RESEARCH ARTICLE

RRT-CS: A free-collision planner for capsule-like SCORBOT by iterated learning

Hung Nguyen[1], Thanh Phuong Nguyen[1], Song Hung Nguyen[2,3], Ha Quang Thinh Ngo[2,3]*

1 HUTECH Institute of Engineering, HUTECH University, Ho Chi Minh, Vietnam, **2** Faculty of Mechanical Engineering, Ho Chi Minh City University of Technology (HCMUT), Ho Chi Minh, Vietnam, **3** Vietnam National University-Ho Chi Minh City (VNU-HCM), Ho Chi Minh, Vietnam

* nhqthinh@hcmut.edu.vn

Abstract

In this study, we present an enhanced Rapidly-exploring Random Trees (RRT) algorithm integrated with a visual servoing technique for recognizing unknown environments. The robotic platform utilized is the SCORBOT-ER-VII, which consists of five links, servo motors, gearboxes, and an end-effector. Several target objects are used to define the initial position, obstacles, and destination. To evaluate the effectiveness and robustness of our approach, we conducted both numerical simulations and hardware experiments across three test scenarios, ranging from obstacle-free environments to complex obstacle configurations. The results indicate that planning time increases proportionally with scenario complexity. The trajectory smoothing process accounts for less than 10% of the total processing time, while path shortening constitutes one-third, and RRT-based profile generation comprises the remaining two-thirds. These findings clearly demonstrate the efficiency of our approach in terms of computational time, making it well-suited for real-world applications.

Citation: Nguyen H, Nguyen TP, Nguyen SH, Ngo HQT (2025) RRT-CS: A free-collision planner for capsule-like SCORBOT by iterated learning. PLoS One 20(5): e0323045. https://doi.org/10.1371/journal.pone.0323045

Editor: Himadri Majumder, G H Raisoni College of Engineering and Management, Pune, INDIA

Received: February 15, 2025

Accepted: April 1, 2025

Published: May 19, 2025

Data availability statement: All relevant data are within the paper and its Supporting Information files.

Funding: The author(s) received no specific funding for this work.

Competing interests: The authors have declared that no competing interests exist.

1 Introduction

In the field of robotic control, autonomy is a key characteristic for exploring various environments [1,2], including unknown or foggy maps [3], dynamic obstacles, and emergency rescue scenarios [4]. As a fundamental research topic in autonomous robotics, motion planning has attracted significant attention. Most profile generation procedures require searching for multiple collision-free trajectories from the starting point to the destination [5–7]. In some cases, planning optimization also aims to minimize travel distance while satisfying additional constraints.

Probabilistic Roadmaps (PRM) and Rapidly-exploring Random Trees (RRT) are classified as sampling-based path planning algorithms, where a point is sampled within the working map and its nearest neighbor is explored from the existing tree. In the early stages of development, these methods prioritized rapid planning solutions over optimality. In [8], researchers highlighted that RRT is not asymptotically optimal

and proposed Neural RRT*—a novel approach incorporating learning-based capabilities for potentially optimal path planning. This strategy is particularly suitable for practical applications such as autonomous driving and warehouse robotics. In [9], investigators leveraged convolutional neural networks to detect potential collisions while considering an optimal distance metric to estimate sample costs based on path curvature.

In this paper, our contributions are threefold: (i) introducing the robotic platform and related techniques, (ii) developing a motion planning scheme that integrates a robot control strategy with an advanced training model for a computer vision-based approach, and (iii) validating the proposed scheme on a real-world robotic system. The structure of this paper is as follows: Previous Works section reviews related works, followed by a description of the robotic platform, its parameters, the training model, and the camera setup in Preliminaries section. The Proposed Approach section presents the proposed approach, including the conceptual design, motion planner, and post-processing steps. Simulations and Experiments section discusses the experimental results, and Conclusions section concludes the paper with insights and directions for future research.

2 Previous works

Generally, RRT planners could be decomposed into various primitives while alterations or likenesses among them are visible [10–13]. Many scholars focused on the parameters and heuristics in sampling-based planners and their implementations in optimizing motion of hardware platform. The existing challenges in the sampling-based path planning are to cost large memory of computer, low speed of convergence, depend on nearest neighbor search and require post processing. On account of these troubles, several efforts for enhancing this motion planner have been made in recent years.

2.1 Sampling algorithm

Sampling process guides robot to extend the configuration space. Because it could spend a lot of time and cost in wide area and not all configurations are uniform, some algorithms have been investigated to overcome such as Voronoi graph [14,15], probabilistic roadmap method [16] or goal biasing [17].

2.2 Autonomous exploration

The ability of exploration becomes one of the key characteristics of robot in the complicated environment. In both visibility region [18] and multi-level region [19], adaptive sampling scheme is useful for autonomous robot even motion planner fails to discover.

2.3 Parameter metric

Most of the sampling-based strategies rely on metrics for their expandable search. Picking a proper metric is always hard for an operator so that it requires a good

estimation and better understanding of this system [20,21]. Also, it can be used multiple times during the trajectory planning procedure.

2.4 Collision checking

One of the critical issues to evaluate the motion planner is to generate free-collision path. In some cases, it is classified into C_{free} which provides free-collision path and C_{obs} which contains the paths colliding with any obstacle. Collision checking would be called several times while robot is moving [22–24].

2.5 Robotic singularity

Conventionally, an industrial manipulator could be controlled in both joint space and Cartesian space. For joint-space driving commands, a reference set of joint positions should be provided. Later, it is translated or rotated each joint to the desired joint positions. For Cartesian-space motion commands, both the orientational and positioning pose of robotic gripper must be inserted. Then, motion planner would compute the inverse position and velocity kinematics for all robotic links. In this stage, a robot singularity in which its end-effector becomes blocked in certain direction, might arise. Consequently, there are several techniques to avoid such as vector field [25], geometric method [26] or genetic algorithm [27].

For further analysis, Table 1 summarizes the key limitations and its impact of the state-of-the-art researches in related topics. They are categorized in three groups which represent the separated applications. Then, in the same domain, they are classified owing to the specific techniques or definite design. Each of those researches is investigated to specify the key challenges and its effects in the theme of robotic control.

3 Preliminaries

To clarify our works, several components including hardware platform, model training and interactive objects, system configuration for both 5-DOF (Degree-Of-Freedom) robot and digital camera.

3.1 Robotic platform

In this system, five DOFs robotic arm as Fig 1 is deployed to manipulate in front of objects. Its mechanical structure is vertical articulated hardware. There are five servo motors and harmonic drives, belt and pulley, and optical encoders to feedback signals. To pick an object, it is necessary to attach one robotic gripper at the end-effector. Optionally, two-finger gripper or three-finger gripper should be considered. However, to demonstrate the obstacle avoidance algorithm, robot arm without gripper is sometimes utilized.

3.2 Model training and target objects

Computer vision has attracted a lot of researches in various domains such as robotics, machining design, or information technology. It assists to capture, grasp, and analyze the highly understanding level of visual contents. To match with our purpose, object recognition, localization and segmentation are available to study. Currently, You Only Look Once (YOLO) [39] is very popular to detect an object with significantly high accuracy.

To customize our detector, there are four steps such as data collection, data preparation, model training and model inference. In Fig 2, three kinds of objects are start, obstacle and goal. Dataset is collected from different views and shapes to provide precise detection. Robot must begin its task from start, avoid obstacle and toward to goal. Since these obstacles have rectangular shapes, the point-cloud coordinates of top-view corners of bounding box are A, B, C and D respectively. It is projected four corners down to table surface with depth $h = depth\ of\ ABCD - depth\ of\ table$ as Fig 3.

To obtain the 3D coordinate, the camera-inspired relative calibration is depicted as Fig 4. For more details, mathematical relations of these parameters are

Table 1. List of the systematic analysis for the state-of-the-art researches.

Category	Issue	Key limitations	Impact
Path planning	Standard RRT [28]	Purely random sampling, no trajectory optimization	Results in suboptimal paths with excessive waypoints and sharp turns
	Enhanced RRT [29]	Asymptotically optimal but computationally expensive	Slow convergence, especially in high-dimensional spaces
	Bi-RRT [30]	Faster than RRT but lacks smoothing	Still prone to jerky motions, requires post-processing
	Learning-based RRT [31]	Requires large datasets and training time	Not adaptable to dynamic environments without retraining
Collision avoidance	Axis-Aligned Bounding Boxes (AABB) [32]	Overestimates obstacles, leading to overly conservative paths	Unnecessary detours
	Convex Hulls [33]	Fails to represent concave obstacles	Risk of false negatives (missed collisions)
	Point Clouds (Voxel-Based) [34]	Computationally expensive for real-time planning	Slow path updates
Visual solution for motion planning	Integrated interactive framework [35]	Be unable to provide continuous, reliable feedback for control	Becomes particularly problematic during large lateral displacements or in dynamic scenes where obstacles may occlude important features
	3D vision-based system [36]	Lack the necessary accuracy, robustness, and resolution in the existing vision systems, especially under variable lighting conditions or on uneven surfaces	The integration of advanced 3D vision technologies enhances the precision and reliability of damage recognition
	Safe and efficient navigation [37]	Suffer from issues such as random sampling inefficiencies, high memory demands, and limited adaptability to dynamic obstacles	It provides a comprehensive review of both global and local planning strategies
	Path planning in unstructured environment [38]	Struggle with adequately incorporating kinematic constraints and managing complex environmental dynamics, often resulting in inefficient or unsafe paths	The proposed improved kinematically constrained bi-directional RRT approach enhances planning efficiency and path quality. This advancement is expected to significantly benefit agricultural robotics by enabling safer and more reliable navigation in challenging, unstructured outdoor environments.

https://doi.org/10.1371/journal.pone.0323045.t001

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{1}$$

where

$\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$: intrinsic matrix of digital camera, $f_x,\ f_y$: focal length, $c_x,\ c_y$: principal points

$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$: 2D image coordinate

$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$: 3D world coordinate

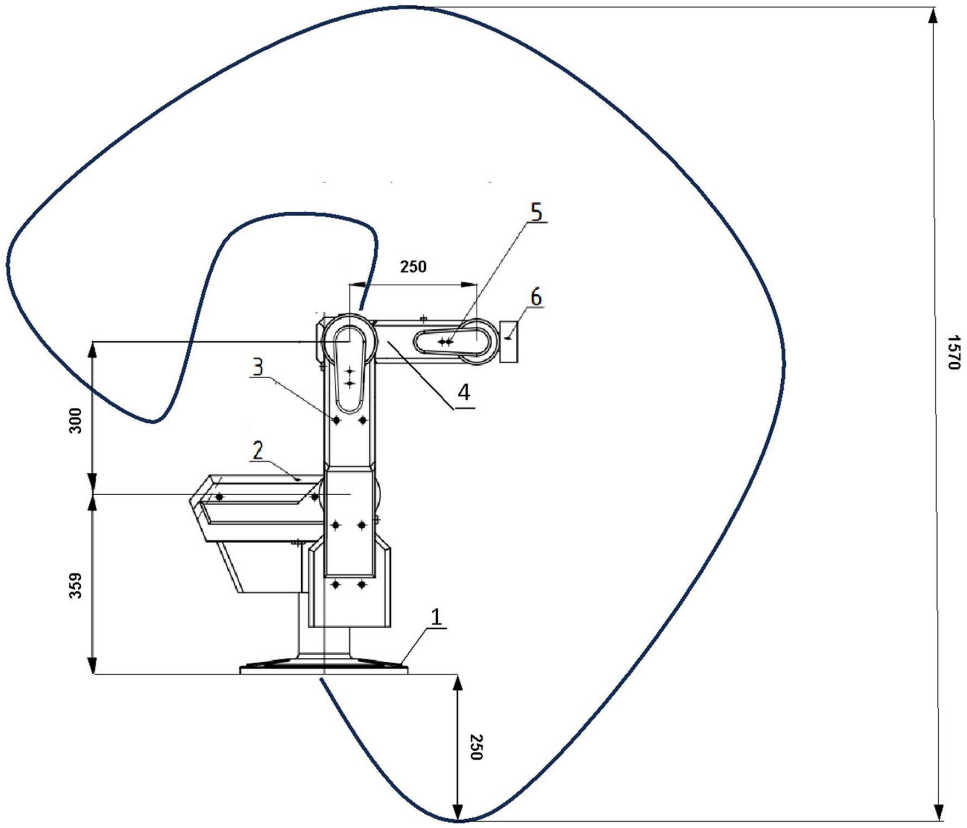

Fig 1. Workspace of the proposed robotic platform, (1) base platform, (2) joint 1, (3) socket head screw M5, (4) joint 3, (5) socket head screw M3 and (6) end-effector.

https://doi.org/10.1371/journal.pone.0323045.g001

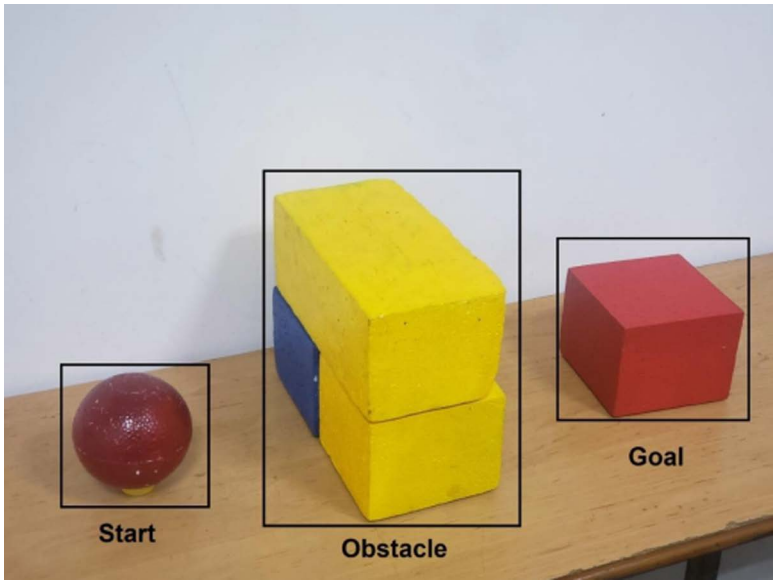

Fig 2. Target objects in our research.

https://doi.org/10.1371/journal.pone.0323045.g002

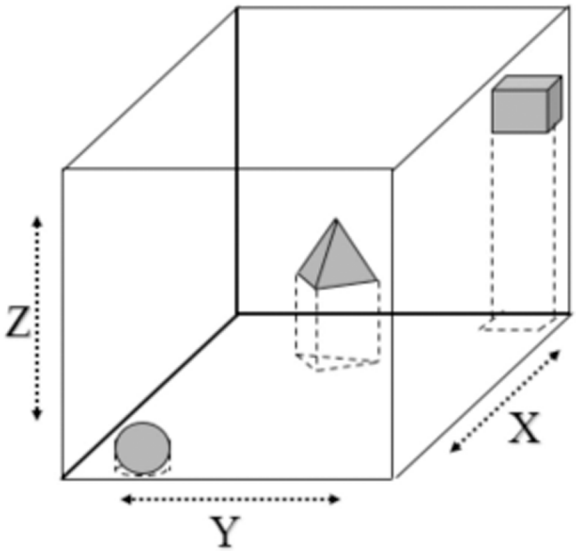

Fig 3. Illustration of scenario modeling.

https://doi.org/10.1371/journal.pone.0323045.g003

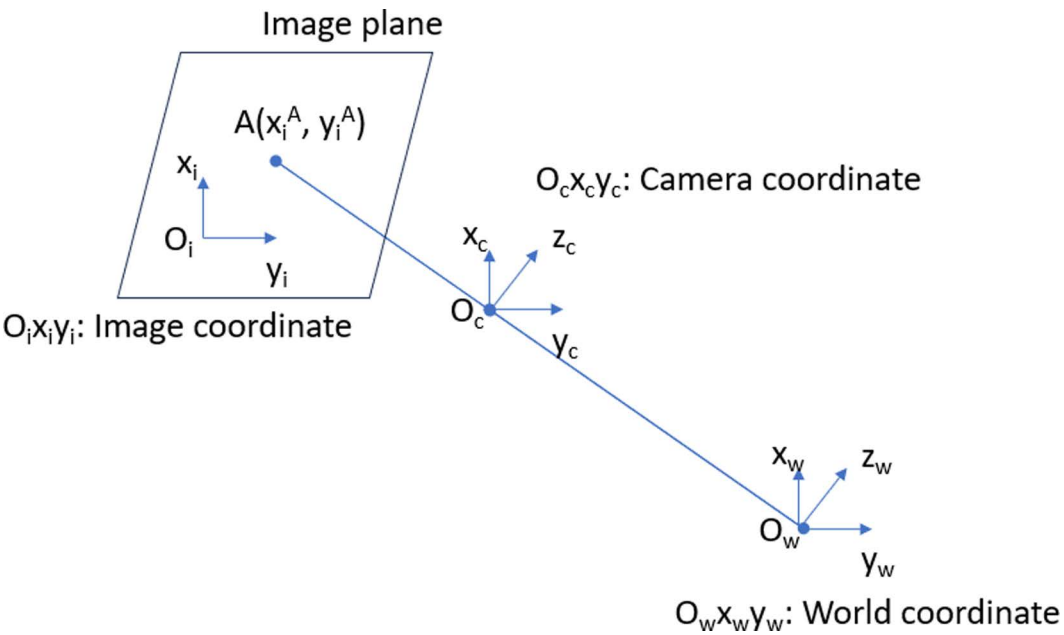

Fig 4. Relative constraints in multiple coordinates.

https://doi.org/10.1371/journal.pone.0323045.g004

$\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}$: extrinsic matrix of digital camera, r_{11}, r_{12}, r_{13}, r_{21}, r_{22}, r_{23}, r_{31}, r_{32}, r_{33}: camera rotation, t_1, t_2, t_3: camera translation

Using the Software Development Kit (SDK) of our camera, then intrinsic matrix of those parameters is measured as

$$\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 349.8 & 0 & 338.12 \\ 0 & 349.8 & 190.3415 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

Later, relative transformations should be identified to convert the real-world coordinate to image coordinate on a Charge Coupled Device (CCD) sensor of camera. We assume that X, Y, Z are the actual coordinates of the object (mm) relative to the camera in space and x, y are the coordinates of the pixel (mm) on the CCD optical sensor. In Fig 3, we apply Thales theorem

$$\frac{Y}{Z} = \frac{y}{f_y}, \frac{X}{Z} = \frac{x}{f_x} \rightarrow x = \frac{f_x X}{Z},\ y = \frac{f_y Y}{Z} \tag{3}$$

We have the conversion matrix of image coordinates on CCD and real coordinates as follows

$$\begin{bmatrix} \widetilde{x} \\ \widetilde{y} \\ \widetilde{z} \end{bmatrix} = T \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{4}$$

In which matrix: $T = \begin{bmatrix} f_x & 0 & 0 & 0 \\ 0 & f_y & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

Regularly, the image coordinates would be moved to the left corner of the image and considered as coordinates (0,0). The mm coordinates are converted to pixel coordinates.

$$x = (u - u_0) \times s_x \rightarrow u = \frac{x}{s_x} + u_0 \tag{5}$$

$$y = (v - v_0) \times s_y \rightarrow v = \frac{y}{s_y} + v_0 \tag{6}$$

where,

$u_0,\ v_0$ are the pinhole center coordinate

s_x, s_y are the length of each pixel in mm in the x and y direction

The formula to convert mm coordinates on the frame to pixel coordinates is as follows

$$x = (u - u_0) \times s_x \rightarrow u = \frac{x}{s_x} + u_0 = \frac{f_x X}{Z s_x} + u_0 \tag{7}$$

$$y = (v - v_0) \times s_y \rightarrow v = \frac{y}{s_y} + v_0 = \frac{f_y Y}{Z s_y} + v_0 \tag{8}$$

Hence,

$$p' = \begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} \frac{f_x}{s_x} & 0 & u_0 \\ 0 & \frac{f_y}{s_y} & v_0 \\ 0 & 0 & 1 \end{bmatrix} = K \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}, s = Z \tag{9}$$

$$p = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{p'}{s} \tag{10}$$

K: Internal parameter matrix of the camera

We suppose that this camera has coordinates relative to a different origin system. As a result, we have a general formula that transforms from the system

$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = K \times M \times \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{11}$$

where,

M: external parameter matrix used to convert object coordinates from global coordinates to camera coordinates. In order to convert a pixel coordinate to real coordinates, we proceed as following

$$Zm_p = KM_c = K(RM_W + T) => Zm_p = K(RM_W + T) => M_W = R^{-1}(K^{-1}Zm_p - T) \tag{12}$$

$m_p = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$: pixel coordinate and $M_W = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$: global coordinates

Camera distance to the workspace $Z = 350$ (mm). We have the internal matrix of the camera $K = \begin{bmatrix} \frac{349.3}{350} & 0 & 338 \\ 0 & \frac{347.4}{350} & 198 \\ 0 & 0 & 1 \end{bmatrix}$.

In order to convert a point from one coordinate system to another (in this case, change from camera coordinate system to robot coordinate system), we need to find the transfer matrix M

$$\begin{array}{ccc} P_R & = & M_{trans}P_C \\ \leftrightarrow \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} & = & \begin{bmatrix} R_{11} & R_{12} & R_{13} & T_X \\ R_{21} & R_{22} & R_{23} & T_Y \\ R_{31} & R_{32} & R_{33} & T_Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \end{array} \tag{13}$$

where,

P_R: The known coordinates in advance of the robot coordinate system

P_C: The known coordinates in advance of the camera coordinate system

M_{trans}: transformation matrix

To determine the M matrix according to above equations, it is necessary to find any four points in local coordinates by moving robotic arm to four different points in the workspace as Fig 5. After that, its values in pixels are recorded by using a camera as Fig 6.

For avoiding the appearance of outliers, four selected points as Table 2 must have at least one point which is not on the same plane as the remaining points (specifically z much be different).

At that point, applying these parameters to the formular, we get:

$$M_{trans} = \begin{bmatrix} 0 & -1 & 0 & 504 \\ -1 & 0 & 0 & 460 \\ 0 & 0 & -1 & 591 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{13}$$

Therefore, we obtain relative parameters during our calibration as Table 3.

3.3 Forward kinematic

Systematically, forward kinematic challenge could be solved by defining Denavit-Hartenberg (D-H) parameters as Table 4. It is a method to find homogenous transformation matrices among serial links of robot arm as [40]. d_{tool} is the distance

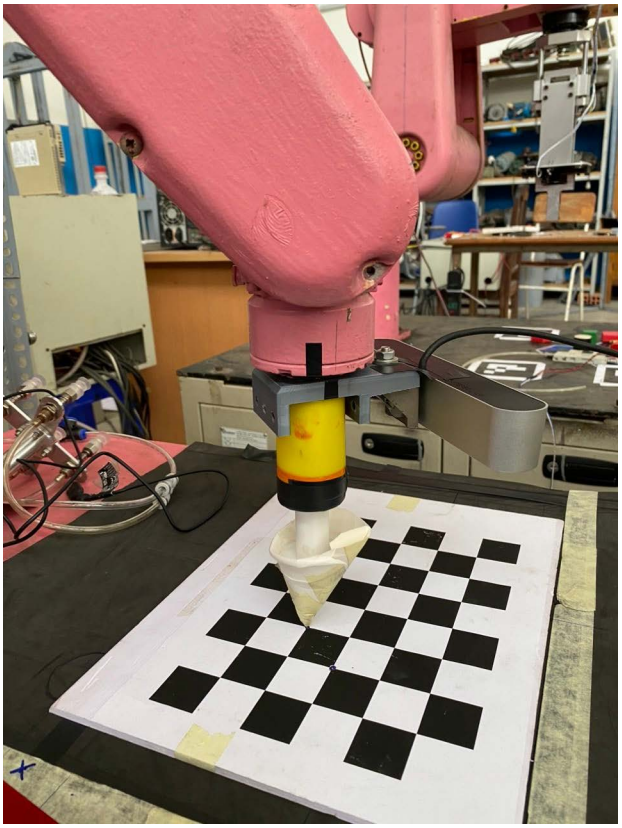

Fig 5. Demonstration of the first step for points taking process.

https://doi.org/10.1371/journal.pone.0323045.g005

from the center of 4th joint to the center of our gripper as Fig 7. Owing to this design, the homogenous transformation matrix ${}^{i-1}_{i}T$: ca is generally well-defined as below

$$ {}^{i-1}_{i}T = \begin{bmatrix} \cos\theta_i & -sin\theta_i \cos\alpha_i & \sin\theta_i \sin\alpha_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\theta_i \cos\alpha_i & -cos\theta_i \sin\alpha_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{14} $$

$$ {}^{0}_{1}T = \begin{bmatrix} \cos\theta_1 & 0 & -sin\theta_1 & 50cos\theta_1 \\ \sin\theta_1 & 0 & \cos\theta_1 & 50sin\theta_1 \\ 0 & -1 & 0 & 358,5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{15} $$

$$ {}^{1}_{2}T = \begin{bmatrix} \cos\theta_2 & -sin\theta_2 & 0 & 300cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & 0 & 300sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{16} $$

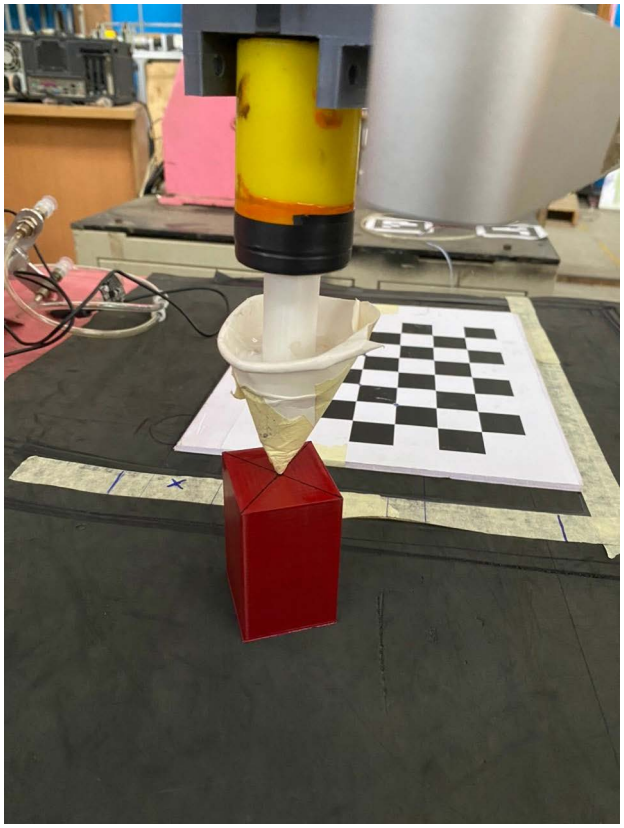

Fig 6. Demonstration of the second step for points taking process.

https://doi.org/10.1371/journal.pone.0323045.g006

Table 2. Results of measurements in relative parameters.

Robot Coordinate (X, Y, Z)	(360,150,183)	(357,207,183)	(387,120,183)	(340, −65,240)
Camera Coordinate (x, y, z)	(178,185,465)	(134,186,465)	(200,163,465)	(355,200,400)

https://doi.org/10.1371/journal.pone.0323045.t002

$$ {}^{2}_{3}T = \begin{bmatrix} \cos\theta_3 & -sin\theta_3 & 0 & 250cos\theta_3 \\ \sin\theta_3 & \cos\theta_3 & 0 & 250sin\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{17} $$

$$ {}^{3}_{4}T = \begin{bmatrix} \cos\theta_4 & 0 & -sin\theta_4 & 0 \\ \sin\theta_4 & 0 & \cos\theta_4 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{18} $$

$$ {}^{4}_{5}T = \begin{bmatrix} \cos\theta_5 & -sin\theta_5 & 0 & 0 \\ \sin\theta_5 & \cos\theta_5 & 0 & 0 \\ 0 & 0 & 1 & d_{tool} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{19} $$

Table 3. Results of relative parameters in the process of camera calibration.

Experiment coordinate x	Experiment coordinate y	Robot x coordinate	Robot y coordinate	Displacement x	Displacement y
329	98	325	93	1.24%	5.38%
355	117	345	112	2.9%	4.47%
290	163	304	165	3.34%	1.22%
361	226	353	225	0.84%	0.45%
301	163	310	165	2.91%	1.22%
423	60	435	55	0.48%	9.1%
411	117	421	110	2.15%	6.37%
400	107	415	100	3.62%	7%
431	227	426	220	1.18%	3.19%
250	97	246	100	1.63%	3%
272	111	264	115	0.75%	3.48%
260	218	248	225	1.97%	3.12%
305	241	295	245	3.39%	1.64%
250	97	239	100	1.24%	5.38%
Overall				2.04%	3.82%

https://doi.org/10.1371/journal.pone.0323045.t003

Table 4. D-H table for robotic parameters.

Joint	d_i (*mm*)	a_i (*mm*)	α_i (*rad*)	θ_i (*rad*)
1	358,5	50	$-\frac{\pi}{2}$	θ_1
2	0	300	0	θ_2
3	0	250	0	θ_3
4	0	0	$-\frac{\pi}{2}$	θ_4
5	$d_S = d_{tool}$	0	0	θ_5

https://doi.org/10.1371/journal.pone.0323045.t004

Thus, the tool coordinate is computed as

$$ {}^{0}_{5}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T\,{}^{4}_{5}T = \begin{bmatrix} R_{11} & R_{12} & R_{13} & P_x \\ R_{21} & R_{22} & R_{23} & P_y \\ R_{31} & R_{32} & R_{33} & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{20} $$

where

$R_{11} = cos\,(\theta_4 + \theta_3 + \theta_2)\cos(\theta_1)\cos(\theta_5) + sin\,(\theta_1)\sin(\theta_5)$

$R_{12} = -cos\,(\theta_4 + \theta_3 + \theta_2)\cos(\theta_1)\cos(\theta_1)\cos(\theta_5) + sin\,(\theta_1)\cos(\theta_5)$

$R_{13} = -sin\,(\theta_4 + \theta_3 + \theta_2)\cos(\theta_1)$

$R_{21} = cos\,(\theta_4 + \theta_3 + \theta_2)\sin(\theta_1)\cos(\theta_5) - cos\,(\theta_1)\sin(\theta_5)$

$R_{22} = -cos\,(\theta_4 + \theta_3 + \theta_2)\sin(\theta_1)\sin(\theta_5) - cos\,(\theta_1)\cos(\theta_5)$

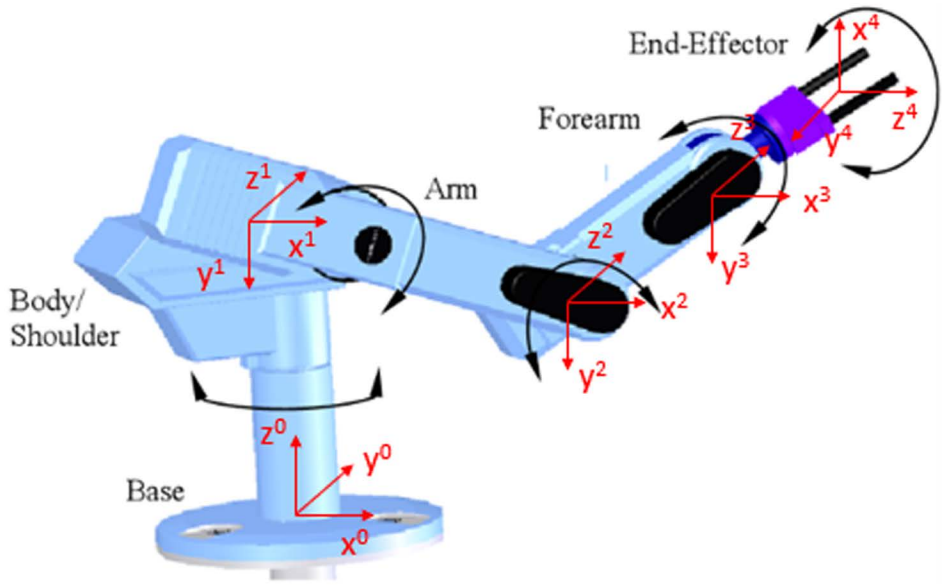

Fig 7. Modeling of joint-coordinate for our manipulator.

https://doi.org/10.1371/journal.pone.0323045.g007

$$R_{23} = -sin\,(\theta_4 + \theta_3 + \theta_2)\,\sin\,(\theta_1)$$

$$R_{31} = -sin\,(\theta_4 + \theta_3 + \theta_2)\,\cos\,(\theta_5)$$

$$R_{32} = sin\,(\theta_4 + \theta_3 + \theta_2)\,\sin\,(\theta_5)$$

$$R_{33} = -cos\,(\theta_4 + \theta_3 + \theta_2)$$

$$P_x = cos\,(\theta_1)\,[250cos\,(\theta_3 + \theta_2) + 300cos\,(\theta_2) - d_{tool}\,\sin\,(\theta_4 + \theta_3 + \theta_2) + 50]$$

$$P_y = sin\,(\theta_1)\,[250cos\,(\theta_3 + \theta_2) + 300cos\,(\theta_2) - d_{tool}\,\sin\,(\theta_4 + \theta_3 + \theta_2) + 50]$$

$$P_z = -300sin\,(\theta_2) - 250sin\,(\theta_3 + \theta_2) - d_{tool}\,\cos\,(\theta_4 + \theta_3 + \theta_2) + 358,5$$

3.4 Inverse kinematic

Target solution of inverse kinematic is to find the potential configurations of the set of joint angles θ of this robotic platform when the end-effector towards to destination

$$\theta = \begin{bmatrix}\theta_1 & \theta_2 & \theta_3 & \theta_4 & \theta_5\end{bmatrix} \tag{21}$$

In this stage, all values of homogenous transformation matrix in equation (8) are given. By solving trigonometric equation, we have

$$\frac{P_y}{P_x} = \frac{\sin\theta_1}{\cos\theta_1} = tan\theta_1$$

$$\rightarrow \theta_1 = \tan^{-1}\frac{P_y}{P_x} \tag{22}$$

$$\frac{R_{32}}{R_{31}} = -\frac{\sin\theta_5}{\cos\theta_5} = -tan\theta_5$$

$$\rightarrow \theta_5 = -\tan^{-1}\frac{R_{32}}{R_{31}} \tag{23}$$

$$And\ \theta_2 + \theta_3 + \theta_4 = \cos^{-1} R_{33} \tag{24}$$

$$P_x\cos\theta_1 + P_y\sin\theta_1 = 250cos\left(\theta_2 + \theta_3\right) + 300cos\theta_2 - d_{tool}\sin\left(\theta_2 + \theta_3 + \theta_4\right) + 50 \tag{25}$$

Hence,

$$\begin{cases} 250cos\left(\theta_2 + \theta_3\right) + 300cos\theta_2 = P_x\cos\theta_1 + P_y\sin\theta_1 \\ +d_{tool}\sin\left(\theta_2 + \theta_3 + \theta_4\right) - 50 \\ 250sin\left(\theta_2 + \theta_3\right) + 300sin\theta_2 = -P_z + 358,5 \\ -d_{tool}\sin\left(\theta_2 + \theta_3 + \theta_4\right) \end{cases} \tag{26}$$

Let

$$A_e = P_x\cos\theta_1 + P_y\sin\theta_1 + d_{tool}\sin\left(\theta_2 + \theta_3 + \theta_4\right) - 50 \tag{27}$$

$$B_e = -P_z + 358,5 - d_{tool}\sin\left(\theta_2 + \theta_3 + \theta_4\right) \tag{28}$$

Then, equation (14) becomes

$$\begin{cases} 300cos\theta_2 - A_e = -250cos\left(\theta_2 + \theta_3\right) \\ 300sin\theta_2 - B_e = -250sin\left(\theta_2 + \theta_3\right) \end{cases} \tag{29}$$

$$\rightarrow \frac{A_e}{\sqrt{A_e^2 + B_e^2}}cos\theta_2 + \frac{B_e}{\sqrt{A_e^2 + B_e^2}}sin\theta_2 = \frac{A_e^2 + B_e^2 - 27500}{600} \tag{30}$$

We get

$$\sin\theta_e = \frac{A_e}{\sqrt{A_e^2 + B_e^2}} \tag{31}$$

$$\cos\theta_e = \frac{B_e}{\sqrt{A_e^2 + B_e^2}} \tag{32}$$

$$\rightarrow \theta_2 = \sin^{-1}\left(\frac{A_e^2 + B_e^2 - 27500}{600\sqrt{A_e^2 + B_e^2}}\right) - \theta_e \tag{33}$$

$$\rightarrow \theta_3 = \cos^{-1}\left(\frac{A_e - 300cos\theta_2}{250}\right) - \theta_2 \tag{34}$$

In case that there are multiple inverse kinematic results, unique solution must be voted according to these constraints on the joint limit angle.

3.5 Camera installation

The set-up configuration of vision system is eye2hand method. It consists of two steel bars, metal load and base. Since it looks like console beam, there is a counter-load in the other end of bar while digital camera is hung in one end. This configuration ensures global coverage, stable image.

To identify the dimension of camera standee, some geometric constraints should be considered. With vertical FOV (Field of View), it must cover the working space of robot such $(z \times x \times y) : (400 \times 800 \times 800)(mm)$. The potential dimension could be estimated as

$$h \geq \frac{800}{2} \times cotan\left(\frac{60^o}{2}\right) + 400 \approx 1093(mm) \tag{35}$$

$$x \geq h \times tan\left(\frac{60^o}{2}\right) + 100 \approx 740(mm) \tag{36}$$

In addition, force/moment is analyzed to balance without tipping over. The mass of counter-load must satisfy below condition as

$$\begin{aligned} &\sum_i M_i^B = 0 \\ &\leftrightarrow F_b \times L_2 + F_c \times L_3 - F_w \times L_1 = 0 \\ &\leftrightarrow m_c = \frac{F_w \times L_1 - F_b \times L_2}{g \times L_3} \end{aligned} \tag{37}$$

where

F_c: force by weight of counter-load

F_b: force by weight of steel bar. It locates at the center of bar

F_w: force by weight of digital camera

4 The proposed approach

In this investigation, the proposed design as Fig 8 comprises four components such workspace data capture, motion planner, profile generation and robotic platform. Objects in the working space are captured by a powerful camera and sent to host computer. Later, data is extracted to achieve necessary spatial information of start, goal, and obstacles. Path planner sketches out the candidate traveling routes and elect the shortening collision-free one. Then, motion data is exchanged and interpolated using inverse kinematic procedure. Additionally, for each link, various profiles are generated to ensure the smooth motion in time-space domain. Our design is firstly validated to obtain reference path in Matlab software, consequently command data is input to drive mechanical robot. Whole system could be supervised by an operator via digital camera.

4.1 General concept

Main program which is embedded into microprocessor, involves three fundamental steps. Initially, the surrounding context that could be captured by digital camera, plays as input of sub-program. Secondly, motion planner is triggered to elect the shortening collision-free route in RRT path planning sub-program. Thirdly, driving command is performed in the level of motion controller.

From the visual data, it is fed to YOLO algorithm for object detection involved in path planning. It returns the type of objects, 2D bounding boxes and its centroid pixel coordinate of each box. Moreover, software development kit (SDK) [41]

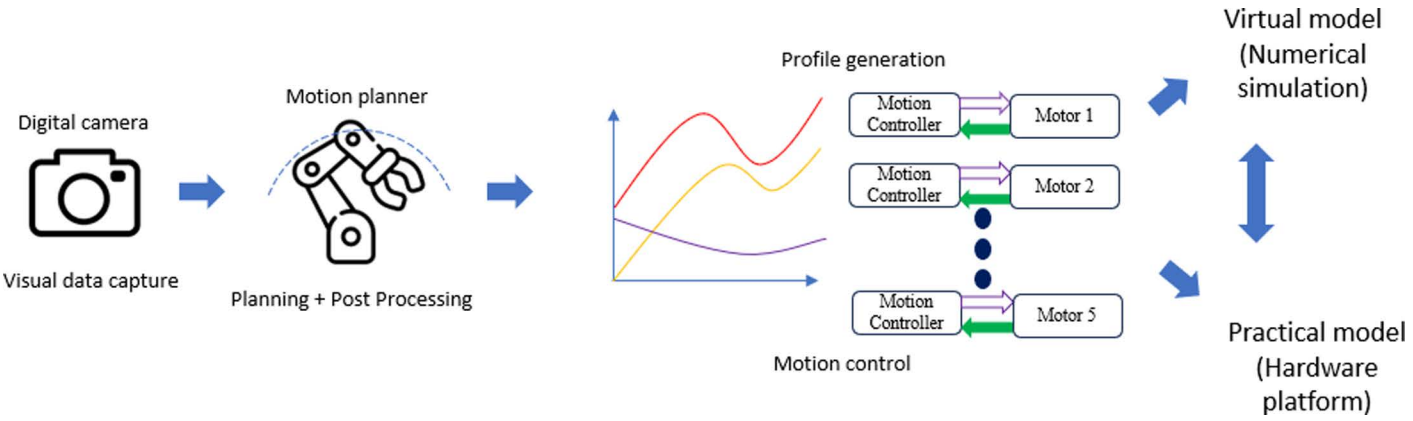

Fig 8. Overall scheme for our development.

https://doi.org/10.1371/journal.pone.0323045.g008

of camera is utilized to get 3D coordinates of multiple point-clouds. Subsequently, it is multiplied with extrinsic matrix to transform these 3D coordinates to the robot basement. According to the type of detected objects as above, the occupied volume in workspace could be determined via its shape and centroid coordinate.

To lessen the heavy computations, robotic platform requires to simplify its links into a simpler geometric model such as ellipsoids [42], cylinders [43], and spheres [44]. In this work, our concept is to deploy a model-based capsule for stimulating the robotic links. This approach is not only well fitted in the shape of robotic arm but also deliver the efficient computations. Due to this design, each solid segment represents the skeleton of robotic manipulator. Similarly, an object or obstacle is also modeled as the shaded capsules and the distance among capsules of the closest pair is very sensitive to avoid collision. We assume that robotic platform at the state $q \in \mathbb{R}^N$ occupies its workspace in the Cartesian space $C(q) \in \mathbb{R}^3$. Then, an object with its state x_0 also occupies an area in the environment $\Theta(x_0) \in \mathbb{R}^3$. It is supposed that the Euclidean distance vector from position of robot $\mathcal{P}_{r0}$ to position of object $\mathcal{P}_{ob}$: $\mathbb{R}^3 \times \mathbb{R}^3 \rightarrow \mathbb{R}^3$, consequently the norm of relative distance vector between robot and object is identified as

$$\| d_{r0}(q, x_0) \| = min \| d_i(\mathcal{P}_{r0}, \mathcal{P}_{ob^i}) \| \mathcal{P}_{r0} \in C(q), \mathcal{P}_{ob^i} \in \Theta(x_0) \tag{38}$$

In Fig 9, it is denoted that robotic links are represented as geometric model with different dimensions, for instance long or short capsule, indicating hardware consumption in workplace. Thus, our concept of capsule-like SCORBOT is proposed as Fig 10. This capsule shape comprises a cylinder with two semi-spherical ends, has the same width as its physical size of link. Henceforth, value d_{ro} becomes distance between two axes. Our robotic platform is modeled as three bounding volumes since they have the highest probability of collision when robot is moving.

4.2 Path planner

The core idea of our path planning is to randomly generate a tree in the searching space, and maintain the Euclidian distance among nodes less than or equal to step size. If a new node might be in collision, it should be ignored and get another new node. This scheme is terminated if the goal is reached, and then the path planning is obtained. Due to our design, the proposed path planner is illustrated as Fig 11. From the start and goal, the step of inverse kinematic returns a random configuration $q_{rand} = [q_1, q_2, q_3, q_4, q_5]$. Goal bias factor α is defined as upper limit to improve the planning convergence in faster and higher success rate. If the random probability is equal or less than its value, q_{rand} would be the goal configuration. Otherwise, it would be random configuration in range of $-\pi$ and π. In fact, this factor α should be experimentally tuned based on a trade-off between exploration and convergence. A higher α speeds up

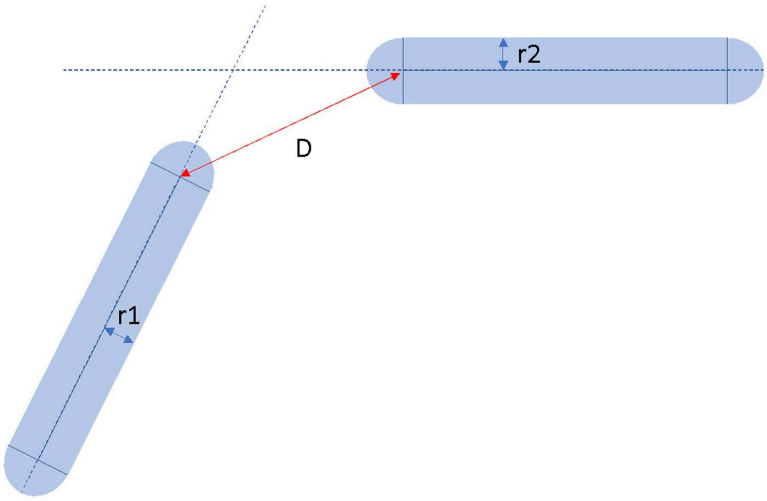

Fig 9. Estimation of distance between two capsules.

https://doi.org/10.1371/journal.pone.0323045.g009

Fig 10. Illustration of capsule-like robot in our research.

https://doi.org/10.1371/journal.pone.0323045.g010

convergence but increases the risk of local minima, while a lower α ensures better exploration but slows convergence. In our best knowledge, the proposed algorithm achieves a 30% faster convergence rate in simple scenarios when α equals to 0.3. In more complicated environments, our algorithm ensures higher success rates if the value of this factor α is 0.6.

After obtaining q_{rand}, the nearest configuration would be found and the Euclidian distance d_{ro} between q_{rand} and nearest one should be computed. If d_{ro} is greater than step size, q_{new} becomes the nearest one plus q_{rand} configuration with step size length. Or else, new configuration would be q_{rand} immediately. After that, this configuration would be checked for joint limit and collision conditions.

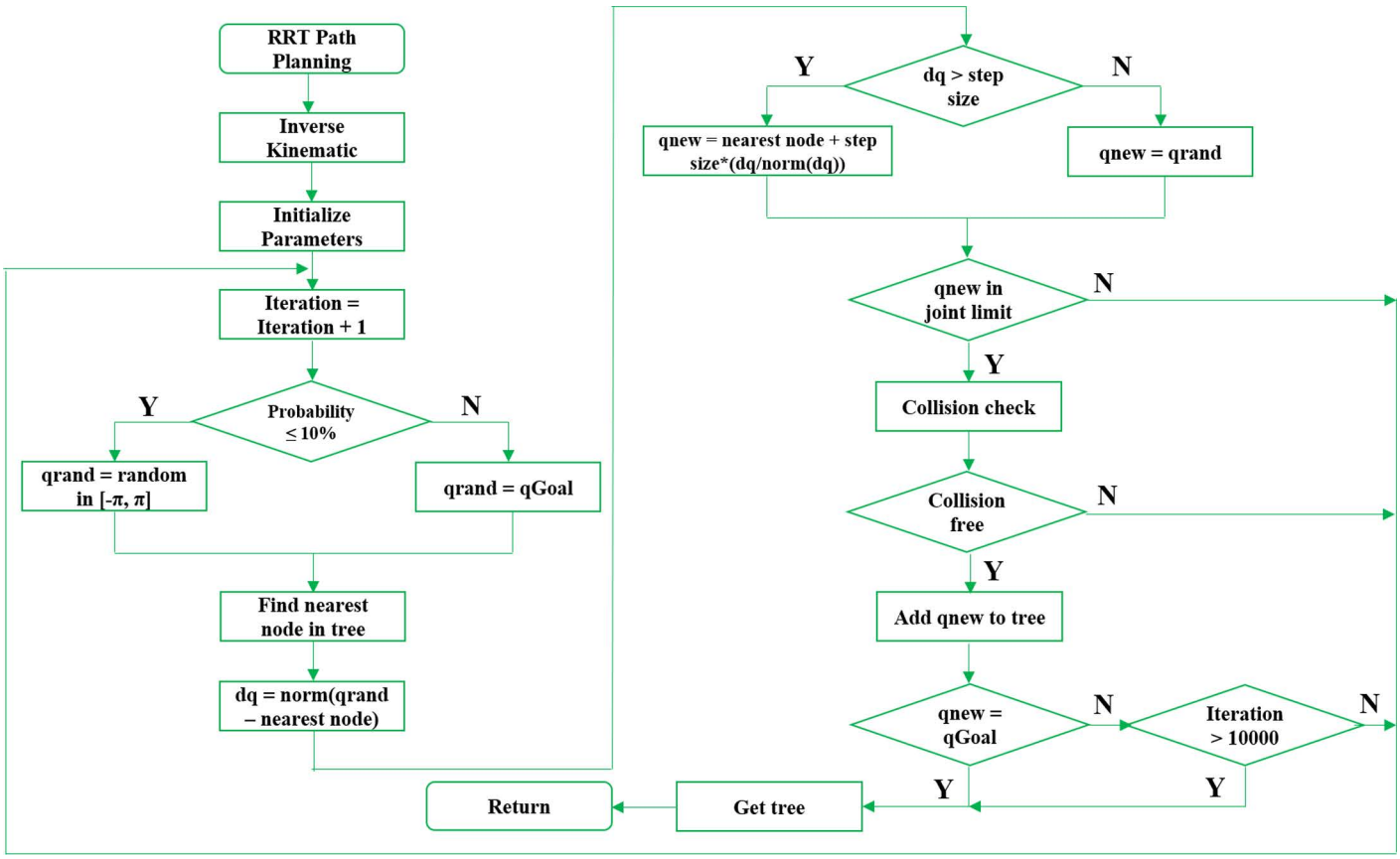

Fig 11. The proposed path planning scheme for SCORBOT-ER-VII.

https://doi.org/10.1371/journal.pone.0323045.g011

4.3 Path post processing

In fact, path planning by traditional RRT is not optimal since it has many redundant nodes and sharp turning point as Fig 12. This path is not proper for driving manipulator in term of traveling distance and smoothness. The modified RRT path planning scheme must satisfy two requirements such that reducing path nodes are free collision and sudden change should be avoided for reliably mechanical system and safely human interaction.

Our motivation to shorten traveling path is to assign a temporary configuration q_{temp} as $q[0]$ at start, and then connect $q[0]$ with $q[i]$, $i = 1, 2, \ldots N$ checked for potential collision sequentially. If there is no collision, the validation for next configuration should be kept without changing q_{temp}. Else, changing q_{temp} to previous configuration $q[i-1]$, add q_{temp} to the reduced tree, and do the same collision-checking procedure for the next configuration $q[i+1]$. It would be ended when the checking process reaches to the last configuration as Fig 13.

To ensure the smoothness of traveling path, cubic spline interpolation should be deployed. There are two reasons such that the spline curve produces the continuation of the acceleration vector and preserves the sufficient smoothness even in the presence of small curvatures. In Fig 14, this technique is implemented into RRT-CS scheme so that (i) the number of nodes is two, it would process linear interpolation for simplicity, and (ii) node number is larger than two, it fits several cubic spline interpolation curves through those nodes with the same pre-defined traveling time.

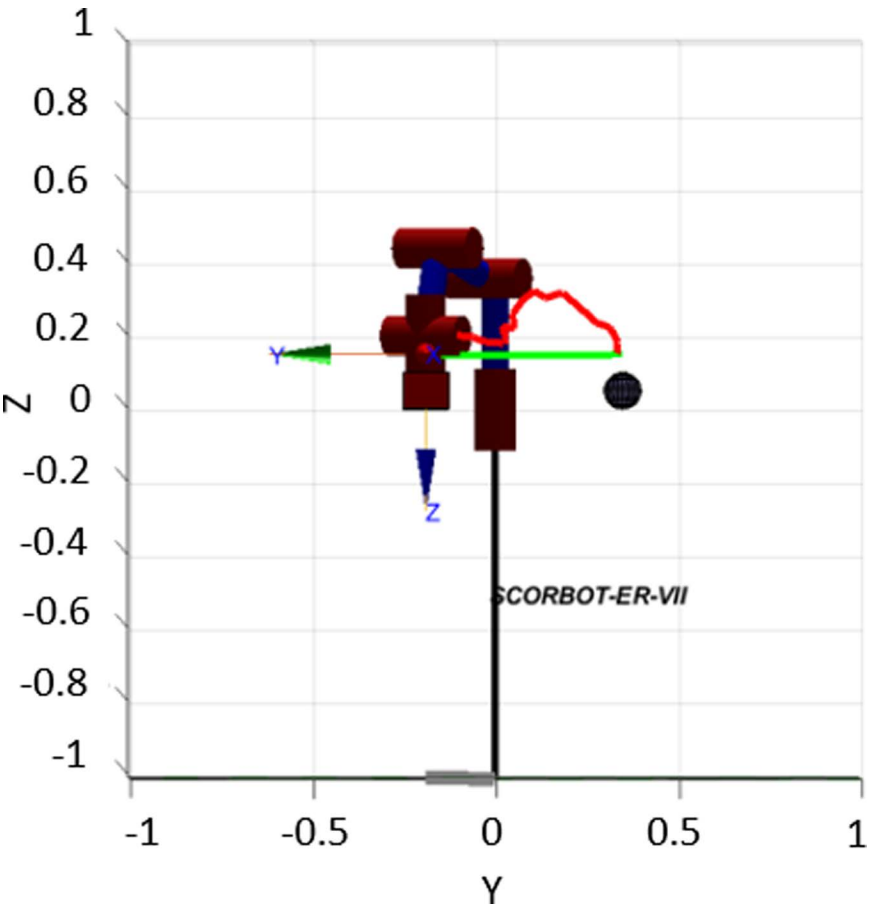

Fig 12. Description of traditional RRT path (red color) and post processed RRT path (green color).

https://doi.org/10.1371/journal.pone.0323045.g012

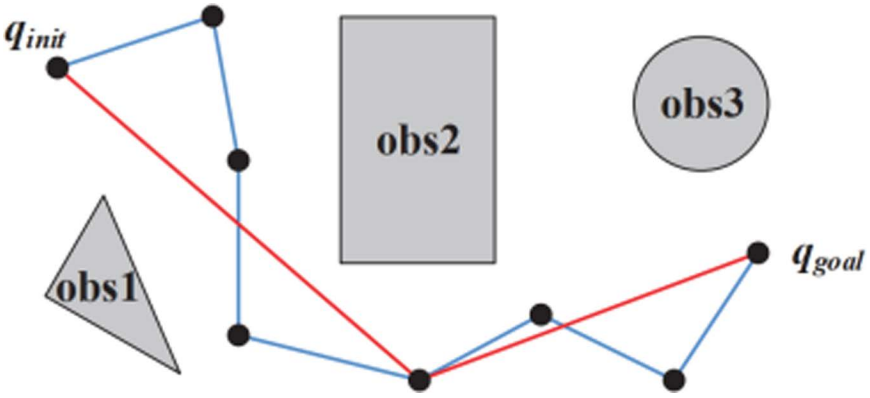

Fig 13. Description of traditional RRT path (green color) and path shortening technique (red color).

https://doi.org/10.1371/journal.pone.0323045.g013

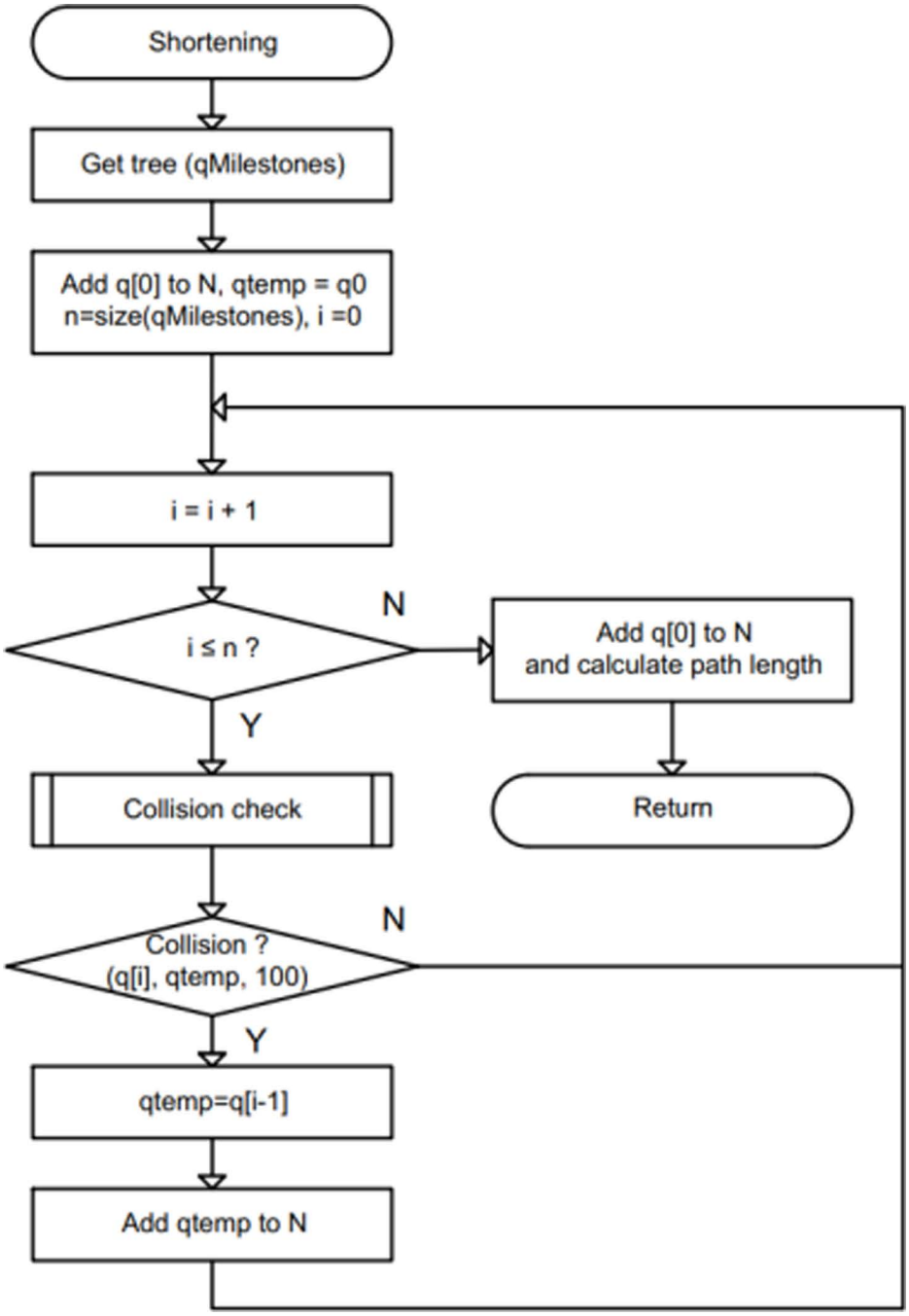

Fig 14. Description of traditional RRT path (red color) and post processed RRT path (green color) (a) and path shortening technique (b).

https://doi.org/10.1371/journal.pone.0323045.g014

5 Simulations and experiments

To prove the effectiveness and feasibility of our approach, some test scenarios have been launched as Fig 15. In front of robotic manipulator, there is a bar console to hang one digital camera. The advantages of eye-to-hand camera are to deliver global coverage, stable image and not hinder the motion of robot. Both simulation and experiment as Fig 16 are

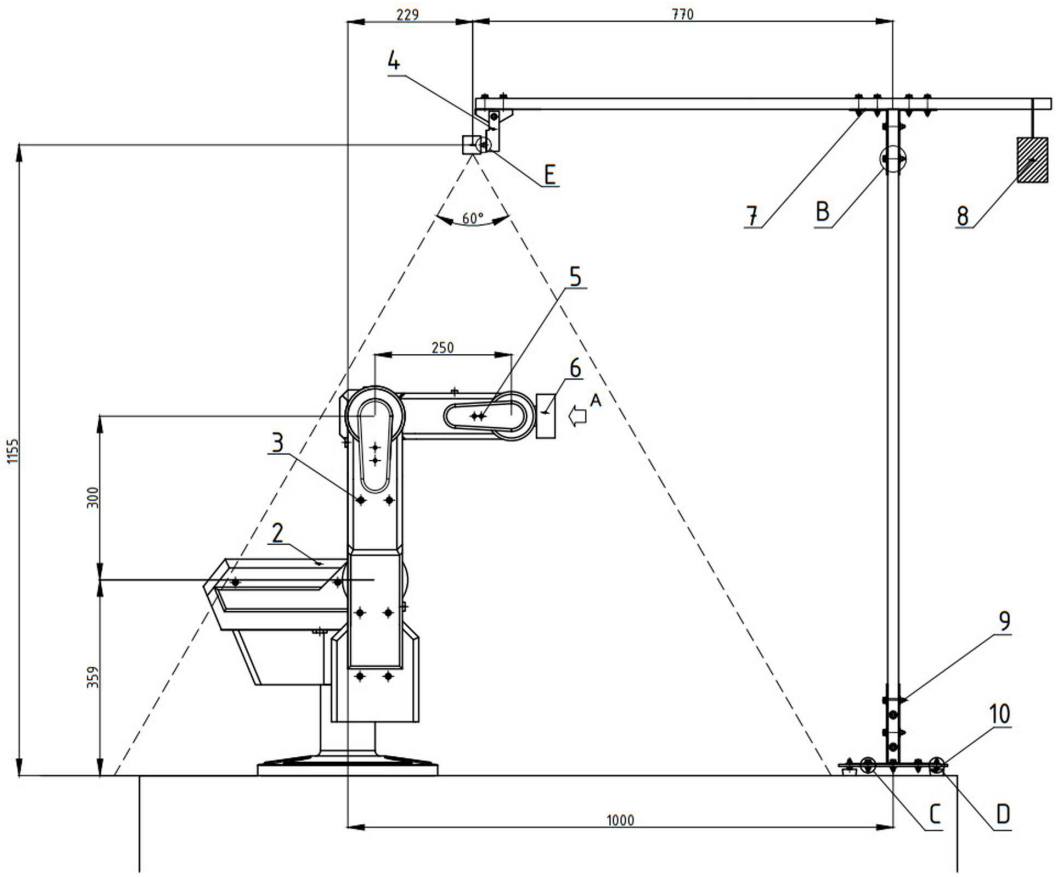

Fig 15. Layout setup of overall model.

https://doi.org/10.1371/journal.pone.0323045.g015

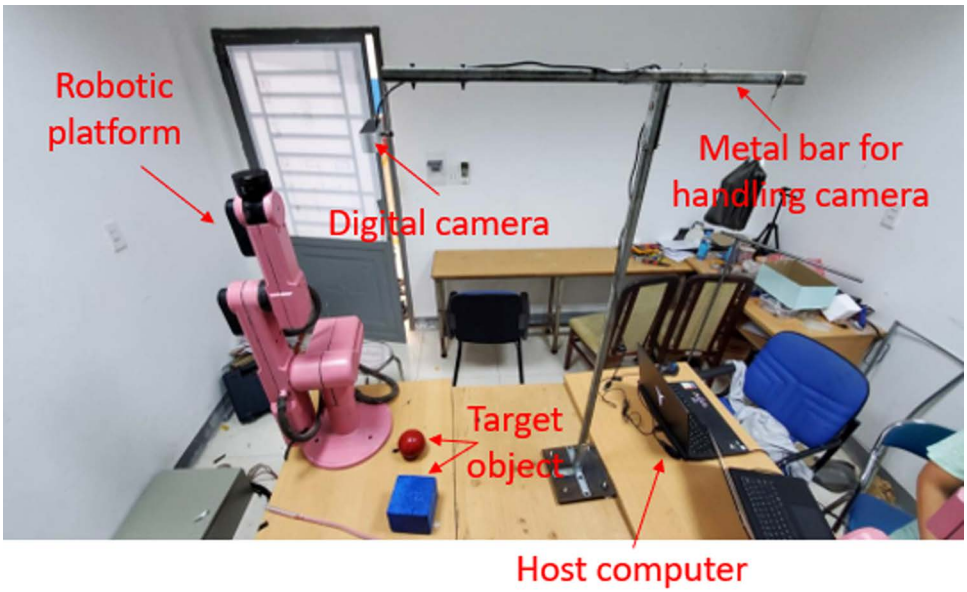

Fig 16. Layout setup of experimental scenario.

https://doi.org/10.1371/journal.pone.0323045.g016

entirely executed on computer with 4.5GHz CPU Intel i7-6700HQ, 16GB RAM, card GPU NVIDIA GTX960M and Windows OS 11.

Our test scene is inspired by the real-world circumstances, i.e., unknown environment which robot was not taught before. Additionally, the experimental validation of this robotic platform is proceeded in both daylight and night. In these tests, three different scenarios from free-obstacle to hard obstacle are illustrated. The first case as Fig 17 is the simplest one when manipulator only generates motion trajectory from red cube to red sphere, and there is no obstacle. In the second scenario as Fig 18, two yellow rectangular boxes are placed between cube and sphere. Our robotic manipulator must compute more flexible traveling path to avoid yellow boxes. In last case as Fig 19, the number of obstacles is more and its height is greater. To overcome these yellow boxes, robot must try its best to travel smoothly and stably.

In our research, most of validations should be evaluated by three criteria such as completeness, path quality and timing characteristics. In the view of completeness, the success rate is to identify the ability of searching the superior result if possible. Otherwise, it returns failure in finite time. Consequently, the second criterion measures path length and smoothness of trajectory. In addition, traveling time is also important to estimate how good planner is. This criterion is to assess whether motion planner is proper in the real-time applications. An offline planner would capture the environmental situation once, then it generates a path in advance. During the motion execution, it does not need to re-generate traveling trajectory until it ends. Reversely, a real-time planner must update the system state continuously and produce an up-to-date trajectory with very short time. It does not require the intervention of an operator. In the dynamic environment, this criterion becomes a very critical factor for the autonomous system.

In the sampling-based planner, the completeness of path planning means the probability of searching convergence to 1 when the number of iterations tends to infinity. For our works, each test case is validated during 50 times and comparative results are shown as Table 5. Test case 3 has the lowest number of successes runs because more obstacles are added.

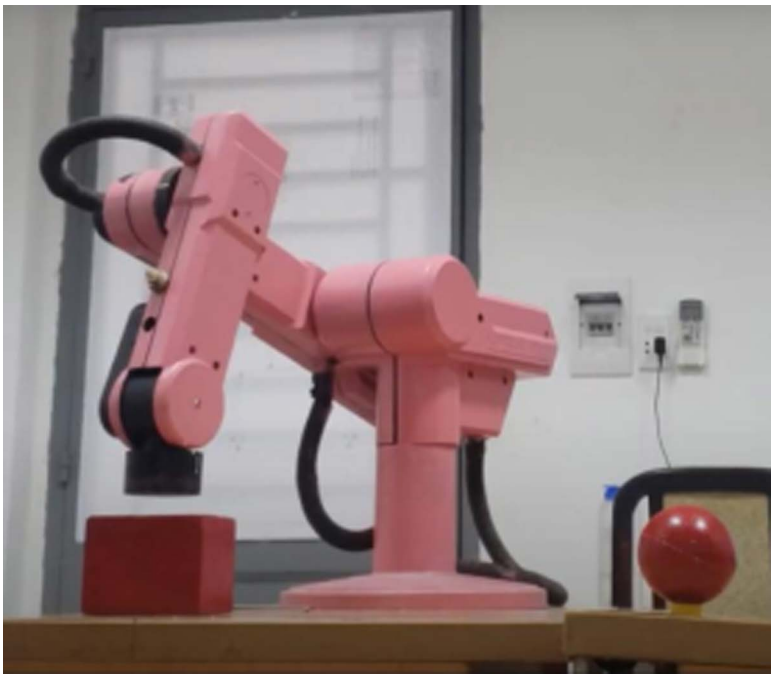

Fig 17. Description of the first test scenario.

https://doi.org/10.1371/journal.pone.0323045.g017

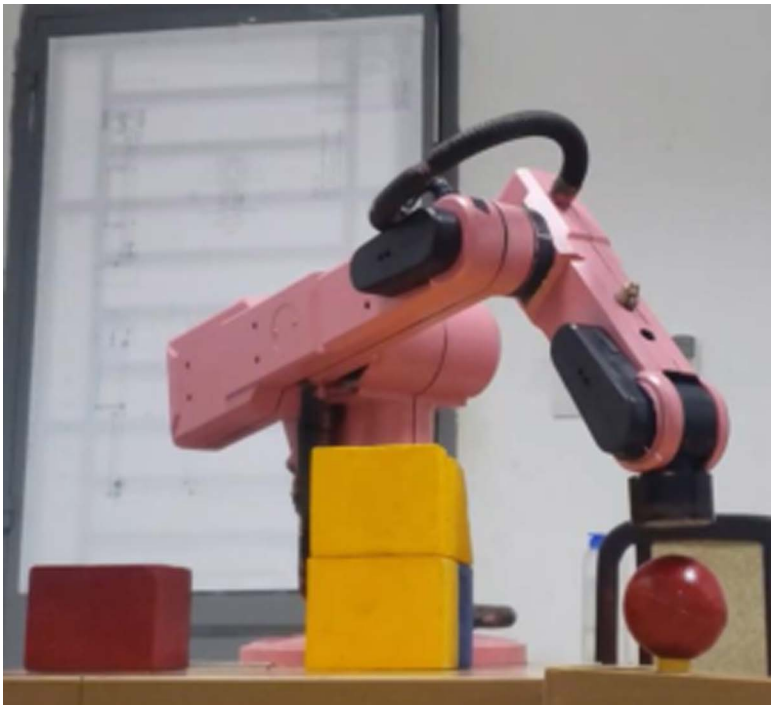

Fig 18. Description of the second test scenario.

https://doi.org/10.1371/journal.pone.0323045.g018

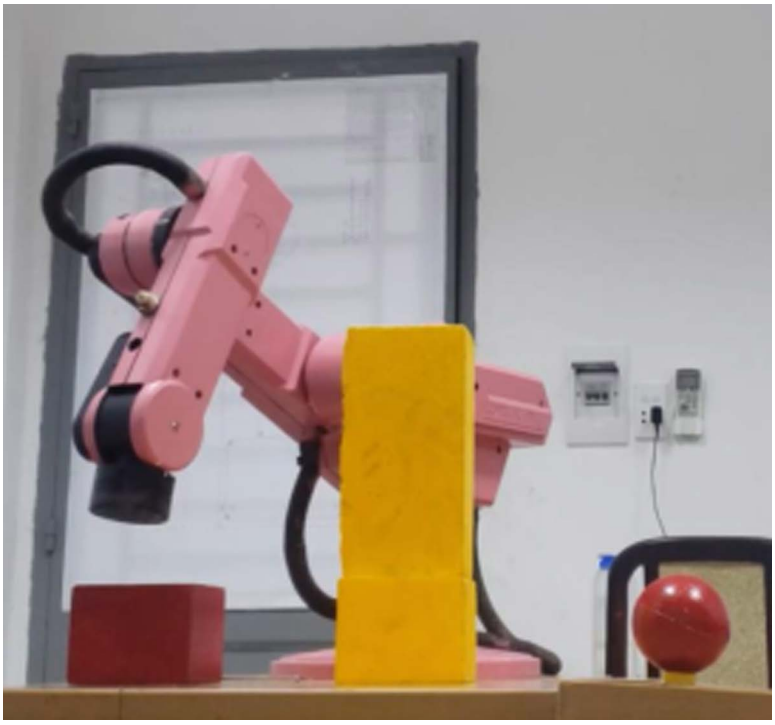

Fig 19. Description of the third test scenario.

https://doi.org/10.1371/journal.pone.0323045.g019

Table 5. Results of evaluation in completeness.

Item	Case 1	Case 2	Case 3
Average of iterations	131	1066	1241
Number of success runs/Total runs	50/50	43/50	35/50
Average of success rate	100%	86%	70%

https://doi.org/10.1371/journal.pone.0323045.t005

The first test case is perfect to plan traveling route without any obstacle. In test case 2, not all runs found a good solution and robot still makes an effort to drive.

In Fig 20, path planning result by traditional RRT (red color) is still not optimal because of the random characteristics of motion planner. After using our method, traveling trajectory (green color) is re-constructed and satisfies our requirements. Fig 21 demonstrates the experimental result when our robotic hardware is used. For more details, traveling distance as Fig 22 by the proposed approach is shorter than traditional RRT, approximately 36%. Furthermore, it can be seen visually that in the test case 1, our algorithm releases the smoother curve for each joint as Fig 23 and Fig 24. These results might help to reduce any vibration in the robotic mechanism when it moves rapidly.

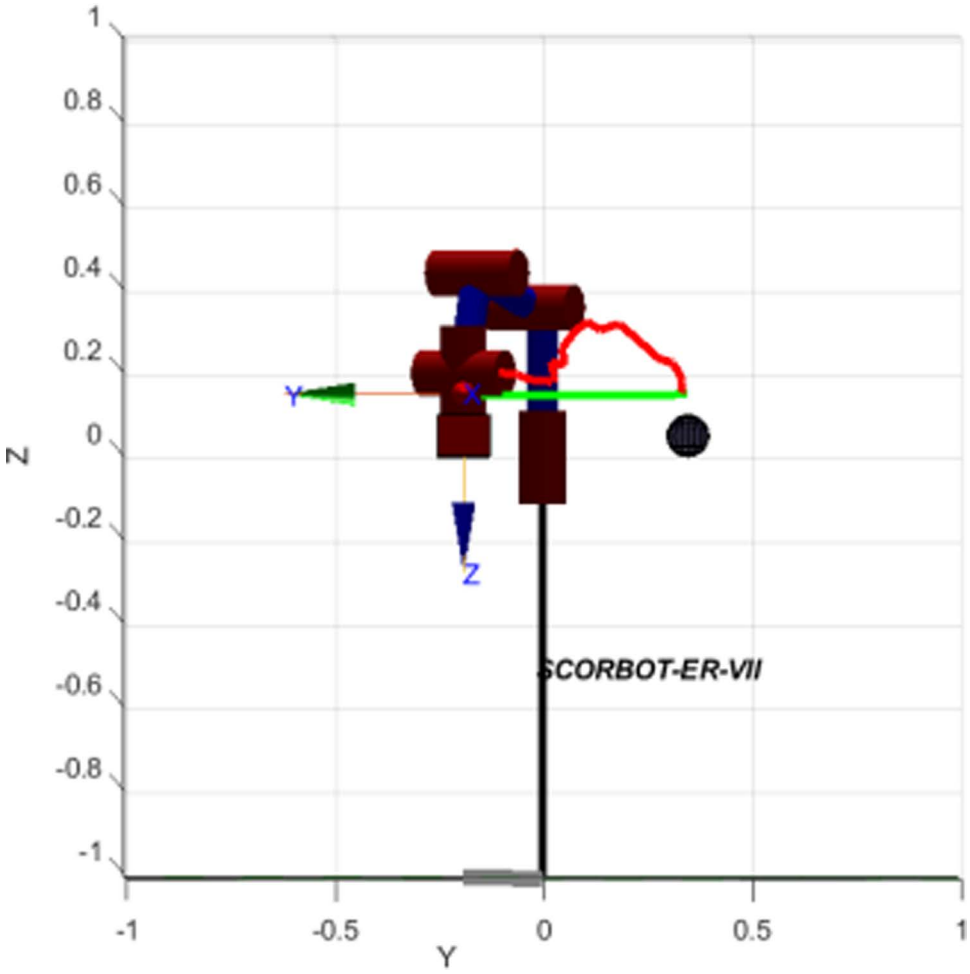

Fig 20. Simulation result of the proposed system using our motion planning scheme in case 1.

https://doi.org/10.1371/journal.pone.0323045.g020

Fig 21. Experimental result of the proposed system using our motion planning scheme in case 1.

https://doi.org/10.1371/journal.pone.0323045.g021

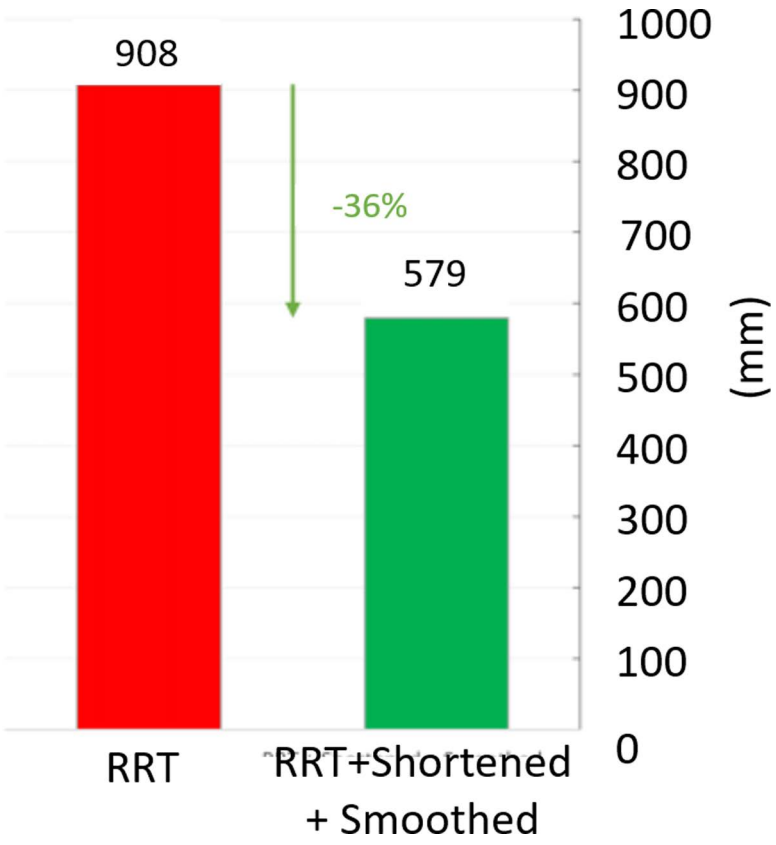

Fig 22. Comparative result of path length between our approach (green) and traditional RRT scheme (red) in case 1.

https://doi.org/10.1371/journal.pone.0323045.g022

Similarly, test case 2 as Fig 25 seems to be more difficult because more obstacles are added. Theoretically speaking as Fig 26 and Fig 27, our method generates better traveling route and ensure collision-free motion. Comparing to the other method, the proposed approach achieves shorter path, roughly 21% as Fig 28 and Fig 29. It also means that robot can reach to destination in the earlier time. Besides, path planning by our scheme as Fig 30 and Fig 31 certifies the smoother curves although there exist more fluctuations owing to complex obstacles. Test case 3 as Fig 32 indicates the

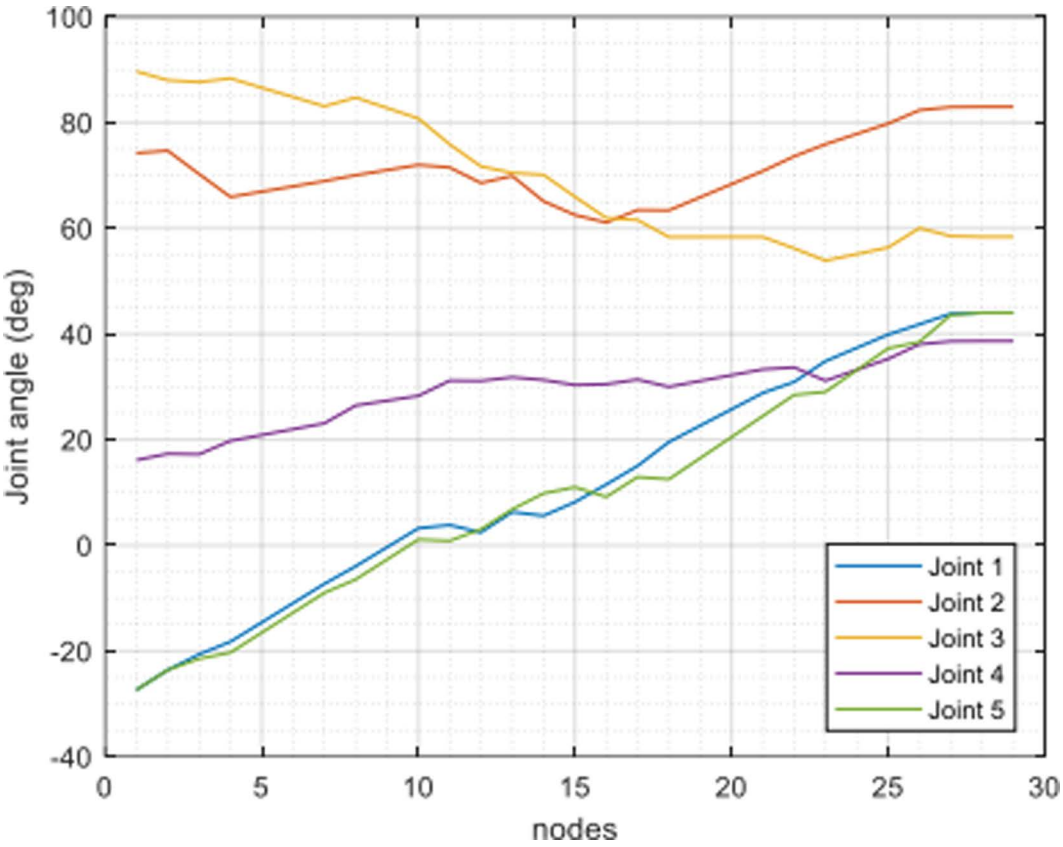

Fig 23. Experimental result of joint angle using traditional RRT scheme in case 1.

https://doi.org/10.1371/journal.pone.0323045.g023

most complicated challenge for the reason that there are three obstacles between start and target point. Consequently, robot must try its best to overcome these obstacles. In Fig 33, host computer handles larger computations to generate more flexible paths. In our comparison, the proposed approach attains 19% shorter traveling route than traditional method. Likewise, the smoothness of motion profile as Fig 34 is reserved while robot warrants free-collision trajectory.

Table 6 describes the comparative time between different schemes. From test case 1 to test case 3, total planning time is increased proportionally in respect to the hardness of trial scenarios. In three schemes, smoothness takes the least time, i.e., less than 10%, and can be ignored if necessary. Shortening procedure spends one third portion of total processing time while RRT profile generation costs two third.

Besides, it is essential to compute the quantitative smoothness metrics to provide a rigorous assessment as Table 7. The first indicator is path curvature (C) which measures the deviation of the trajectory from a straight-line path. If a path is perfectly smooth, the change in angle (θ) between consecutive waypoints should be minimal.

$$C = \frac{1}{N}\sum_{i=1}^{N-1} |\theta_{i+1} - \theta_i| \tag{39}$$

where

N: Total number of waypoints in the trajectory

θ_i: Heading angle of the robot at waypoint i^{th}

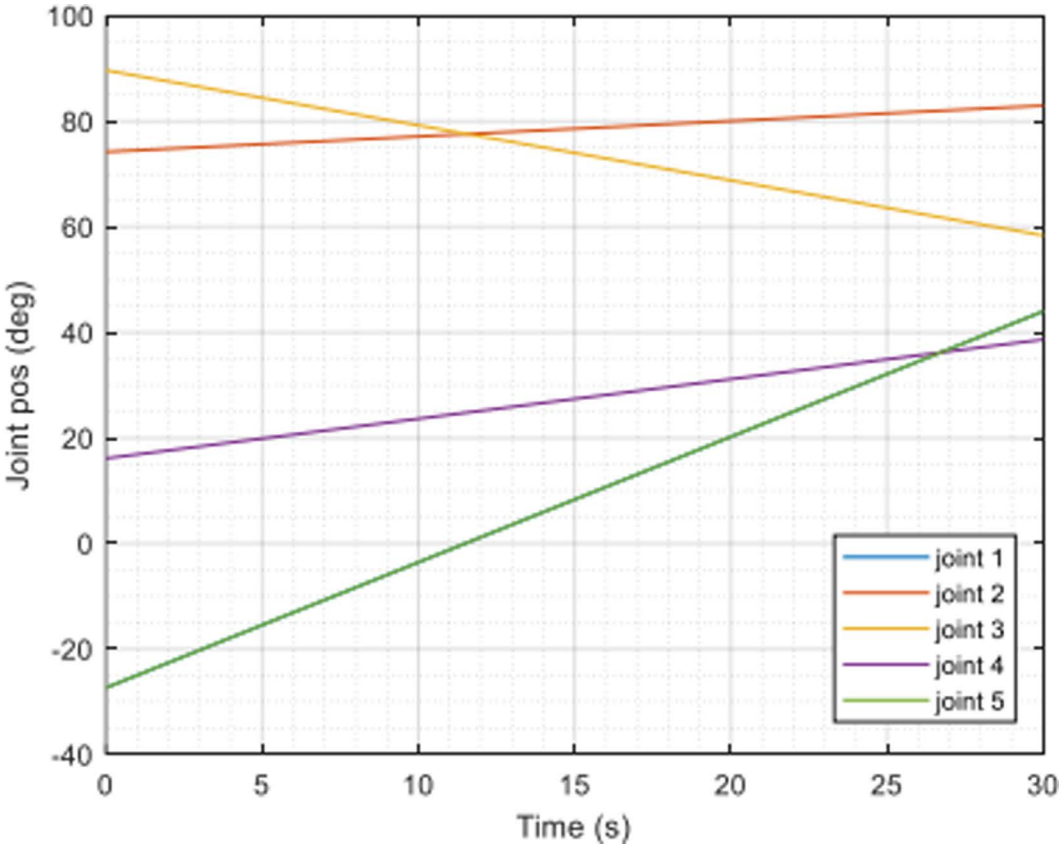

Fig 24. Experimental result of joint angle using our approach in case 1.

https://doi.org/10.1371/journal.pone.0323045.g024

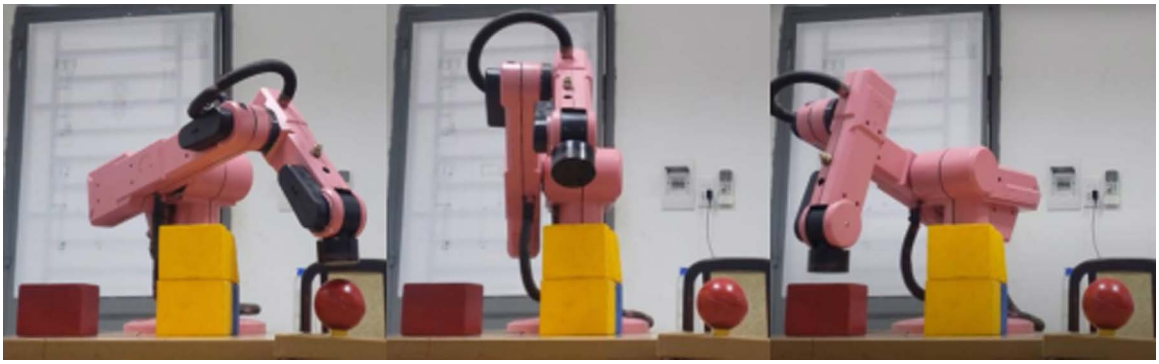

Fig 25. Experimental result of the proposed system using our motion planning scheme in case 2.

https://doi.org/10.1371/journal.pone.0323045.g025

From equation (39), it is well-recognized that higher values of (C) would indicate more abrupt turns and less smooth motion. If $C = 0$, the path is a perfect straight line. If (C) is large, the trajectory has many sharp turns, signifying jerky motion. Secondly, jerk (J) indicator is defined as the rate of change of acceleration. It represents how quickly acceleration varies.

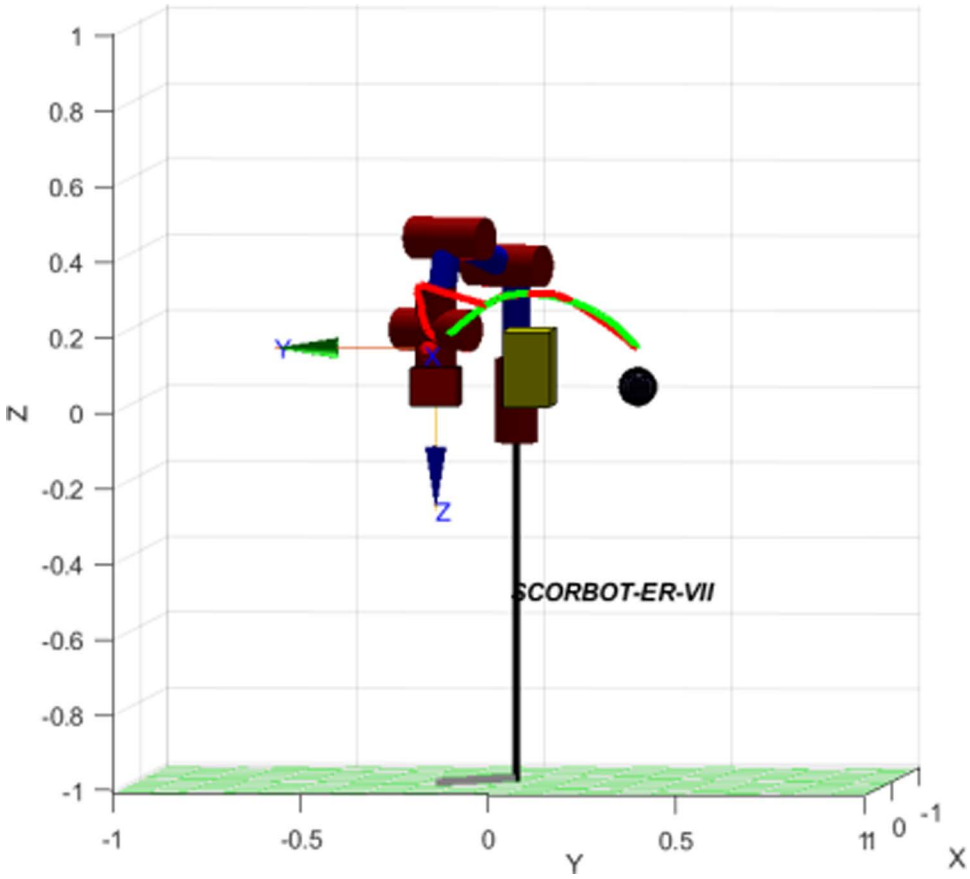

Fig 26. Simulation result of the proposed system using our motion planning scheme in case 2.

https://doi.org/10.1371/journal.pone.0323045.g026

$$J = \frac{1}{T}\sum_{i=1}^{N-1}\left|\frac{d^3x}{dt^3}\right| \tag{40}$$

where

T: Total time duration of motion

x: Position of the robot along the trajectory

Jerk indicator illustrates motion stability. If its value is high, the robot accelerates or decelerates abruptly. At that moment, vibrations could occur in this system. Third indicator is total variation of joint angles ($TVJA$).

$$TVJA = \sum_{j=1}^{M}\sum_{i=1}^{N-1}|q_{i+1}^{j} - q_{i}^{j}| \tag{41}$$

where

M: Number of robot joints

q_i^j: Joint angle of j^{th} at waypoint i^{th}

In equation (41), it measures total changes in joint angles over the trajectory. In the case that high $TVJA$ means joints move erratically, leading to unstable motion. Otherwise, low value of $TVJA$ indicator specifies joint movements are gradual

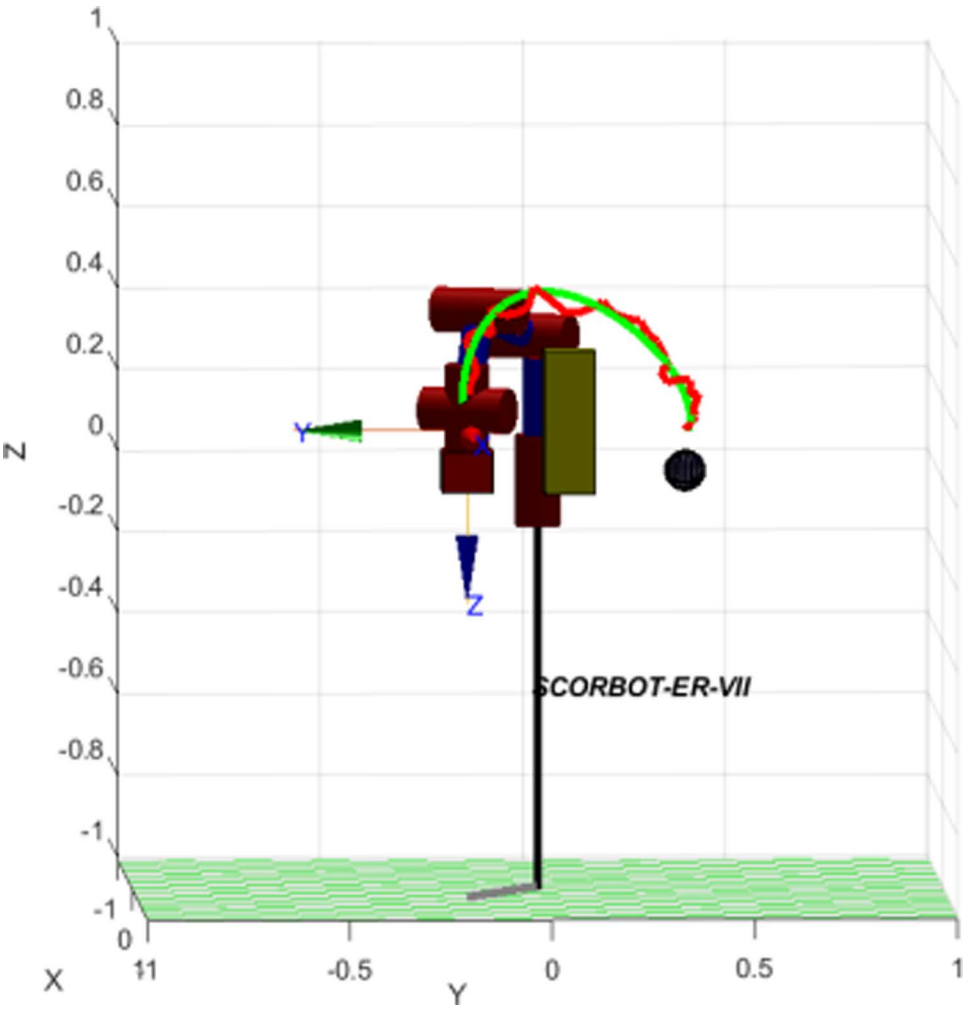

Fig 27. Simulation result of the proposed system using our motion planning scheme in case 3.

https://doi.org/10.1371/journal.pone.0323045.g027

and smooth. In this study, we have incorporated a comparative quantitative analysis of the conventional RRT [45] and our approach. It could be observed evidently that the proposed strategy gains the improvements in both indicators, and our smoothness evaluation is quantitative, statistically valid, and technically robust.

From those results, it can be seen clearly that our technique enables rapid obstacle detection, adaptive path planning, and trajectory re-generation in dynamic environments. This integration involving RRT scheme and visual servoing technology enhances the robustness and adaptability of robotic motion planning. With the visual approach, it plays a role as the senses of the system, continuously capturing and interpreting data from surrounding. In Table 8, several competitions between our approach and conventional method are carried out.

Threats to validity. Although the capsule-like model is effective for collision avoidance by reason of its smooth shape and simplified mathematical representation. Nevertheless, several barriers in handling irregular obstacles and highly dynamic environments could affect on the accurate manipulation and computational efficiency. To summarize the pros and cons of capsule-like model, Table 9 demonstrates the evaluation of possible solutions if our method is utilized. In fact, in the working environment of extremely irregular obstacles, the proposed approach could be implemented the switching mechanism to deploy capsule-like model for simplified objects and complex convex hulls or voxel-based prototype for

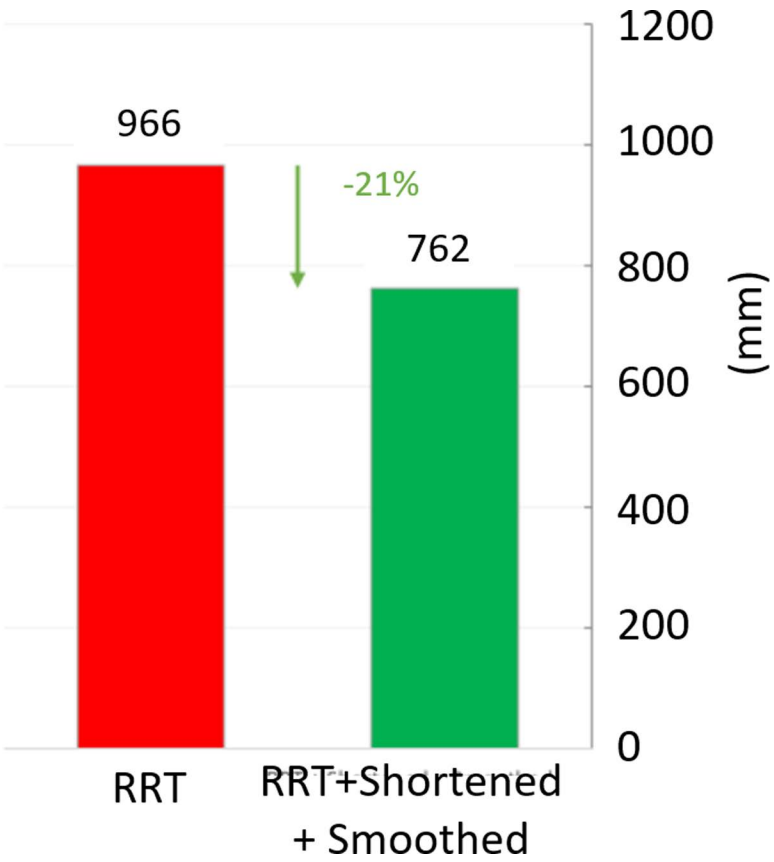

Fig 28. Comparative result of path length between our approach (green) and traditional RRT scheme (red) in case 2.

https://doi.org/10.1371/journal.pone.0323045.g028

irregular objects. Secondly, in the narrow space, capsule-based model produces a uniform safety margin which is not proper to varying constraints. An adaptive capsule of sizing design might be suitable for dense obstacles. In the highly dynamic environments, frequent re-computations of capsule-based distance could be computationally expensive. It requires to implement predictive motion model, i.e., Kalman filter or deep learning-based technique to anticipate dynamic object movements and reduce unnecessary re-computation.

In point of fact, the path shortening and smoothing technique introduce additional computational overhead, but the impact depends on the implementation strategy. Path shortening removes unnecessary nodes from the RRT-generated trajectory by connecting non-consecutive nodes directly while ensuring collision-free movement. Henceforth, path shortening slightly rises computation time but remains feasible with optimizations. In term of path smoothing, it could be performed by using the flexible interpolators such as cubic spline, to generate continuous and differentiable motion trajectories. Therefore, path smoothing is computationally lightweight in most cases and feasible with local fitting methods.

6 Conclusions

In this study, a novel motion planner by using vision-based approach for capsule-like SCORBOT in different scenarios was developed. Primarily, robotic theory including forward kinematic and inverse kinematic, camera installation and system setup were mentioned. Several image processing techniques have been integrated in order to obtain the object coordinate, object recognition and classification. Our concept is to propose the motion planning for capsule-like model, combine the shorten and smoothen profile, as well as vision-based approach. To verify the effectiveness of the proposed

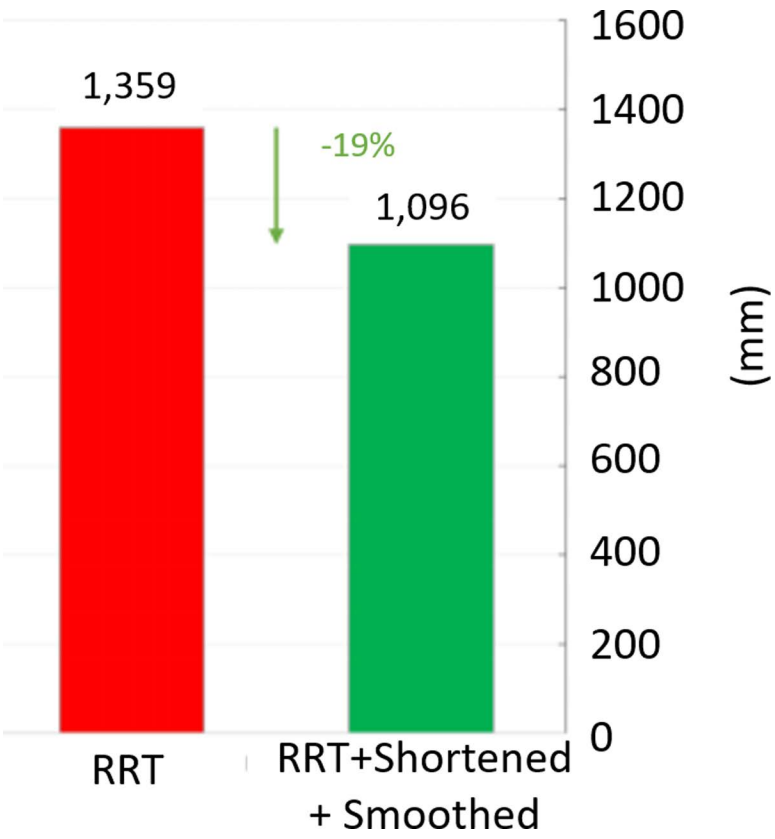

Fig 29. Comparative result of path length between our approach (green) and traditional RRT scheme (red) in case 3.

https://doi.org/10.1371/journal.pone.0323045.g029

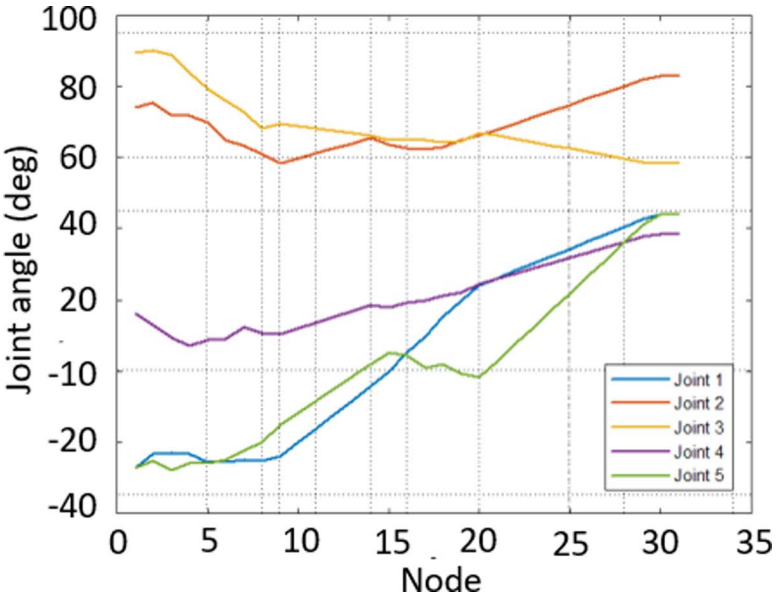

Fig 30. Experimental result of joint angle using traditional RRT scheme in case 2.

https://doi.org/10.1371/journal.pone.0323045.g030

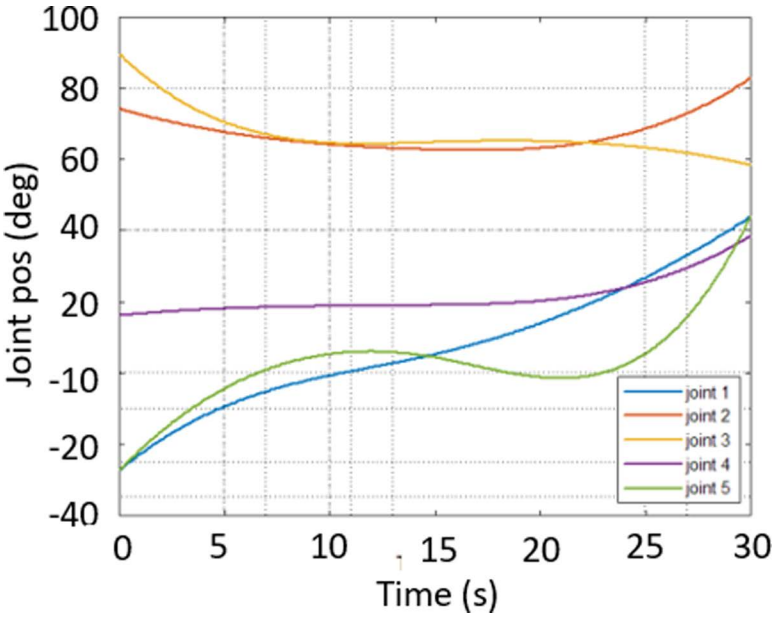

Fig 31. Experimental result of joint angle using our approach in case 2.

https://doi.org/10.1371/journal.pone.0323045.g031

Fig 32. Experimental result of the proposed system using our motion planning scheme in case 3.

https://doi.org/10.1371/journal.pone.0323045.g032

method, both simulations and experiments have been accomplished. From these results, it could be seen obviously that our approach is robust, available, and feasible for the real-world applications.

Future work is a must. Our method is available for various industrial manipulators with the intricate grippers. However, with the untrained objects for instance irregular shapes or unknown profiles, it is essential to integrate the advanced machine learning techniques such as deep reinforcement learning or Q-learning for enhanced object shape recognition in cluttered environment. Besides, the adaptive control algorithms, i.e., model predictive control or adaptive neural controller, could be implemented to dynamically tune planning parameters to deal with varying constraints. In addition, collaborative multi-robot system may be explored to extend the applicability of our approach. Also, extensive simulation results with supplementary runs per scenario as well as wider experiments are needed to statistically validate robustness.

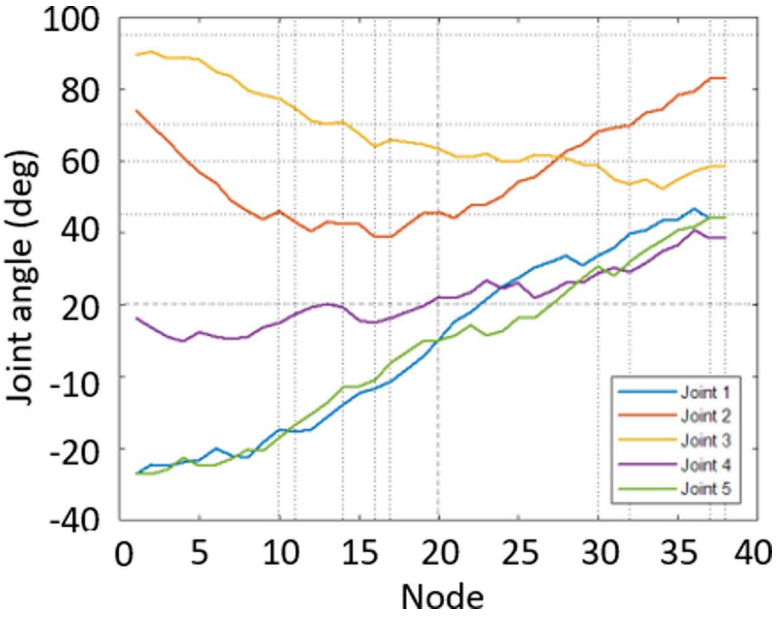

Fig 33. Experimental result of joint angle using traditional RRT scheme in case 3.

https://doi.org/10.1371/journal.pone.0323045.g033

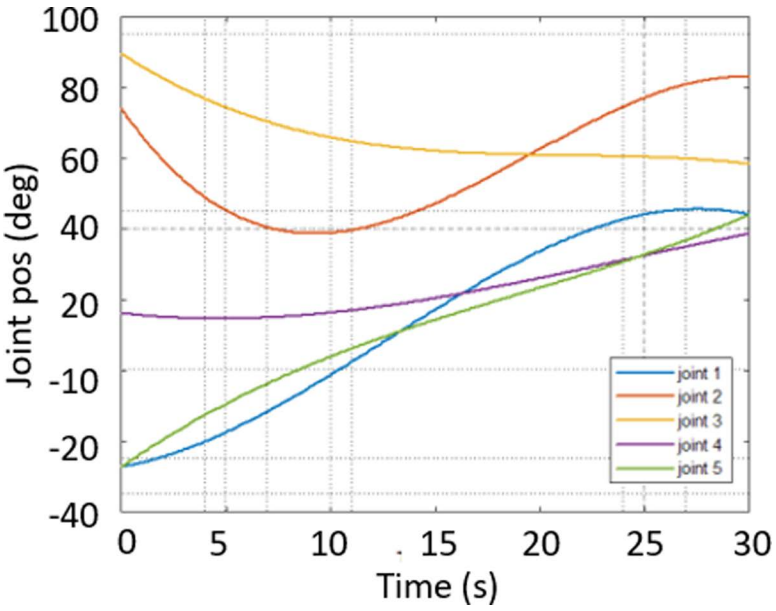

Fig 34. Experimental result of joint angle using our approach in case 3.

https://doi.org/10.1371/journal.pone.0323045.g034

Table 6. List of timing comparison in various schemes.

Item	Case 1		Case 2		Case 3	
	Time (s)	Ratio (%)	Time (s)	Ratio (%)	Time (s)	Ratio (%)
Collision valid	0,428	34,2	0,512	33,2	0,891	46,9
RRT	0,820	65,6	0,925	60,0	1,135	59,7
Shortening	0,408	32,6	0,520	33,7	0,701	36,9
Smoothing	0,022	1,8	0,097	6,3	0,065	3,4
Total	1,250	100	1,542	100	1,901	100

Collision check time includes both collisions check in RRT and shortening algorithm.

https://doi.org/10.1371/journal.pone.0323045.t006

Table 7. List of timing comparison in various schemes.

Item	Traditional RRT	Proposed approach	Improvement
Path Curvature (C)	1,24	0,65	47.6% Smoother
Jerk (J)	0,93	0,52	44.1% Less Jerk
TVJA	5,87	3,21	45.3% Reduction

https://doi.org/10.1371/journal.pone.0323045.t007

Table 8. Competitive performance between traditional method and our approach.

Traditional method	Our approach
Assumes a static environment	Continuously updates the environment model in real time
Pre-planned path with no reactivity	Dynamically replans and adjusts the path on-the-fly
Slow convergence due to random sampling	Goal bias factor & adaptive sampling speed up convergence
Sharp turns and unsmoothed paths	Spline interpolation ensures smooth trajectories
Collision handling requires full re-planning	Rewiring technique reduces computation time

https://doi.org/10.1371/journal.pone.0323045.t008

Table 9. List of the performance validation for our model and possible solutions in the complicated environments.

Scenario	Capsule-like model	Alternative method
Simple static obstacles	Yes	Not Applied
Highly irregular obstacles	No	Convex hulls, voxel grids
Narrow corridors or gaps	Partially	Adaptive capsule radius
Fast-moving obstacles	No	Motion prediction

https://doi.org/10.1371/journal.pone.0323045.t009

Supporting information

S1 File. Related data for this research.

(RAR)

Acknowledgments

We acknowledge Ho Chi Minh City University of Technology (HCMUT), VNU-HCM for supporting this study.

Author contributions

Conceptualization: Ha Quang Thinh Ngo.

Methodology: Hung Nguyen, Thanh Phuong Nguyen.

Project administration: Thanh Phuong Nguyen.

Software: Song Hung Nguyen.

Supervision: Hung Nguyen.

Validation: Song Hung Nguyen.

Writing – original draft: Ha Quang Thinh Ngo.

Writing – review & editing: Ha Quang Thinh Ngo.

References

1. Reda M, Onsy A, Haikal AY, Ghanbari A. Path planning algorithms in the autonomous driving system: A comprehensive review. Robotics and Autonomous Systems. 2024;174:104630. https://doi.org/10.1016/j.robot.2024.104630
2. Lau BPL, Ong BJY, Loh LKY, Liu R, Yuen C, Soh GS, et al. Multi-AGV's Temporal Memory-Based RRT Exploration in Unknown Environment. IEEE Robot Autom Lett. 2022;7(4):9256–63. https://doi.org/10.1109/lra.2022.3190628
3. Lindqvist B, Agha-Mohammadi AA, Nikolakopoulos G. Exploration-RRT: A multi-objective path planning and exploration framework for unknown and unstructured environments. In 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE; 2021, 3429–35.
4. Yuan C, Liu G, Zhang W, Pan X. An efficient RRT cache method in dynamic environments for path planning. Robotics and Autonomous Systems. 2020;131:103595. https://doi.org/10.1016/j.robot.2020.103595
5. Wahab MNA, Nefti-Meziani S, Atyabi A. A comparative review on mobile robot path planning: Classical or meta-heuristic methods?. Annual Reviews in Control. 2020;50:233–52. https://doi.org/10.1016/j.arcontrol.2020.10.001
6. Salzman O, Halperin D. Asymptotically Near-Optimal RRT for Fast, High-Quality Motion Planning. IEEE Trans Robot. 2016;32(3):473–83. https://doi.org/10.1109/tro.2016.2539377
7. Zohaib M, Pasha SM, Javaid N, Iqbal J. IBA: Intelligent Bug Algorithm–A novel strategy to navigate mobile robots autonomously. In Communication Technologies, Information Security and Sustainable Development: Third International Multi-topic Conference, IMTIC 2013, Jamshoro, Pakistan, December 18--20, 2013, Revised Selected Papers 3. Springer International Publishing; 2014, 291–9.
8. Wang J, Chi W, Li C, Wang C, Meng MQ-H. Neural RRT*: Learning-Based Optimal Path Planning. IEEE Trans Automat Sci Eng. 2020;17(4):1748–58. https://doi.org/10.1109/tase.2020.2976560
9. Yu J, Chen C, Arab A, Yi J, Pei X, Guo X. RDT-RRT: Real-time double-tree rapidly-exploring random tree path planning for autonomous vehicles. Expert Systems with Applications. 2024;240:122510. https://doi.org/10.1016/j.eswa.2023.122510
10. Noreen I, Khan A, Habib Z. Optimal Path Planning using RRT* based Approaches: A Survey and Future Directions. ijacsa. 2016;7(11). https://doi.org/10.14569/ijacsa.2016.071114
11. Zhang H, Wang Y, Zheng J, Yu J. Path Planning of Industrial Robot Based on Improved RRT Algorithm in Complex Environments. IEEE Access. 2018;6:53296–306. https://doi.org/10.1109/access.2018.2871222
12. Zhou C, Huang B, Fränti P. A review of motion planning algorithms for intelligent robots. J Intell Manuf. 2022;33(2):387–424. https://doi.org/10.1007/s10845-021-01867-z
13. Wahab MNA, Nefti-Meziani S, Atyabi A. A comparative review on mobile robot path planning: Classical or meta-heuristic methods?. Annual Reviews in Control. 2020;50:233–52. https://doi.org/10.1016/j.arcontrol.2020.10.001
14. Denny J, Qin D, Zhou H. A Fast and Approximate Medial Axis Sampling Technique. 2021 IEEE International Conference on Robotics and Automation (ICRA). 2021:10213–9. https://doi.org/10.1109/icra48506.2021.9562067
15. Lathrop P. Motion Planning Algorithms for Safety and Quantum Computing Efficiency (Doctoral dissertation, UC San Diego). 2023.
16. Marcucci T, Petersen M, von Wrangel D, Tedrake R. Motion planning around obstacles with convex optimization. Sci Robot. 2023;8(84):eadf7843. https://doi.org/10.1126/scirobotics.adf7843 PMID: 37967206
17. Li Y, Wei W, Gao Y, Wang D, Fan Z. PQ-RRT*: An improved path planning algorithm for mobile robots. Expert Systems with Applications. 2020;152:113425. https://doi.org/10.1016/j.eswa.2020.113425
18. Li B, Chen B. An Adaptive Rapidly-Exploring Random Tree. IEEE/CAA J Autom Sinica. 2021;9(2):283–94. https://doi.org/10.1109/jas.2021.1004252
19. Orthey A, Akbar S, Toussaint M. Multilevel motion planning: A fiber bundle formulation. The International Journal of Robotics Research. 2024;43(1):3–33. https://doi.org/10.1177/02783649231209337

20. Wong C, Mineo C, Yang E, Yan X-T, Gu D. A Novel Clustering-Based Algorithm for Solving Spatially Constrained Robotic Task Sequencing Problems. IEEE/ASME Trans Mechatron. 2020;26(5):2294–305. https://doi.org/10.1109/tmech.2020.3037158
21. Zhang X, Belfer R, Kry PG, Vouga E. C-Space tunnel discovery for puzzle path planning. ACM Trans Graph. 2020;39(4). https://doi.org/10.1145/3386569.3392468
22. Tu H, Deng Y, Li Q, Song M, Zheng X. Improved RRT global path planning algorithm based on Bridge Test. Robotics and Autonomous Systems. 2024;171:104570. https://doi.org/10.1016/j.robot.2023.104570
23. Li Y, Wei W, Gao Y, Wang D, Fan Z. PQ-RRT*: An improved path planning algorithm for mobile robots. Expert Systems with Applications. 2020;152:113425. https://doi.org/10.1016/j.eswa.2020.113425
24. Zohaib M, Pasha SM, Javaid N, Salaam A, Iqbal J. An Improved Algorithm for Collision Avoidance in Environments Having U and H Shaped Obstacles. SIC. 2014;23(1). https://doi.org/10.24846/v23i1y201410
25. Yao W, de Marina HG, Lin B, Cao M. Singularity-Free Guiding Vector Field for Robot Navigation. IEEE Trans Robot. 2021;37(4):1206–21. https://doi.org/10.1109/tro.2020.3043690
26. Baron N, Philippides A, Rojas N. A robust geometric method of singularity avoidance for kinematically redundant planar parallel robot manipulators. Mechanism and Machine Theory. 2020;151:103863. https://doi.org/10.1016/j.mechmachtheory.2020.103863
27. Rebouças Filho PP, da Silva SPP, Praxedes VN, Hemanth J, de Albuquerque VHC. Control of singularity trajectory tracking for robotic manipulator by genetic algorithms. Journal of Computational Science. 2019;30:55–64. https://doi.org/10.1016/j.jocs.2018.11.006
28. Karaman S, Walter MR, Perez A, Frazzoli E, Teller S. Anytime Motion Planning using the RRT*. 2011 IEEE International Conference on Robotics and Automation. 2011:1478–83. https://doi.org/10.1109/icra.2011.5980479
29. Cheng X, Zhou J, Zhou Z, Zhao X, Gao J, Qiao T. An improved RRT-Connect path planning algorithm of robotic arm for automatic sampling of exhaust emission detection in Industry 4.0. Journal of Industrial Information Integration. 2023;33:100436. https://doi.org/10.1016/j.jii.2023.100436
30. Wang J, Li B, Meng MQ-H. Kinematic Constrained Bi-directional RRT with Efficient Branch Pruning for robot path planning. Expert Systems with Applications. 2021;170:114541. https://doi.org/10.1016/j.eswa.2020.114541
31. Wang J, Chi W, Li C, Wang C, Meng MQ-H. Neural RRT*: Learning-Based Optimal Path Planning. IEEE Trans Automat Sci Eng. 2020;17(4):1748–58. https://doi.org/10.1109/tase.2020.2976560
32. Elfizar E. Implementation of Enhanced Axis Aligned Bounding Box for Object Collision Detection in Distributed Virtual Environment. J Appl Data Sci. 2024;5(3):1286–98. https://doi.org/10.47738/jads.v5i3.226
33. Wei X, Liu M, Ling Z, Su H. Approximate convex decomposition for 3D meshes with collision-aware concavity and tree search. ACM Trans Graph. 2022;41(4):1–18. https://doi.org/10.1145/3528223.3530103
34. Shah H, Ghadai S, Gamdha D, Schuster A, Thomas I, Greiner N, et al. GPU-Accelerated Collision Analysis of Vehicles in a Point Cloud Environment. IEEE Comput Graph Appl. 2022;42(5):37–50. https://doi.org/10.1109/MCG.2022.3177890 PMID: 35613062
35. Li C, Zhang X, Gao H, Wang R, Fang Y. Bridging the Gap Between Visual Servoing and Visual SLAM: A Novel Integrated Interactive Framework. IEEE Trans Automat Sci Eng. 2021;19(3):2245–55. https://doi.org/10.1109/tase.2021.3067792
36. Hu K, Chen Z, Kang H, Tang Y. 3D vision technologies for a self-developed structural external crack damage recognition robot. Automation in Construction. 2024;159:105262. https://doi.org/10.1016/j.autcon.2023.105262
37. Tang Y, Qi S, Zhu L, Zhuo X, Zhang Y, Meng F. Obstacle avoidance motion in mobile robotics. Journal of System Simulation. 2024;36(1):1–26.
38. Ye L, Wu F, Zou X, Li J. Path planning for mobile robots in unstructured orchard environments: An improved kinematically constrained bi-directional RRT approach. Computers and Electronics in Agriculture. 2023;215:108453. https://doi.org/10.1016/j.compag.2023.108453
39. Ngo HQT. Design of automated system for online inspection using the convolutional neural network (CNN) technique in the image processing approach. Results in Engineering. 2023;19:101346. https://doi.org/10.1016/j.rineng.2023.101346
40. Ullah MI, Ajwad SA, Irfan M, Iqbal J. Non-linear Control Law for Articulated Serial Manipulators: Simulation Augmented with Hardware Implementation. EIAEE. 2016;22(1). https://doi.org/10.5755/j01.eee.22.1.14094
41. Nguyen TP, Nguyen H, Ngo HQT. Visual application of navigation framework in cyber-physical system for mobile robot to prevent disease. International Journal of Advanced Robotic Systems. 2023;20(2):172988062311622. https://doi.org/10.1177/17298806231162202
42. Ruan S, Poblete KL, Wu H, Ma Q, Chirikjian GS. Efficient Path Planning in Narrow Passages for Robots With Ellipsoidal Components. IEEE Trans Robot. 2022;39(1):110–27. https://doi.org/10.1109/tro.2022.3187818
43. Baxter J, Yousefi MR, Sugaya S, Morales M, Tapia L. Deep prediction of swept volume geometries: Robots and resolutions. In 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE; 2020, 6665–72.
44. XUE Z, LIU J, WU C, TONG Y. Review of in-space assembly technologies. Chinese Journal of Aeronautics. 2021;34(11):21–47. https://doi.org/10.1016/j.cja.2020.09.043
45. Karaman S, Walter MR, Perez A, Frazzoli E, Teller S. Anytime Motion Planning using the RRT*. 2011 IEEE International Conference on Robotics and Automation. 2011:1478–83. https://doi.org/10.1109/icra.2011.5980479