

Non-contact Storage Classification Based on Mechanical Arm in the Context of Prevention and Treatment of COVID-19

Naixiang Gao *

The Grainger College of Engineering, University of Illinois Urbana-Champaign, Champaign, America

* Corresponding author: ngao4@illinois.edu

Abstract. COVID-19 has become a global infectious disease, which has a high transmission rate, a wide transmission range, and a variety of transmission modes. Therefore, to reduce the transmission efficiency of COVID-19, human beings try to reduce the manpower ratio to create a non-contact environment in the industry. Therefore, a new model is proposed in this paper. This new model aims to be applied to storage classification in the context of the prevention and treatment of COVID-19 by using the mechanical robot arm. This model will use the UR3 robot arm, Logitech C300 web camera, and OpenCV public repository to implement the computer vision algorithm, and forward and inverse kinematics analysis. This model can successfully implement non-contact storage classification. Through the real experiment, this paper has a good application prospect. It gives a new solution for future research and experiment. In addition, in the summary and outlook part of this paper, new research directions in the future are given.

Keywords: OpenCV, Kinematics analysis, Six-axis robot arm, Covid-19, Storage classification.

1. Introduction

The Coronavirus Disease 2019 (COVID-19) is widely spread among countries in current three years. it has three main ways of transmission: person-to-person, aerosol, and indirect transmission [1]. Every country has different policies to prevent and control it. However, the current logistics industry situation is that most storage classification still needs manpower to achieve, which might greatly increase the probability of COVID-19 transmission because the virus can attach to those storage items (indirect transmission). To solve this problem, using robot arms to do the storage classification is a relatively ideal solution to replace manpower since it can reduce the frequency of direct contact between people and storing items. Currently, robotic arms are widely used in highly repetitive work, such as production lines, and some high-risk environments [2]. Two of the main components of robot arm applications are robotic kinematics and computer vision system. Robotic kinematics is the basic model to simulate the body motions of a robotics mechanism without considering the forces and torques that cause the motions [3]. The computer vision system is the combination of the hardware design and software algorithm to simulate the same way as human vision [4].

Based on the current application of the robot arms, this paper proposes a kinematics model to combine the robot arm and computer vision technology to simulate the storage classification process in the logistics industry to reduce employees' involvement, which might cause COVID-19 transmission, in the storage classification. The paper will illustrate a six-axis robot arm kinematics model in the frame changes among space, robot, and camera, forward kinematics, and inverse kinematics and the combination with a computer vision system.

2. Technical Background

2.1. Kinematic Analysis

2.1.1. Forward kinematics analysis

Forward kinematics analysis means the pose and position of the robot end-effector in a given uniform coordinate system can be calculated by the parameters of links and joints of the robot [5].

Mathematically, the representation of the end-effector status $T_{end-effector}^{base}$ can be calculated by the multiplication of the transformation matrices T_i^{i-1} [6].

$$T_{end-effector}^{base} = T_1^{base} * T_2^1 * ... * T_{end-effector}^{n-1} \quad (1)$$

T_i^{i-1} represents the transformation for the robots from state $i - 1$ to state i . Base means the base frame (robot frame). End-effector means the position and pose of the device or tools connected at the end of the robot arm.

2.1.2. Inverse kinematics analysis

Inverse kinematics analysis is the opposite process of forward kinematics analysis [5]. Given the end-effector position of the robot, by combining the parameters of links and the corresponding configuration space, inverse kinematics analysis can calculate the variables of each joint of the robot [5]. Two main algorithms to do inverse kinematics analysis are the numerical inverse kinematics algorithm and the geometrical inverse kinematics algorithm. This paper will illustrate how to apply the geometrical algorithm in the robot arm model.

2.2. Open Computer Vision (OpenCV)

This paper will demonstrate the utilization of OpenCV to construct the computer vision system. OpenCV is a public computer vision repository with Python, which can be used to capture videos, manipulate images, and track objects [7]. OpenCV repository supplies several high-level application programming interfaces (API) to help people conveniently apply the desired computer algorithms, such as choosing the specific filters to track the target objects and using the specific color mode to be compatible with different types of cameras.

3. Experiment and Analysis

3.1. Hardware

To implement the robot arm model for the storage classification under COVID-19 prevention and control, in this study, a six degree-of-freedom (DOF) robot arm (UR3, figure 1) and a web camera, Logitech C300, are combined to implement the kinematics model proposed in the project. UR3 is a robot with an effective load of 3kg and a weight of 11kg [8]. All joints in UR3 can rotate 360 degrees independently [8]. Logitech C300 is a 1.3-megapixel web camera with RGB color mode, which is a relatively common and easy-to-obtain camera.



Figure 1. UR3 robot arm

3.2. Software: OpenCV Application

To do the storage classification, it is necessary to recognize the storage items [9]. There are three steps to implement that by using the OpenCV public repository:

1) *Step 1*: For the different kinds of items, defining the different colors, which can represent the items themselves, can help classify during the recognition process and locate the desired position for each item, respectively.

2) *Step 2*: Converting the chosen color to hue-saturation-value (HSV) color space, which can one-to-one represent the hardware device RGB color model [10]. Then change the range of HSV three field values (hue, saturation, value) by $\pm 10\%$ to get the lower and upper bound of each color to reduce the error caused by the light source environment.

3) *Step 3*: Using the specific filter from the OpenCV repository to mask the noise during the recognition process and then using the *SimpleBlobDetector()* function to locate corresponding storage items and get their position in the camera frame.

3.3. Mathematical Model

The mathematical model consists of three parts, frame transformation, forward kinematics analysis application, and inverse kinematics analysis application [11].

3.3.1. Frame transformation

This project has three frames: camera frame, space frame, and robot frame. In order to do the simulation, it is necessary to unify these three coordinate systems. Converting the camera and space frames to the robot frames is convenient for doing the kinematics analysis.

A. Camera frame transformation

To convert the camera frame to the robot frame, it is necessary to convert it to the space frame first. After that, the general method, of converting the space frame to the robot frame can be applied to the converted camera frame to get the corresponding robot frame coordinates [12].

1) *Step 1 Camera Calibration*: In the camera view, the unit of the distance is pixel. However, in the real world, the unit of distance is meter. Therefore, camera calibration can find the ratio between the pixel and meter by using a calibration stick. Obtaining the ratio for calibration can be achieved by using the length of the stick in pixel under the camera view divides the length of the stick in meter in the real world.

2) *Step 2 Camera Frame Transform*: Because the camera is fixed above the workspace, the z coordinate of the objects can be measured directly in the robot frame. Therefore, converting the camera frame to the space frame is just transforming the x_c and y_c coordinates to the x_w and y_w coordinates by multiplying the corresponding transformation matrix T_c^w . (x_c, y_c) is the coordinate under the camera frame. (x_w, y_w) is the coordinate under the space frame. T_c^w is the transformation matrix that can convert the camera frame coordinate to the space frame coordinate.

B. Space frame transformation

Generally, converting the space frame to the robot frame is obvious since we can measure the offsets between each coordinate and then do simple calculations based on figure 2.

$$x_0 = x_w + 0.15 \quad (2)$$

$$y_0 = y_w - 0.15 \quad (3)$$

$$z_0 = z_w - 0.01 \quad (4)$$

3.3.2. Forward kinematics analysis application

1) *Step 1 Determine the zero configuration (initial position)*: Based on table 1 and figure 3, the zero configuration can be represented by a 4x4 homogeneous transformation matrix (M) corresponding to the initial position of the UR3 robot end-effector.

$$M = \begin{bmatrix} 0 & -1 & 0 & 390 \\ 0 & 0 & -1 & 401 \\ 1 & 0 & 0 & 215.5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5)$$

2) Step 2 Get the screw axes: Screw axis can be represented by:

$$S = (\omega, v), \quad \omega = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}, \quad v = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \quad (6)$$

Where ω is the joint axis and v can be represented by:

$$v = -\omega \times q \quad (7)$$

Where p is the position of the joint center with respect to the robot frame. Then convert S_i to the Lie group $SE(3)$ matrix form [6]:

$$[S_i] = \begin{bmatrix} [\omega_i] & v_i \\ 0 & 0 \end{bmatrix}, \quad [\omega_i] = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \quad (8)$$

3) Step 3 Reach the desired status of end-effector: After getting the screw axes, it is easy to calculate the $T_{end-effector}^{desired}$:

$$T_{end-effector}^{desired} = e^{[s_1]\theta_1} e^{[s_2]\theta_2} e^{[s_3]\theta_3} e^{[s_4]\theta_4} e^{[s_5]\theta_5} e^{[s_6]\theta_6} \quad (9)$$

Then, using the calculated transformation matrix $T_{end-effector}^{base}$ and the end-effector initial position M to get the desired status F :

$$F = T_{end-effector}^{desired} * M \quad (10)$$

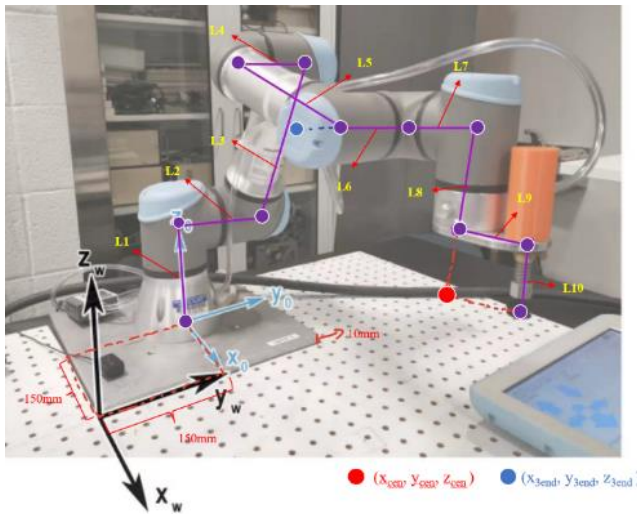


Figure 2. Space frame(black frame), robot frame (blue frame) and basic setting of UR

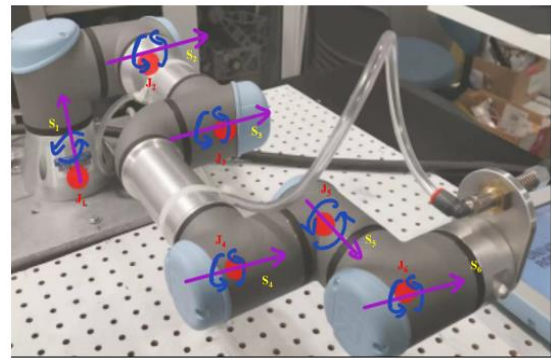


Figure 3. Initial position and screw axes of the six joints

Fig. 2: x_0 , y_0 and z_0 represent the coordinate in the robot frame. x_w , y_w and z_w represent the coordinate in the space frame. L1 to L10 represent the links between the joints. x_{3end} , y_{3end} and

z_{3end} (project endpoint), which lies with the L6 axis, represent a virtual point that locates at the same x-z plane as the L1 to help calculate the angle of joint 1(θ_1). x_{cen} , y_{cen} and z_{cen} (wrist center), which lies with the L9 axis, represent a virtual point that locates at the same x – y plane as the end-effector to help calculate the status of the end-effector.

Fig. 3: J_1 to J_6 represent each joint, S_1 to S_6 represent the screw axes for each joint.

Table 1. Links length(meter)

Link	Length(meter)
32.00 ETH	0.152
L1	0.120
L2	0.244
L3	0.093
L4	0.213
L5	0.083
L6	0.083
L7	0.082
L8	0.0535
L9	0.059
L10	0.152

Table 2. The list of ω_i , p_i , and v_i

	S_1	S_2	S_3
ω	(0,0,1)	(0,1,0)	(0,1,0)
p	(-0.15, 0.15, 0)	(-0.15,0, 0.162)	(0.094,0,0.162)
v	(0.15,0.15,0)	(-0.162,0, -0.15)	(-0.162,0,-0.094)
	S_4	S_5	S_6
ω	(0,1,0)	(1,0,0)	(0,1,0)
p	(0.307,0,0.162)	(0,0.26,0.162)	(0.39,0,0.162)
v	(-0.162,0,0.307)	(0,0.162,-0.26)	(-0.162,0,0.39)

3.3.3. Inverse Kinematics Analysis Application

To do the inverse kinematics, given the desired end-effector position, the system uses the geometric method to return the angle values of all joints(θ_1 to θ_6). For the stability of the classification process, θ_5 is always -90 degrees and θ_4 is set such that link 7 and link 9 are always parallel to the space x,y plane.

1) *Step 1:* It is necessary to convert the space frame coordinate ($xWgrip$, $yWgrip$, $zWgrip$) to the robot frame ($xgrip$, $ygrip$, $zgrip$):

$$xgrip = xWgrip + 0.15 \quad (11)$$

$$ygrip = yWgrip - 0.15 \quad (12)$$

$$zgrip = zWgrip - 0.01 \quad (13)$$

2) *Step 2:* According to figure 2, it is clear to see that z_{cen} equals $zgrip$. By using the top view of UR3, the relationship between (x_{cen} , y_{cen} , z_{cen}) and ($xgrip$, $ygrip$, $zgrip$) is:

$$x_{cen} = xgrip - L9 * \cos(yaw) \quad (14)$$

$$y_{cen} = ygrip - L9 * \sin(yaw) \quad (15)$$

$$z_{cen} = zgrip \quad (16)$$

yaw angle in figure 4 is the input parameter angle(usually 0) between L9 and the x-axis of an auxiliary coordinate system which is parallel to the robot base frame in the top view [10].

3) Step 3: Based on figure 4, to get θ_1 , it is necessary to get γ by using $\arcsin()$ function first.

$$\gamma = \arcsin\left(\frac{L_6+0.027}{|AB|}\right) = \arcsin\left(\frac{L_6+0.027}{\sqrt{(x_{cen}^2+y_{cen}^2)}}\right) \quad (17)$$

Then solving θ_1 can be implemented by utilizing $\text{atan2}()$ function, which can return the values in the correct quadrant.

$$\theta_1 = \text{atan2}(y_{cen}, x_{cen}) - \gamma \quad (18)$$

4) Step 4: Based on figure 4, ψ is critical to find θ_6 and ψ is the complementary angle of yaw.

$$\psi = \frac{\pi}{2} - \text{yaw} \quad (19)$$

$$\theta_6 = \theta_1 + \psi \quad (20)$$

5) Step 5: Using x_{cen} , y_{cen} , z_{cen} , and θ_1 can directly find the projected endpoint (x_{3end} , y_{3end} , z_{3end}).

$$\begin{matrix} x_{3end} \\ p_3^r = y_{3end} \\ z_{3end} \end{matrix} = T_c^r * p_3^c, T_c^r = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & x_{Cen} \\ \sin\theta_1 & \cos\theta_1 & 0 & y_{Cen} \\ 1 & 0 & 1 & z_{Cen} \\ 0 & 0 & 0 & 1 \end{bmatrix}, p_3^c = \begin{matrix} -L7 \\ -L6 - 0.027 \\ L8 + L10 \end{matrix} \quad (21)$$

6) Step 6 Get θ_2 , θ_3 , and θ_4 : Based on figure 5, to obtain β it is convenient to use $\text{atan2}()$ function.

$$\beta = \text{atan2}((Z_{3end} - L_1), |CD|) \quad (22)$$

ρ is the complementary angle of β .

$$\rho = \frac{\pi}{2} - \beta \quad (23)$$

Then, α and φ can be obtained by using the Law of cosine.

$$\alpha = \arccos\left(\frac{L_3^2+a^2-L_5^2}{2*L_3*a}\right), \varphi = \arccos\left(\frac{L_3^2+L_5^2-a^2}{2*L_3*L_5}\right) \quad (24)$$

Because the sum of the interior angles of a triangle is π rad, it is obvious to get δ .

$$\delta = \pi - \alpha - \beta \quad (25)$$

Therefore, θ_2 , θ_3 , and θ_4 are found.

$$\theta_2 = -(\alpha + \beta) \quad (26)$$

$$\theta_3 = \pi - \varphi \quad (27)$$

$$\theta_4 = -(\pi - \alpha - \beta - \varphi) = -(\theta_2 + \theta_3) \quad (28)$$

After doing those above steps, the whole 6 joint values (θ_1 to θ_6) of UR3 can be obtained by the inverse kinematics analysis.

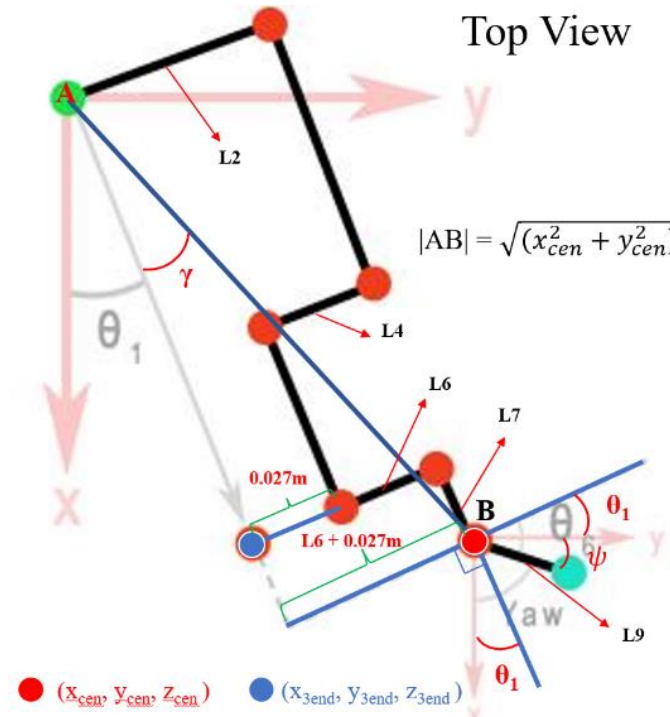


Figure 4. Top view of UR3

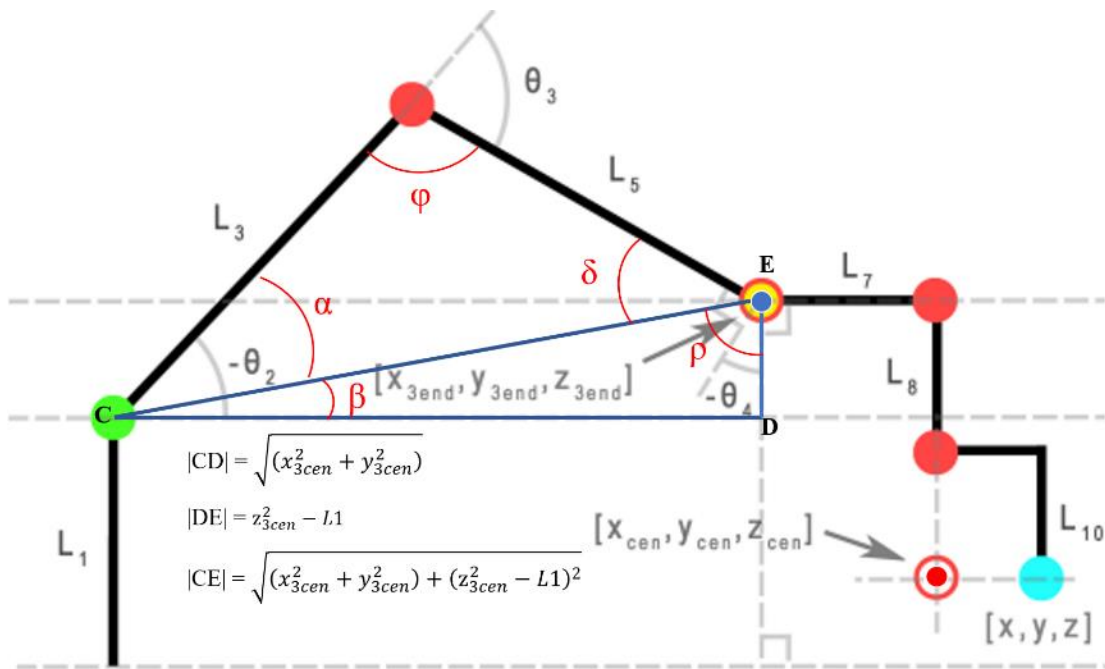


Figure 5. Side view of UR3

θ_1 to θ_6 are the joint angles of UR3. α , β , γ , ψ , ϕ , δ and ρ are the auxiliary angles that help to obtain θ_1 to θ_6 . AB, CD, DE, and CE are the auxiliary lines that help to obtain θ_1 to θ_6 .

3.4. Simulation

To apply this mathematical model to non-contact storage classification in the context of prevention and treatment of COVID-19, the whole process has several steps.

1) *Step 1:* Assigning the different storage items different colors can help to do the classifications and the camera can recognize the colors to locate the item positions.

2) *Step 2*: The system will use the item positions as the final position of the robot arm end-effector and pass them as the inverse kinematics analysis parameters.

3) *Step 3*: Inverse kinematics analysis will use the desired parameters to calculate each joint angles and pass the angles into the forward kinematics analysis.

4) *Step 4*: Forward kinematics analysis will make the robot arm to reach the desired position and grip the storage items. Then, the robot arm will do the storage classification.

4. Conclusion

In summary, this paper illustrates the mathematical model and simulation process to implement the non-contact storage classification based on mechanical arm in the context of prevention and treatment of COVID-19. The mathematical model and simulation process use UR3 robot and Logitech C300 web camera as the hardware support and utilize OpenCV public repository, forward and inverse kinematics analysis as the software development. This simulation can successfully achieve the desired functionality. Although this model can implement the classification, it still has several potential problems that needed to be solved, such as dynamically recognizing the size of the storage items in the camera view. In the future, more research and experiments are required to develop more comprehensive computer vision algorithm and more high efficiency system model.

References

- [1] Singh P K, Nandi S, Ghafoor K Z, et al. Preventing covid-19 spread using information and communication technology [J]. IEEE Consumer Electronics Magazine, 2020, 10 (4): 18 - 27.
- [2] Lee J D, Li W C, Shen J H, et al. multi-robotic arms automated production line[C]//2018 4th International Conference on Control, Automation and Robotics (ICCAR). IEEE, 2018: 26 - 30.
- [3] Waldron K J, Schmiedeler J. Kinematics [M]//Springer handbook of robotics. Springer, Cham, 2016: 11 - 36.
- [4] Feng X, Jiang Y, Yang X, et al. Computer vision algorithms and hardware implementations: A survey [J]. Integration, 2019, 69: 309 - 320.
- [5] Wen G, Xu L, He F. Offline kinematics simulation of 6-DOF welding robot[C]//2009 International Conference on Measuring Technology and Mechatronics Automation. IEEE, 2009, 2: 283 - 286.
- [6] Lynch K M, Park F C. Modern robotics [M]. Cambridge University Press, 2017.
- [7] Howse J. OpenCV computer vision with python [M]. Birmingham: Packt Publishing, 2013.
- [8] Lin C J, Li M Y. Motion planning with obstacle avoidance of an UR3 robot using charge system search [C]//2018 18th International Conference on Control, Automation and Systems (ICCAS). IEEE, 2018: 746 - 750.
- [9] Zuo J, Zhu X, Peng Y, et al. A new method of posture recognition based on WiFi signal [J]. IEEE Communications Letters, 2021, 25 (8): 2564 - 2568.
- [10] Shopa P, Sumitha N, Patra P S K. Traffic sign detection and recognition using OpenCV[C]//International conference on information communication and embedded systems (ICICES2014). IEEE, 2014: 1 - 6.
- [11] Okuonghae D, Oname A. Analysis of a mathematical model for COVID-19 population dynamics in Lagos, Nigeria [J]. Chaos, Solitons & Fractals, 2020, 139: 110032.
- [12] Ghiță A Ș, Gavril A F, Nan M, et al. The AMIRO social robotics framework: deployment and evaluation on the pepper robot [J]. Sensors, 2020, 20 (24): 7271.