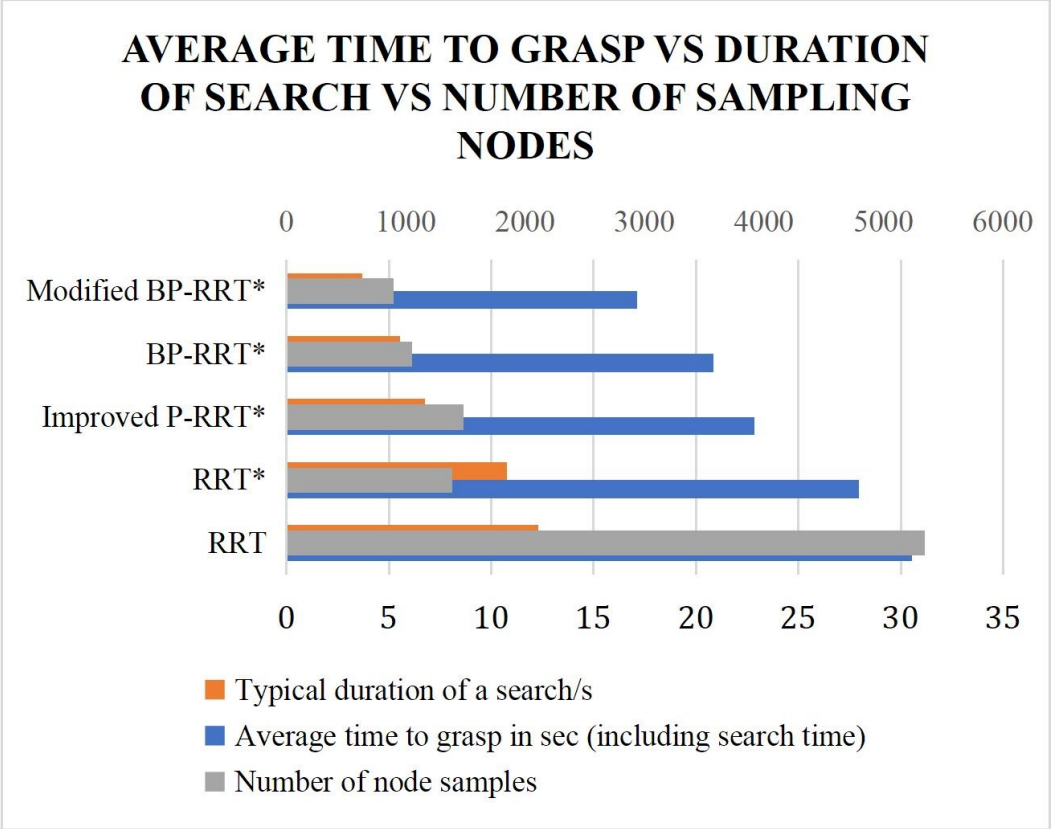



Figure 18: Comparison of average time to grasp with duration of search with number of sampling nodes

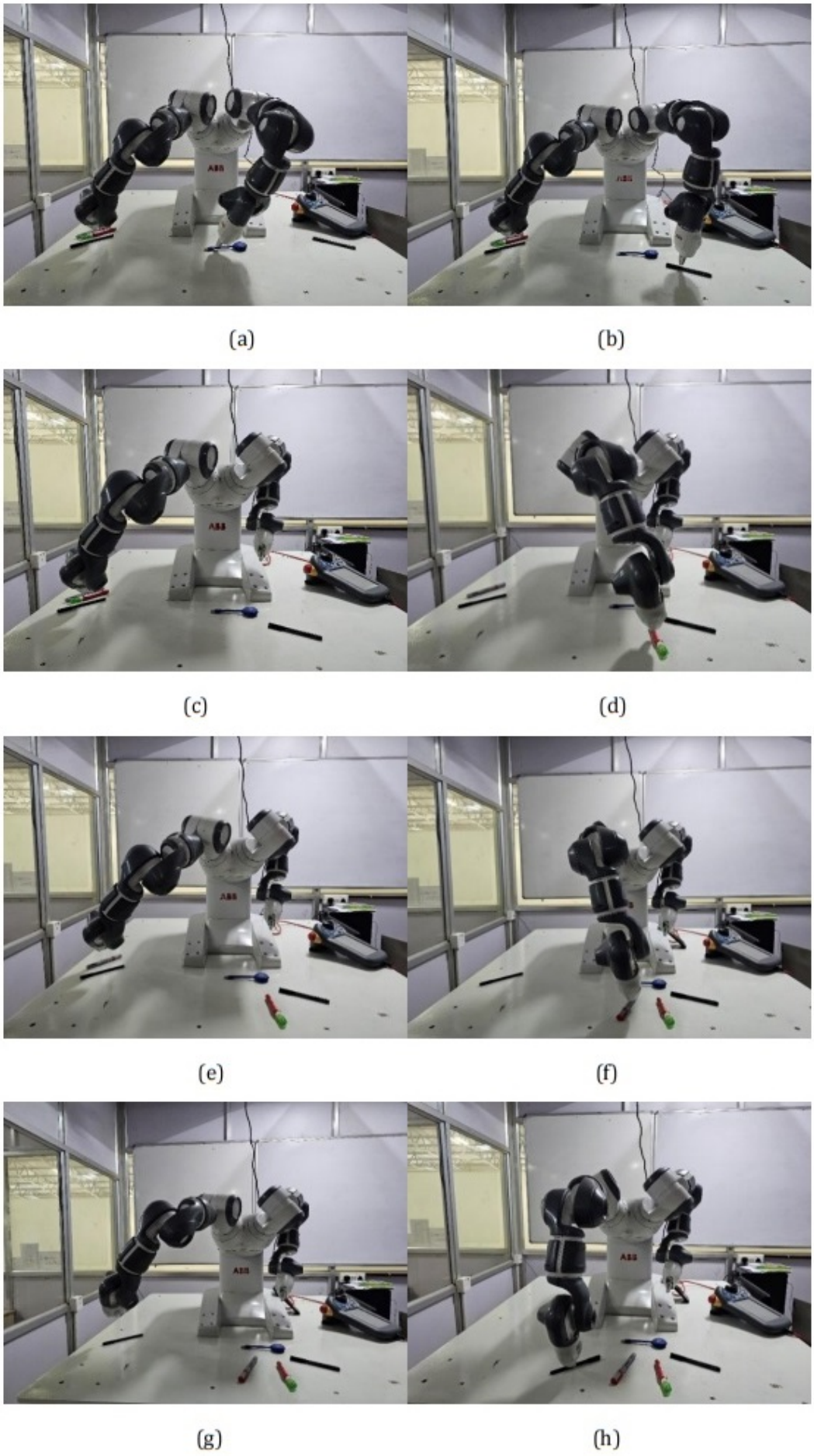



Figure 19: Physical experiments with ABB YuMi (a) YuMi operating in the left arm region, (c)-(d)-(e)-(f) YuMi operating in the shared workspace region, (g)-(h) YuMi operating in the right arm region

## 4 Conclusions and future work

In today's smart manufacturing industries, we look for a dual-arm robot more than a single-arm robot, as they can work on a bigger workspace and do work independently as well as collaboratively. When obstacles are present in the workspace, they create a unique challenge in collaborative robots as the robot should find the shortest path to grasp an object without hitting any obstacle in that extended workspace. When an algorithm is developed, prospects of its computational efficiency, scalability and easy implementation have to be considered. In search-based algorithms, the computational requirements are very high. When computational requirements are very high, then its future use cases in embedded systems and IOT are minimal as standalone systems cannot afford large computational resources. So, it is necessary to create a robust and foolproof sampling-based algorithm for collaborative robots exclusively. There are many sampling-based algorithms developed for single-arm robots namely RRT [14], RRT*[12], Improved P-RRT*[23] and BP-RRT*[8]. Among these algorithms, BP-RRT* showed a greater maturity in the typical duration of a search, average amount of sample nodes created, path length and average grasp time despite producing a 100The BP-RRT* algorithm divides the sampling space and determines the sampling probability of every part of the nodes based on the density of obstacles present in that multi-obstacle 3D space to render a staged local sampling. The modified BP-RRT* algorithm proposed the division of the extended workspace W into 54 grids and classified them into 3 namely for the left arm region, right arm region and a shared workspace region. So, each region will have 18 workspace grid regions. This resulted in much lesser typical duration of a search, average amount of sample nodes created, path length and average grasp time all with a 100% search success rate. In the initial steps of the algorithm, the obstacle, start point and target point define which region among the three has the target point as well as obstacles. Hence, only 18 grids are searched and sampled. This further simplified the process as in BP-RRT*, the sampling has to be performed for 27 workspace grid regions. The average time to grasp, the typical duration for search, and the number of node samples created have reduced by 17.84%, 33.45% and 14.79% from BP-RRT* algorithm respectively. Efficient path planning has optimized the robot's movements, limiting the number of node samples created, and hence path length, reducing the time required to search, and grasp. This is crucial for tasks where speed and efficiency are important while avoiding collisions with obstacles or other objects in their environment. Modified BP-RRT* algorithm has contributed to achieving high levels of accuracy and precision in the execution of tasks with obstacles of known geometry as well as unknown geometry which is a must-have feature for a dual-arm collaborative robot as they handle more complex tasks and navigate intricate environments faster. These are the implications of this novel Modified BP-RRT* algorithm. This work can be improvised in the future remarkably. Especially in dynamic environments [4] where robots are deployed in applications of sorting damaged products in assembly operations, assembling a specific part in manufacturing operations, general sensor-guided object manipulation in automation industries, applications involving part presentation through conveying mechanisms etc. The future scope also extends with implementing a multi-modal path planning engineered with a deep-learning-based object localization and grasp localization not only in a structured clutter environments but also in piled clutter environments [27] with a low-level feedback mechanism for successful path planning and grasping.

## References

[1] A'Campo-Neuen, Annette (2022). Lambert's Work on Geographic Map Projections, *In: Mathematical Geography in the Eighteenth Century: Euler, Lagrange and Lambert, Springer*, 183-202, 2022.

[2] Barber, C.B. and Dobkin, D.P. and Huhdanpaa, H (1993). The Quickhull Algorithm for Convex Hulls, *ACM Transactions on Mathematical Software*, 22(4), 1993.

[3] Brezovnik, S., Gotlih, J., Balič, J., Gotlih, K., and Brezočnik, M. (2014). Optimization of an Automated Storage and Retrieval Systems by Swarm Intelligence. *25th Daaam International Symposium on Intelligent Manufacturing and Automation*, 100, 1309-1318, 2014.

[4] Chen, Pengzhan, and Weiqing Lu. (2021). Deep Reinforcement Learning Based Moving Object Grasping, *Information Sciences* 565, 62-76, 2021.

[5] Elbanhawi, Mohamed, and Milan Simic (2014). Sampling-Based Robot Motion Planning: A Review, *IEEE Access* 2, 56-77, 2014.

[6] Erke, S., Bin, D., Yiming, N., Qi, Z., Liang, X., and Dawei, Z. (2020). An Improved A-Star Based Path Planning Algorithm for Autonomous Land Vehicles, *International Journal of Advanced Robotic Systems* 17(5), 1729881420962263, 2020.

[7] Foumani, Mehdi, Asghar Moeini, Michael Haythorpe, and Kate Smith-Miles (2018). A Cross-Entropy Method for Optimising Robotic Automated Storage and Retrieval Systems, *International Journal of Production Research* 56(19), 6450-6472, 2018.

[8] Gao, Q. and Yuan, Q. and Sun, Y. and Xu, L. (2023). Path Planning Algorithm of Robot Arm Based on Improved RRT* and BP Neural Network Algorithm, *Journal of King Saud University-Computer and Information Sciences* 35(8), 101650, 2023.

[9] Ladra, S. and Paramá, J.R. and Silva-Coira, F. (2016). Compact and Queryable Representation of Raster Datasets, *Proceedings of the 28th International Conference on Scientific and Statistical Database Management (SSDBM '16), ACM* 1-12, 2016.

[10] He, Zhibo, Chenguang Liu, Xiumin Chu, Rudy R. Negenborn, and Qing Wu (2022). Dynamic Anti-Collision A-Star Algorithm for Multi-Ship Encounter Situations, *Applied Ocean Research* 118, 102995, 2022.

[11] Husain, Zainab, Amna Al Zaabi, Hanno Hildmann, Fabrice Saffre, Dymitr Ruta, and A. F. Isakovic. (2022). Search and Rescue in a Maze-Like Environment with Ant and Dijkstra Algorithms, *Drones* 6(10), 273, 2022.

[12] Karaman, S. and Frazzoli, E. (2010). Optimal Kinodynamic Motion Planning Using Incremental Sampling-Based Methods, *49th IEEE Conference on Decision and Control (CDC)* 7681-7687, 2010.

[13] Kavraki, L.E. and Kolountzakis, M.N. and Latombe, J.-C. (1998). Analysis of Probabilistic Roadmaps for Path Planning, *IEEE Transactions on Robotics and Automation* 14(1), 166-171, 1998.

[14] LaValle, Steven (1998). Rapidly-Exploring Random Trees: A New Tool for Path Planning, *Research Report* 9811, 1998.

[15] Li, Jing, Ji-hang Cheng, Jing-yuan Shi, and Fei Huang. (2012). Brief Introduction of Backpropagation (BP) Neural Network Algorithm and its Improvement, *Advances in Computer Science and Information Engineering, Springer* 553-558, 2021.

[16] Vitter, Jeffrey Scott. (1984). Faster Methods for Random Sampling, *Communications of the ACM* 27(7),703-718,1984.

[17] Liu, Chenguang, Qingzhou Mao, Xiumin Chu, and Shuo Xie (2019). An Improved A-Star Algorithm Considering Water Current, Traffic Separation and Berthing for Vessel Path Planning, *Applied Sciences* 9(6),1057,2019.

[18] Patel, Utsav, Nithish K. Sanjeev Kumar, Adarsh Jagan Sathyamoorthy, and Dinesh Manocha. (2021). DWA-RL: Dynamically Feasible Deep Reinforcement Learning Policy for Robot Navigation Among Mobile Obstacles, *2021 IEEE International Conference on Robotics and Automation (ICRA)* 6057-6063, 2021.

[19] Qi, J. and Yang, H. and Sun, H. (2020). MOD-RRT*: A Sampling-Based Algorithm for Robot Path Planning in a Dynamic Environment, *IEEE Transactions on Industrial Electronics* 68(8), 7244-7251, 2020.

[20] Salem, Israa Ezzat, Maad M. Mijwil, Alaa Wagih Abdulqader, and Marwa M. Ismaeel (2022). Flight-Schedule Using Dijkstra's Algorithm with Comparison of Routes Findings, *International Journal of Electrical and Computer Engineering* 12(2),1675, 2022.

[21] Swinbank, Richard, and R. James Purser. (2006). Fibonacci Grids: A Novel Approach to Global Modelling, *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography* , 132(619), 1769-1793, 2006.

[22] Wang, Jiankun, Wenzheng Chi, Chenming Li, Chaoqun Wang, and Max Q-H. Meng (2020). Neural RRT*: Learning-Based Optimal Path Planning, *EEE Transactions on Automation Science and Engineering* 17(4), 1748-1758, 2020.

[23] Yi, Junhui, Qingni Yuan, Ruitong Sun, and Huan Bai (2020). Path Planning of a Manipulator Based on an Improved P_RRT* Algorithm, *Complex & intelligent systems* 8(3), 2227-2245, 2022.

[24] Zafar, Mohd Nayab, and J. C. Mohanta. (2018). Methodology for Path Planning and Optimization of Mobile Robots: A Review, *Procedia computer science* 133, 141-152, 2018.

[25] Zhou, Yulan, and Nannan Huang. (2022). Airport AGV Path Optimization Model Based on Ant Colony Algorithm to Optimize Dijkstra Algorithm in Urban Systems, *Sustainable Computing: Informatics and Systems* 35, 100716, 2022.

[26] Hippolitus, A.J. and Senthilnathan, R. and Malla, O. (2021). Simulation of Grasp Localization Inferences for a Dual-Arm Collaborative Robot, *IOP Conference Series: Materials Science and Engineering* 1012,1, 012004, 2021.

[27] Newbury, R., Gu, M., Chumbley, L., Mousavian, A., Eppner, C., Leitner, J., Bohg, J., Morales, A., Asfour, T., Kragic, D. and Fox, D. (2023). Deep Learning Approaches to Grasp Synthesis: A Review, *IEEE Transactions on Robotics* 2023.

[28] Wan, W. and Harada, K. (2016). Developing and Comparing Single-Arm and Dual-Arm Regrasp, *IEEE Robotics and Automation Letters* 1(1), 243-250, 2016.






*Cite this paper as:*

Hippolitus, J.A.; Senthilnathan R. (2024). Quicker Path planning of a collaborative dual-arm robot using Modified BP-RRT* algorithm, *International Journal of Computers Communications & Control*, 19(3), 6379, 2024.
https://doi.org/10.15837/ijccc.2024.3.6379