Academic Journal of Computing & Information Science
ISSN 2616-5775 Vol. 6, Issue 11: 159-168, DOI: 10.25236/AJCIS.2023.061120

Modeling and Simulation Verification of Pipeline Cooperative Robots

Jianmin Chen[1], Wusheng Ji[1,*]

[1]School of Electronic Engineering, Tianjin University of Technology and Education, Tianjin, 300222, China
[*]Corresponding author: jiwusheng@tute.edu.cn

Abstract: *With the robot modeling and simulation software CoppeliaSim as the simulation environment, YouBot and the six-degree of freedom robot arm as the main body of the collaborative assembly line. Through analyzing the structure of the McNamum wheel of the moving chassis of YouBot robot, the relationship between the speed of each wheel and the attitude of the chassis is deduced, and its kinematics model is established. And the motion simulation is carried out. MDH method was adopted to carry out kinematics modeling of the manipulator arm of the cooperative robot, MATLAB and Python were used to implement motion trajectory planning, and remote API synchronous mode was used to control the cooperative robot in CoppeliaSim, so as to realize the simulation of the sorting, delivery and material transportation process of the assembly line cooperative robot. The simulation results show that this method can smoothly complete the sorting, delivery and transportation of materials.*

Keywords: *CoppeliaSim; Collaborative robot; Kinematics simulation; Trajectory planning; Material*

1. Introduction

With the deepening of the research on robot related fields, the requirement of robot simulation software is getting higher and higher. Compared with the physical platform verification of robots, simulation software, with its advantages of high efficiency, low cost and safety, has become a key link in the design and manufacturing process of robots. It can verify the effectiveness of robots on the principle level, so as to solve the problems that may occur in the actual operation of robots in advance and avoid the security risks [1]. Selecting suitable simulation tools to apply to different simulation objects can greatly facilitate robot research. Matlab and Python are widely used in the field of robot simulation due to their excellent performance in robot kinematics calculation, verification and trajectory planning, etc. However, they are separated from the actual robot model in the process of motion control and cannot accurately display the robot model and its pose changes in the simulation process. As a result, it is impossible to accurately judge whether there is collision or interference in the robot movement [2]. Most of the current co-simulation systems rely on Matlab and Python and other visual simulation software ADAMS, CATIA, Matlab Simscape, etc. [3] When the simulation object is clear, the motion simulation model of the control robot in ADAMS and CATIA will be inconsistent with the actual model, and it will be difficult to judge the interference of other devices on the cooperative robot during the simulation motion process, and the simulation response effect is not good [4]. Simscape's interactive simulation results are not satisfactory, with limited use scenarios [5].

CoppeliaSim is an open source cross-platform robot modeling and simulation software for integrated development environment. Coppeliasim has the advantages of rich internal and external libraries, high-quality simulation engines, multiple application program interfaces, support for various control methods and multiple programming languages, and has excellent operation results in calculation routines, processing and detection. It can be the main choice of robot simulation. However, the simulation process of this software relies too much on scripts and has poor operability [6].

In this paper, two cooperative robots are taken as research objects, and a visual co-simulation experiment combining CoppeliaSim with Matlab and Python assembly language is realized by using LuaSocket synchronous communication mode. Three-dimensional modeling, kinematic analysis, dynamic model establishment and simulation experiment design are carried out for the cooperative robot. Intuitively display and obtain the motion posture and trajectory of the robot during motion. The co-simulation results show that the CoppeliaSim client responds to the external control program accurately without delay and has good controllability.

ISSN 2616-5775 Vol. 6, Issue 11: 159-168, DOI: 10.25236/AJCIS.2023.061120

2. Collaborative robot modeling

This paper selects the youBot robot produced by KUKA Company as the research object. The whole YouBot robot includes chassis, pallet and robot arm. The mobile chassis of the robot adopts the McNamum wheel structure, which enables it to achieve omnidirectional movement and rotation capabilities, making the robot more flexible and reliable in small Spaces [7]. In order to realize the accurate control of youBot chassis, it is necessary to build the kinematics model of the trolley chassis by modeling the motion joints of the chassis. By analyzing the inverse kinematics of youBot chassis, the relationship between the speed of each wheel and the attitude of chassis is deduced, and the motion simulation is carried out. The robot arm of YouBot is a six-degree-of-freedom robot arm composed of five rotating joints (A1-A5) and a corresponding connecting rod mechanism expected by the end effector in series. It is composed of a base, shoulder, elbow, wrist and Java. The joint diagram is shown in Figure 1.

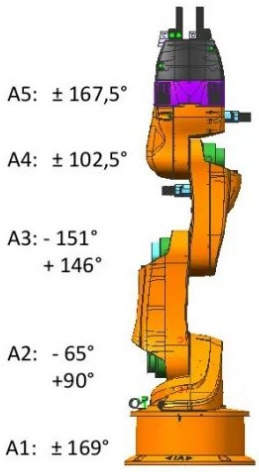

Figure 1: Schematic diagram of the arm joint of a collaborative robot

The chassis joint of the YouBot robot is composed of four Mecanum wheels (referred to as wheat wheels), which can also provide horizontal movement, unlike ordinary fixed wheels that can only move in the front and back direction. The wheat wheel is divided into A and B wheels, and a reasonable combination of wheat wheels can omit the steering mechanism of the chassis, and even can rotate in place, so this chassis with 4 McNamms is very common in mobile robots.

2.1. Establishment of cooperative robot chassis kinematics model

The four wheels of the mobile chassis of the cooperative robot adopt the structure of McNamum wheel, which is a very special wheel design. Compared with ordinary fixed wheels, McNamum wheel can provide both front and back direction movement and left and right direction tangent movement, so that the robot can realize omnidirectional movement [8].

The right front of the robot is the positive direction of the x axis, the left side is the direction of the y axis, the right side is the direction of the z axis, and the right hand rule determines the rotation direction. The advance mode of the omni chassis of the McNamum wheel refers to the motion mode of the chassis moving along the horizontal x axis. In this mode, the McNamum wheels on the chassis achieve movement by adjusting the speed and direction of the individual wheels.

In the side view, the main wheel is parallel to the chassis, and it moves at the same speed as the forward direction of the chassis. In the top view, the small hub is in contact with the ground, and its speed is related to the speed of the main wheel. Let's explain the speed relationship between the small hub and the main wheel. The small hub is connected to the main wheel by a gear mechanism. When the main wheel rolls forward, the small hub will roll outward, and the part in contact with the ground will produce a backward speed component. This backward velocity component creates an opposing friction force that allows the chassis to move forward. The advance mode of the McNamum wheel omnidirectional chassis actually adjusts the speed and direction of the individual wheels so that the chassis can advance along the horizontal X-axis. In this mode, the speed of the main wheel determines the overall direction and speed of the chassis, and the speed relationship between the small wheel hub and the main wheel ensures the stability and controllability of the chassis. Figure 2 shows the motion analysis of a single wheel on

ISSN 2616-5775 Vol. 6, Issue 11: 159-168, DOI: 10.25236/AJCIS.2023.061120

the omni chassis of the McNamum wheel.

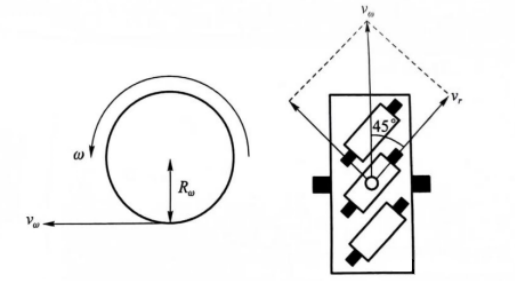

Figure 2: Motion analysis of a single wheel on a McNamum wheel omnidirectional chassis

The variable definition in the figure above is described:

The angular speed of rotation of a single wheel is the speed at which the wheel rotates around its axis and the direction of the arrow is the direction of rotation; The radius of the main wheel is a fixed value that determines the size of the main wheel. The linear speed of the edge of the main wheel, the linear speed of the point on the edge of the main wheel, which is related to the rotation speed and direction of the main wheel. The direction of the arrow indicates the direction of the velocity formed; Refers to the speed generated by the point of contact between the wheel and the ground, which is parallel to the axis direction of the small wheel hub. The direction of the arrow indicates the direction of the velocity generated by the wheel.

The relationship can be obtained from Figure 2:

$$\begin{cases} v_\omega = \omega \times R_\omega \\ v_r = v_\omega \times \cos 45^\circ \end{cases} \tag{1}$$

The motion model of the omni chassis of the McNamum wheel as it moves forward along the horizontal X-axis is shown below.

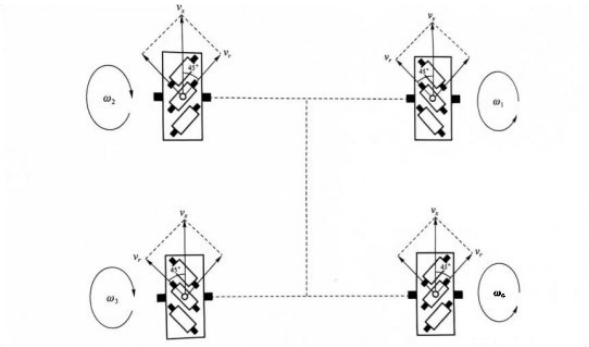

Figure 3: The motion model of the omnidirectional chassis moving along the horizontal X-axis

The variables in the figure 3 above describe the final speed of the entire chassis combined on a single wheel, with the arrows pointing in the direction of the formed speed. When the arrow faces forward, it means that the speed generated here makes the car move forward; When the arrow grass stage is behind, it means that the speed generated here makes the car body move backward; The speed generated by the wheel at the ground contact point is parallel to the direction of the wheel shaft of the small wheel hub, and the arrow direction is the speed direction generated by the wheel; The speed of the wheel as seen when facing the wheel from the outside of the rotation axis of the wheel, the subscript number indicates the number of the wheel. The direction of the arrow is the direction of the wheel rotation seen in this view.

The relationship can be obtained from the figure above:

$$V_r = V_x \times \cos 45^\circ = \frac{\sqrt{2}}{2}(\omega \times R_\omega) \tag{2}$$

The relationship between formula (1) and formula (2) can be derived from:

$$V_\omega = V_x = \omega \times \cos 45^\circ \rightarrow \omega = V_X / R_\omega \tag{3}$$

ISSN 2616-5775 Vol. 6, Issue 11: 159-168, DOI: 10.25236/AJCIS.2023.061120

According to the speed direction in the forward motion model analysis diagram, the speed relationship of the four wheels can be obtained:

$$\begin{cases} \omega_1 = V_x/R_\omega \\ \omega_2 = -V_x/R_\omega \\ \omega_3 = -V_x/R_\omega \\ \omega_4 = V_x/R_w \end{cases} \tag{4}$$

The movement mode of McNamum wheel omnidirectional chassis in the horizontal Y-axis direction is divided into left translation and right translation, the two modes are only different in speed direction, the essence is the same. The left translation mode is analyzed as an example. Similarly, the speed relationship of the four wheels when moving along the horizontal Y-axis can be obtained:

$$\begin{cases} \omega_1 = -V_y/R_\omega \\ \omega_2 = -V_y/R_\omega \\ \omega_3 = V_y/R_\omega \\ \omega_4 = V_y/R_\omega \end{cases} \tag{5}$$

The rotation mode of McNamum wheel omnidirectional chassis is divided into clockwise and counterclockwise. The counterclockwise rotation of robot is analyzed here. When the robot is rotated counterclockwise, the direction of the velocity vector of the McNamm wheel is shown in Figure 4. The velocity vector direction of the front right wheel and the rear left wheel is at an Angle of 45 degrees with the positive direction of the robot, and the velocity vector direction of the front left wheel and the rear right wheel is at an Angle of 135 degrees with the positive direction of the robot. This motion model allows the robot to remain in a fixed position while rotating and be able to move in any direction, as shown in Figure 4. The linear speed generated by the four wheels at the ground contact point, the arrow direction is the speed direction generated by the wheels, parallel to the axis of the small wheel hub; Is the actual linear speed of the rotation of the car body, which is synthesized by the linear speed v of the small wheel hub and the speed component of the free rolling direction of the small wheel hub. The speed direction and the contact point of the main wheel are perpendicular to the line "c" of the center of the car body; Is the wheel speed seen when facing the wheel from the outside of the wheel rotation axis, and the subscript number indicates the number of the wheel. The direction of the arrow is the rotation direction of the wheel seen in this Angle of view; a The transverse distance between the contact point of the McNamham wheel and the center of the car body is a fixed value; b is the longitudinal distance from the contact point of the McNamu wheel to the center of the car body, and is a fixed value; c is the connection between the contact point of the McNamu wheel and the center of the car body, and is also the angular velocity radius of the rotation of the car body; α is the Angle between the contact point of the McNamu wheel and the center of the car body and the horizontal direction, and is also the Angle between the linear speed of the car body rotation and the vertical direction, and the two angles form a supplementary Angle with the same Angle, so they are equal; θ is the Angle between the actual linear speed of the car body rotation and the linear speed generated by the small wheel hub at the contact point; Is the angular velocity of the robot's own rotation.

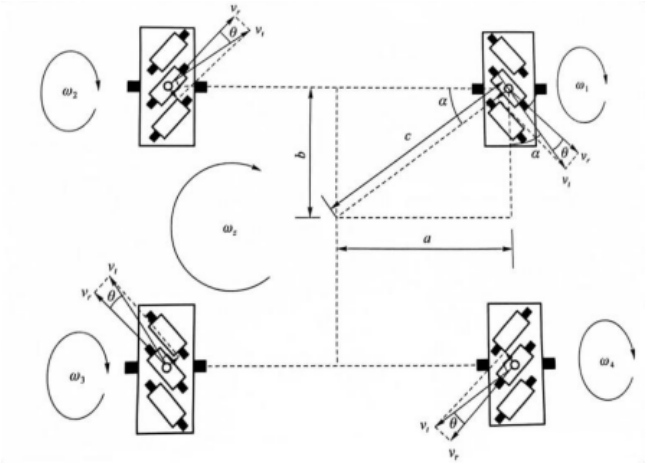

Figure 4: Motion model of omnidirectional chassis rotation

The following relationship can be obtained from Figure 4:

ISSN 2616-5775 Vol. 6, Issue 11: 159-168, DOI: 10.25236/AJCIS.2023.061120

$$\begin{cases} V_t = \omega_z \times v \\ V_t = V_t \times \cos\theta \\ \theta + a = 45° \end{cases} \rightarrow \omega_z \times c = \frac{V_r}{\cos(45° - \alpha)} = \frac{\sqrt{2}V_r}{\cos\alpha + \sin\alpha} \rightarrow \omega_z = \frac{\sqrt{2}V_r c}{a+b} \tag{6}$$

By substituting formula 2 into formula 6:

$$\omega_z = \frac{\omega \times R_\omega}{a+b} \tag{7}$$

According to the rotation direction of the four wheels during rotating movement, the rotational speed relationship of each wheel can be derived:

$$\omega_1 = \omega_2 = \omega_3 = \omega_4 = -\omega_z \times (a+b) / R_\omega \tag{8}$$

All motion states of the omni chassis of the McNamum wheel can be regarded as the compound state of the above three motion modes, so the relationship between the speed of the last four wheels is the sum of the speed of the above motion modes.

$$\begin{cases} \omega_1 = V_x/R_\omega - V_y/R_\omega - \omega_z \times (a+b)/R_\omega \\ \omega_2 = -V_x/R_\omega - V_y/R_\omega - \omega_z \times (a+b)/R_\omega \\ \omega_3 = -V_x/R_\omega + V_y/R_\omega - \omega_z \times (a+b)/R_\omega \\ \omega_4 = V_x/R_\omega + V_y/R_\omega - \omega_z \times (a+b)/R_\omega \end{cases} \tag{9}$$

Variable description in the above formula: the wheel speed seen from the outside of the wheel rotation axis to the wheel, the subscript number represents the number of the wheel; The speed of the car body moving along the horizontal axis is positive when the car body is moving forward, and negative when the car body is moving backward. The speed at which the car body moves along the horizontal y axis. This value is positive when the car body moves to the left, and negative when the car body moves to the right. The radius of the outer circle of the McNamm wheel is a fixed value; The transverse distance between the contact point of the McNam wheel and the center of the car body is a fixed value; b The longitudinal distance between the contact point of the McNamum wheel and the center of the car body is a fixed value.

2.2. Establishment of kinematics model of robot arm of cooperative robot

The robotic arm of a collaborative robot can be viewed as consisting of six connecting rods connected in series by rotating joints, each of which is driven by an independent motor. This kind of robot arm actually adopts a semi-closed loop control structure, that is, the system can only precisely control the position of the joint servo motor, but can not directly control the position and pose of the robot end actuator. The relationship between the joint position and the pose of the robot end effector is determined by kinematic modeling.

In order to carry out kinematic modeling, MDH method is adopted, and each link is numbered from the fixed base of the robot. The first movable link is link 1, which is sorted successively. The modeling steps are as follows

(1) Determine the Z-axis of each joint: According to Figure 5, one end of each connecting rod i-1 has a joint, and the joint axis i-1 is located at the end of connecting rod i-1 near the base. Similarly, joint axis i is located at the end of link i near the base. Select to set the axes along the axis of joint I-1 and set the axis along joint i to the z axis. When determining the positive direction of the Z-axis, the Z-axes of parallel joints are usually chosen to be in the same positive direction.

(2) Determine the x axis of each joint through the z axis: there is a common vertical line between each pair of adjacent joint axes, set the coordinate axis along this common vertical line, and point to the direction of the axis. The intersection of the common vertical with the z axis is defined as the origin.

(3) Determine the Y-axis direction: After joint i-1 determines the axis, axis and origin, the right-handed rule can be used to determine the direction of the Y-axis.

ISSN 2616-5775 Vol. 6, Issue 11: 159-168, DOI: 10.25236/AJCIS.2023.061120

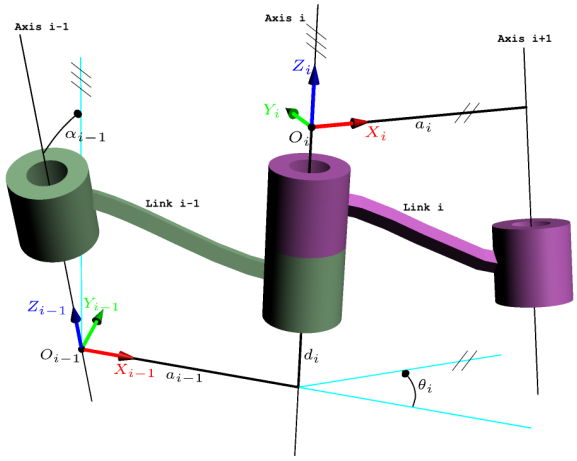

Figure 5: Parameter diagram of rotary joint connecting rod

Similarly, it can be determined that the coordinate system of connecting rod i is located on the axis of the joint, and the origin of the coordinate system and the directions of each axis are shown in Figure 5. In order to describe the pose relationship between two adjacent coordinate systems, the D-H parameters of each link are defined as follows:

Connecting rod twist Angle α_{i-1}: the Angle between joint axis Z_{i-1} and joint axis Z_i.

Connecting rod length a_{i-1}: the length of the common vertical line between the joint axis Z_{i-1} and the joint axis Z_i.

Connecting rod Angle θ_i: the Angle between two common vertical lines a_{i-1} and a_i.

Link distance: The distance between two common vertical lines a_{i-1} and a_i.

By determining the D-H parameters of each link, the pose relationship between the links is established. These parameters are very important for the kinematics model and attitude calculation of the robot, and play a key role in motion planning and control.

KUKA youBot robot arm is a six-joint series robot. In Table 1, MDH modeling method is adopted to establish coordinate system for each connecting rod.

Table 1: Linkage parameters of KUKAyouBot machine

j	Connecting rod Angle θ_i/(°)	Connecting rod distance d_i/m	Connecting rod length a_i/m	Connecting rod twist Angle α_{i-1}/(°)
1	$-169 \leq \theta_i \leq 169$	0.147	0.033	90
2	$-65 \leq \theta_i \leq 90$	0	0.155	0
3	$-151 \leq \theta_i \leq 146$	0	0.135	0
4	$-102 \leq \theta_i \leq 102$	0	0	-90
5	$-167 \leq \theta_i \leq 167$	0.113	0	0

After determining the coordinate system and 4 parameters of each link, the position transformation relationship between link i-1 and link can be determined by establishing the transformation relationship between coordinate system $x_{i-1}y_{i-1}z_{i-1}$ and coordinate system $x_iy_iz_i$. The homogeneous transformation matrix obtained after transformation is:

ISSN 2616-5775 Vol. 6, Issue 11: 159-168, DOI: 10.25236/AJCIS.2023.061120

$$
\begin{aligned}
{}^{i-1}_{i}T &= Rot(x,a_{i-1})T\,rans(\alpha_{i-1},0,0)Rot(z,\theta_{\mathrm{i}})T\,rans(0,0,\mathrm{d}_{\mathrm{i}}) \\
&= \begin{bmatrix} \cos\theta_{\mathrm{i}} & -\sin\theta_{\mathrm{i}} & 0 & a_{i-1} \\ \sin\theta_{\mathrm{i}}\cos a_{i-1} & \cos\theta_{\mathrm{i}}\cos a_{i-1} & -\sin a_{i-1} & -\mathrm{d}_{\mathrm{i}}\sin a_{i-1} \\ \sin\theta_{\mathrm{i}}\sin a_{i-1} & \cos\theta_{\mathrm{i}}\sin a_{i-1} & \cos a_{i-1} & \mathrm{d}_{\mathrm{i}}c9osa_{i-1} \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned} \tag{10}
$$

Therefore, the pose transformation matrix of link 5 relative to the robot base coordinate system is: ${}^{0}_{5}T={}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T\,{}^{4}_{5}T$

3. Co-simulation of CoppeliaSim and Matlab/Python

3.1. Communication simulation principle

CoppeliaSim is a highly customizable simulation software that itself can be customized and customized to perform correctly as needed. V-REP uses Lua language for simulation, but its control of robot simulation is not flexible enough [9]. But CoppeliaSim provides a set of remote interfaces for other languages, making it possible to use shared memory or network communication (LuaSocket) on Windows. To complete the control of external applications (C /C + +, Java, Matlab, Python, etc.) or remote hardware (such as real robots, remote computers, etc.) for simulation [10]. Therefore, it is possible to use Matlab and Python with Lua to write external control programs, and through a series of RemoteAPI interfaces provided by CoppeliaSim, mix a variety of ways for users to remotely operate and control the robot model in CoppeliaSim.

LuaSocket is a module library for multi-network protocol access, which supports four communication modes: blocking function call, non-blocking function call, data flow and synchronous operation. In order to minimize the network delay and load during co-simulation, the communication interface and control program are written in CoppeliaSim script, and the interaction with Matlab is completed through remote API functions in LuaSocket communication.

3.2. Co-simulation of communication flow

The co-simulation adopts the client-server communication mode, with CoppeliaSim as the client and Matlab/Python as the server. The client receives the instructions issued by the server control program continuously through LuaSocket communication to achieve accurate robot motion control.In order to improve the simulation speed, CoppeliaSim and Matlab/Python use synchronous mode for communication.The communication and simulation process is:

1) When the above work is completed, the simulation starts. The client obtains the control instruction of the server and sends it to the execution handle. After the calculation of the control program, the server sends the control instruction to the client through LuaSocket and controls the robot to move according to the instruction after parsing.

2) In the simulation process, it is necessary to start the simulation of CoppeliaSim first, and then run the control code on the client of Python and Matlab respectively to complete the control of the cooperative robot. The writing method and combination of the control code can be changed.

3) At the end of the simulation, the server call statement interrupts the communication between the server and the client.

4. Design of pipeline collaborative robot simulation experiment

In the simulation experiment of the assembly line cooperative robot, the cooperative robot JAKA Zu-12 is designed to recognize the orange and purple materials, and place the purple materials at the target position on the ground, the orange materials at the platform of the material trolley, and the cooperative robot YouBot is used to put the blue and orange materials into the target position of the material pool, as shown in Figure 6.

ISSN 2616-5775 Vol. 6, Issue 11: 159-168, DOI: 10.25236/AJCIS.2023.061120

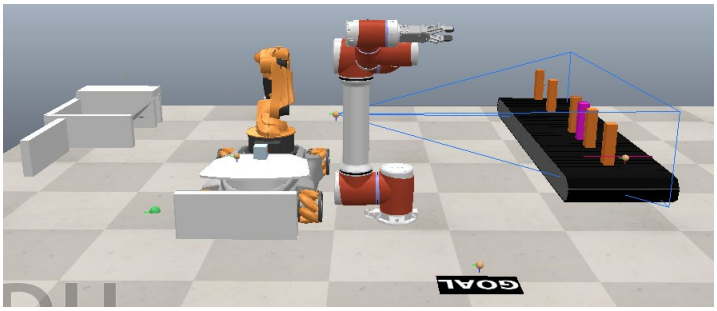

Figure 6: Pipeline simulation scene

4.1. Visual processing

Vision sensor is used in the camera for vision processing. The purpose of using vision sensor is to measure and process image inspection. The vision sensor in VREP can generate two kinds of data streams in the simulation process: color image and depth map. The resulting images can be linked to the Floating view for real-time observation. The data can be obtained through API functions, and then each pixel of the image can be processed, or the filter provided by VREP can be used to process the image. In the case of API function, its flexibility is very good, and the image data to be processed in this experiment is relatively simple, so the visual processing part of this experiment adopts simVision library function, which is an API function for performing simple image processing and processing special types of visual algorithms, and is sufficient to meet the image processing of this project, as shown in Figure 7.

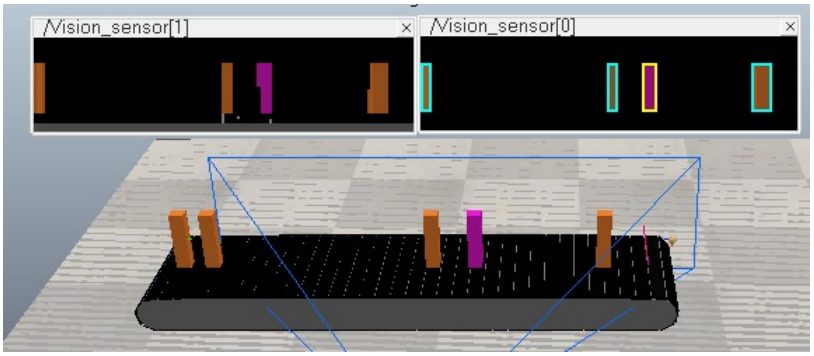

Figure 7: Obtaining image processing results

4.2. Robot trajectory planning and motion

In CoppeliaSim, Path is an "entity" of Scence. Like model, Path can be used to define various motion paths. On Path, the motion process of the robot at every moment in the entire path can be defined given the position of the robot moving to a certain point and its posture at a certain moment. The default Path in CoppeliaSim is to obtain a smooth path by giving some key control points and interpolating Bessel curves between these control points. Dummy is often used to combine with Path. Dummy can move forward along the Path, obtain the position and posture of the current moment on the Path in real time, and act as a "pilot" on the path. Therefore, the trajectory planning and movement of the collaborative robot in this project can be composed of the following five parts:

Obtain the handle of Dummy and Path → read the material information (obtain the material information stored on the vision sensor DataBlock; if there is no material on the conveyor belt, the sim. followPath() function will be called to make the Dummy move along the Path ata certain speed) → the cooperative robot obtains the Dummy handle. The pose information of the Dummy is obtained → The robot follows the pose of the Dummy according to the PD algorithm to follow the trajectory.

ISSN 2616-5775 Vol. 6, Issue 11: 159-168, DOI: 10.25236/AJCIS.2023.061120

4.3. Experimental results and analysis

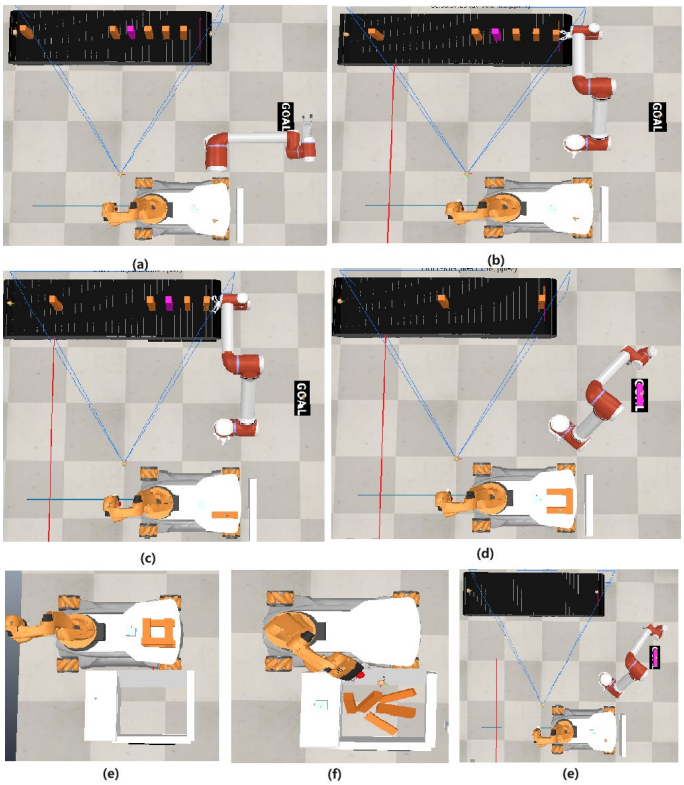

Figure 8: Pipeline simulation process display

The pipeline simulation process is shown in Figure 8: (a) The initial pose of the two robots, and the conveyor belt starts to run at a uniform speed; (b) is the state where JAKA Zu-12 robot grabs the first material, and YouBot robot is in the waiting state; (c) Transfer the orange material to the YouBot robot platform for the JAKA Zu-12 robot and grab other materials on the conveyor belt; (d) The JAKA Zu-12 robot places the purple material at the target position on the ground after it reaches the end of the conveyor belt; (e) The JAKA Zu-12 robot has completed the identification and grasp of all the materials on the conveyor belt, and the YouBot robot has started to transport the materials on its platform to the material pool; (f) For YouBot, place the blue material on its platform on the right side of the material pool platform, and the orange material in the material pool; (g) In order to complete the entire pipeline simulation test, the JAKA Zu-12 robot and the YouBot robot return to the initial position.

There are eight materials in the overall experiment (one purple material, six orange materials, and one blue material). After repeated adjustment of experimental results and procedures, the grasping results of materials are listed in Table 2.

Table 2: Process analysis and results of pipeline collaborative robot simulation experiment

Test number	Overall experiment completion time(S)	Material identification rate(%)	Material grab success rate(%)	Material handling success rate(%)
1-10	85	60%	70%	80%
11-30	76	76%	75%	85%
30-60	69	82%	82%	95%
60-100	64	90%	89%	100%
100-110	61	99%	99%	10%

It can be seen from Table 2 that the simulation experiment of the assembly line cooperative robot can complete the task relatively completely and quickly after several experimental results and program adjustments. Therefore, it can be concluded that during the whole process of the assembly line, the two robots work closely together and use interpolation method to plan the trajectory of the robots in the assembly line system. The motion process of the robot is smooth and reliable, and it is verified that the interpolation method used in this system has a good effect on trajectory planning, and the trajectory is the same as the original trajectory. The overall experiment demonstrates the ability of the two robots to work together.

ISSN 2616-5775 Vol. 6, Issue 11: 159-168, DOI: 10.25236/AJCIS.2023.061120

5. Conclusion

In this paper, CoppeliaSim and Matlab/Python motion control co-simulation of pipeline collaborative robots are used. Through the simulation experiment platform and simulated working environment built, it is verified that the robot has good dynamic response speed and accurate trajectory motion ability under LuaSocket communication. The dynamic motion process of the cooperative robot under the control model is visually demonstrated, which can provide theoretical reference and experimental basis for the development of physical prototype in practical work. The method in this paper is applicable not only to pipeline cooperative robots, but also to robots with other configurations. By writing the internal script of CoppeliaSim and the control command program of Matlab/Python, the control simulation of robots for various purposes can be realized, which has a reference significance for the field of robot simulation.

References

[1] Yu Zhenzhong, CAI Kaiti. Three-dimensional Simulation System of Six-axis Industrial Robot [J]. Computer Engineering and Design,2017,38(9):2489-2493. (in Chinese)

[2] Chen Rongchuan. Virtual Experiment Design and reinforcement learning of automatic loading and unloading platform of robot assembly line [D]. Donghua University,2020.

[3] Zhu Yun, Ling Zhigang, Zhang Yuqiang. Research progress and prospect of machine vision technology [J]. Journal of Graphics, 2019,41(6):871-890. (in Chinese)

[4] Wang Jian, Zhang Gong, He Wenjie et al. Simulation and Experimental research on Cooperative Motion of Two Robots based on V-REP [J]. Modern Manufacturing Engineering,2021(12):21-27.

[5] Zhang Shichao, Tao Cunbing, Huang Wei. Robot Simulation application based on V-REP and MATLAB [J]. Ship Electronic Countermeasures, 2019,43(03):111-114. (in Chinese)

[6] Chen Haodong, Shi Jinfei, GAO Haitao et al. Co-simulation of robot motion Control based on CoppeliaSim and Matlab [J]. Journal of Nanjing Institute of Technology (Natural Science Edition), 2023, 21(1):17-21.

[7] Ding Minghua, Li Yunwang, Wang Yong. Simulation of Obstacle avoidance Algorithm for McNamum Wheel Mobile Platform based on Unity3d [J]. Chinese Science and Technology Papers, 2016, 11(10): 1191-1195. (in Chinese)

[8] Goldman Sachs. Research on Trajectory Planning of Robot Arm based on youbot [D]. Guangdong University of Technology,2018.

[9] Ren Jianxin, Huang Min, Liu Xiangquan et al. Joint simulation of cargo picking robot based on Visual Studio and V-REP [J]. Journal of Chongqing University of Technology (Natural Science), 2019, 34(8): 87-94.

[10] Le Bin, Zeng Xingbin. Research on Remote Control Method of V-REP Robot Simulation [J]. Industrial Control Computer,2018,31(9):41-43. (in Chinese).