Proceedings of the 6th International Conference of Control, Dynamic Systems, and Robotics (CDSR'19) Ottawa, Canada – June 6-7, 2019
Paper No. CDSR 110
DOI: 10.11159/cdsr19.110

Assessment of Visual Servoing Techniques for Target Localization

Khushali Saraiya[1], Dippal Israni[1], Parth Goel[2]
[1]U & P. U. Patel Department Of Computer Engineering CSPIT, CHARUSAT, Changa, India
17PGCE024@charusat.edu.in;dippalisrani.ce@charusat.ac.in
[2]Computer Engineering Department, DEPSTAR
CHARUSAT, Changa, India
er.pathgoel@gmail.com

Abstract - Robotics is advancing rapidly in the research area of designing, controlling and building new robots for domestic, commercial and military applications. In addition to these, it also becomes easy to control objects in dangerous locations like bomb defusal, mines and exploring shipwrecks. Visual Servoing (VS) is a technique that controls a robot and gives the motion to achieve the field of technologies such as bionomics and security guard. In robotics, the most important part is to place cameras in the workspace such that each and every place is visible during the whole task. In Visual Servoing, the challenge is to achieve kinematics that gives ease to pick and place objects. This paper represents an overall idea, state of the art techniques and architecture for camera control techniques and robot control techniques. This paper also highlights examples related to forward kinematics and inverse kinematic using Matlab.

***Keywords*:** Visual Servoing, Position Based Visual Servoing, Image-Based Visual Servoing, Kinematics.

1. Introduction

Robotics is gradually upgrading manufacturing industry to engineer the environment so that it will simplify the detection and retrieval of the item. The vision that is primarily based on absolute control or visual servoing based is used as a solution in robot organization. This also helps to grow the dexterity and intelligence of the robot system. Visual servoing is a technique that controls the motion of the robot.

Visual servoing is used within the self-sufficient automobile systems, cellular robotics, pick and drop packages and to evaluate the reference frame with the streaming body [1]. Similarly, this approach is utilized in path finding algorithms, object tracking [2], monitoring and item reorganization [3].

For any visual servoing technique there are two important aspects 1) Camera technique to be implemented for applying vision to robot 2) Control technique that helps required the object to reach the desired location.

2. Related Work

In visual servoing, the camera position is the most imperative part to make the object reach its destination. Similarly, this technique is also beneficial to reach the destination, based on the distance of an object with respect to the digital camera.

2.1. Camera Techniques

There are three primary methods to place cameras in VS: Eye-in hand, Eye-to hand, and Hybrid.

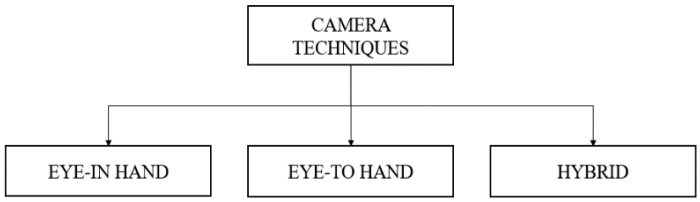


Fig. 1: Camera Techniques for VS.

2.1.1. Eye-In Hand

In eye- In this approach, an advanced camera is associated with the moving hand viewing the relative position of the objectives. The digital camera is expressed as the eye with a restricted motion due to the inflexible setup at the end-effectors [4]. This procedure is actualized in the applications whose objective is to procure enhanced target redesign and review from the end position. For instance, finding an error from an actual reference shape to the current shape and to identify the shape and name of the object [5].

2.1.2. Eye-to hand

In this method, the cameras are constant in single vicinity. Streaming records provide object movement and also hint position and direction of the object. In this scenario, two or more cameras are utilized to cover the workspace outline. This makes it easy to design a workspace model and to instruct the object. This technique is mostly implemented in the clinic and drug stores where robots are used to pick and drop objects [6].

2.1.3. Hybrid

This technique is the combination of eye-in-hand and eye-to-hand techniques which enables clean and smooth tuning of the object position with the robot motion in the workspace. V. Lippiello et al. proposed an application [7] based on the hybrid camera position. The motto is to utilize the eye-to-hand camera for evaluating the robot tool posture and an eye-in- hand camera as an information hotspot for direction estimation.

2.2. Control Techniques

Achieving the end factor and implementing the robot motions in the actual workspace is a critical task. To overcome the problem control technique is implemented robots. These control techniques can be categorized as:

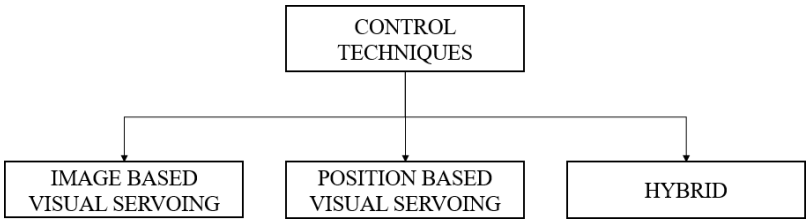


Fig. 2: Control Techniques for VS.

2.2.1. Image-Based Visual Servoing (IBVS)

In IBVS, the errors among the initial and desired positions of the characteristic points in the picture plane are computed. The feature points are managed to transport to the favored positions at the 2D image aircraft [8]. IBVS

applications prove to be beneficial while locating an item in the workspace by finding the distance between objects and cameras. The prior knowledge of camera position and camera parameter are essential [9]. There is a significant number of methods to compute this distance wherein the dominant are triangulation technique, structured light technique and time of flight technique [10]. A straightforward flowchart is demonstrated in figure3 to ascertain the distance of individuals in the video. IBVS has many potential applications such as computer animation [11], to build a smart guard [12] and so on.

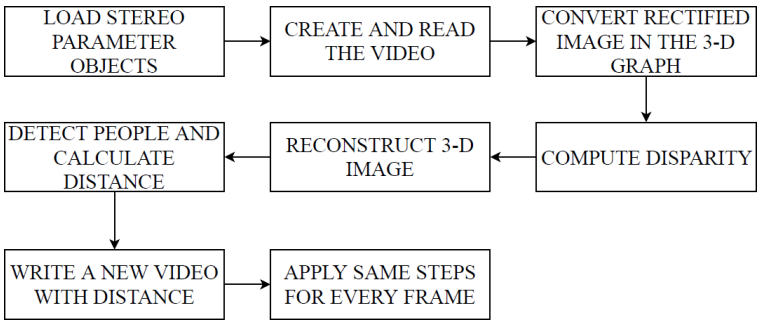


Fig. 3: System Diagram of IBVS.

Brief information of System Diagram for IBVS -

Load stereo parameter objects- In this step, the parameters and geometric relationship of two cameras are loaded. Here, parameters are an Intrinsic parameter that allows mapping between cameras coordinates and pixel coordinates of the image and an Extrinsic parameter indicates the location and orientation of the camera with respect to world frame.

Create and read the video- After loading stereo parameter next step is to create a new frame in which video is streaming.

Convert rectified image in the 3-D graph- In this step, the image is covered into a 3-D graph. To achieve that first, it has to be checked whether the image is a blunder or not. If it is not then it targets one pixel and checks mistake or development in all edges. In addition, it converts the image into a rectified image via anaglyph method. Anaglyph is defined as a stereoscopic photograph with the two images superimposed and printed in different colors (usually it is green and red) which produces a stereo effect.

Compute disparity- The result of redressed stereo pictures take one pixel from the reference image as viewed and take another pixel as a source of perspective. The difference between these two pixels called separation of the camera. This is calculated from the edge of cameras. The separation discovered is called difference and it is proportional to the removal of the related point.

Reconstruct 3-D image-Every single pixel is converted into the 3-D model. After converting the 3D image, the image is transformed into a point cloud object. Here, Point cloud object is a set of points that define a three-dimensional model of an object. All points are identified by its position in space as well the color of characteristics. After converting in the 3-D model, streaming point cloud viewer is created to visualize points in the 3D model.

Detect people and calculate distance–In this step object as well as people from the video is detected. Also, the detected objects are labeled using the distance from the camera. Depth algorithm is used to find the distance between cameras and object.

Write a new video with distance- After detecting and calculating distance on data, the image is imposed in a new video and all the above-mentioned steps are repeated.

2.2.2. Position Based Visual Servoing

In PBVS, the error between the preliminary pose and the preferred pose within the 3D workspace is computed. This error is utilized by a manipulated law for the positioning mission of a robot manipulator [8].

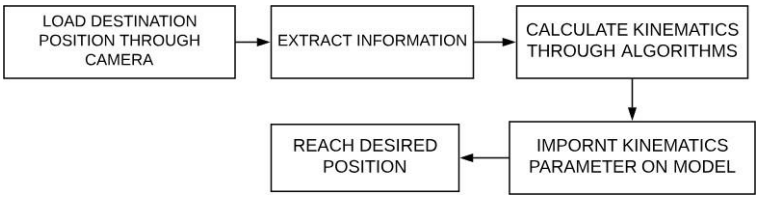


Fig. 4: System Diagram of PBVS.

Kinematic is a study of motion without considering the force. However, the Kinematics of connected links can be conveniently studied using the frame concept of coordination of frames and transformations. There are two major types of robot joints one is a rotation and another one is a translation. Consequently, every link has a homogeneous transformation matrix relative to its first joint and it describes the orientation and position of the other end of the link. The transformation matrix of the end factors represented by using three position components along with the base point and three rotations of the base body axes. Cartesian space is creating using these six vectors as well joint space is created using joint vectors. Through, Cartesian and Joint space it discovers end-effector position identified with each other by forward and inverse kinematics.

Robot-Kinematics is the connection between the joint variable space and the position and the orientation of the end-effector of a robot arm. This described analytically in motion geometry with respect to a fixed reference coordinate system. There are three ways to find kinematics,

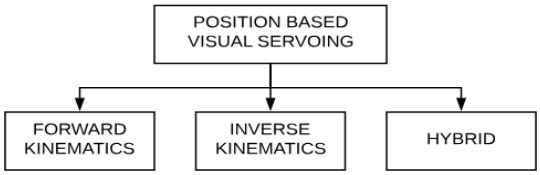


Fig. 5: Position based VS.

i. Forward Kinematics (FK) - In forward kinematics object position is estimated based on the values of Link parameters and Joint angles.

ii. Inverse Kinematics (IK) - Find object position using the link parameter and position of the object.

iii. Hybrid Kinematics -In this approach both FK and IK techniques are used to meet the desired goal. In addition, it is sub classified into 2 ½ D servoing and Motion partition-based technique.

Forward Kinematics (FK) - The main idea of FK is to discover the End-Effector (EE) from the given joints. There are two types of joints which are considered when kinematics are performed (I) Rotation and (II) Prismatic. In addition, Forward kinematic is a transformation from the joint space to Cartesian space. It calculates a unique position and orientation of the end-effector for the given set of joint angles. Denavit-Hatenberg Parameters (DH parameter) is one of the common and easy methods to implement joints and link in the Cartesian plane as well to find the Forward Kinematics, Jacobian matrix can be applied. However Jacobian Inverse algorithm is used to calculate IK problems.

***Denavit-Hatenberg Parameters (DH parameter*) -**To represent the robot model D-H parameter is used. Robot Skelton is a combination of joint, link and wrist. For every n joint, n+1 links are available. Where the 0^{th} link is the main link between the manipulator and n^{th} link becomes a carrier of end-effector [13]. Also, each joint is attached via a link to the previous joint. In the D-H parameter, frame describes the pose of a link frame with respect to the previous link frame. Robot joint is mainly described through four parameters-

i. Offset length d: offset along the previous z to the common normal.
ii. Joint angle Θ: angle about the previous z, from old x to new x.
iii. Link Length r: length of the common normal.
iv. Twist angle α: angle about common normal, from old z-axis to new z-axis.

So, the equation to find DH parameter to below robot arm can be estimated as mentioned in equation (1).

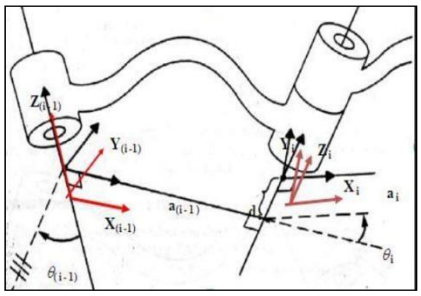


Fig. 6: D-H Matrix.

$$A_i = \mathrm{Rot}_{x,\Theta(i)}, \mathrm{Trans}_{x,d(i)}, \mathrm{Trans}_{x,a(i)}, \mathrm{Rot}_{x,\alpha(i)} \quad (1)$$

$$\begin{bmatrix} \cos\theta i & -\sin\theta i \cos\alpha i & \sin\theta i \sin\alpha i & \alpha i \cos\theta i \\ \sin\theta i & \cos\theta i \cos\alpha i & -\cos\theta i \sin\alpha i & \alpha i \sin\theta i \\ 0 & \sin\alpha i & \cos\alpha i & di \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

Jacobian matrix for a forward kinematics-Jacobian matrix of a manipulator that describes the velocity of the system and how it affects the end effectors' position.

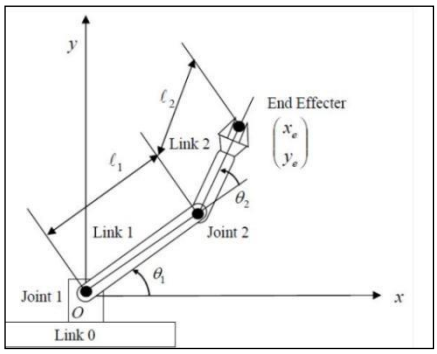


Fig. 7: Link Position in Cartesian Frame.

Here, the end effector is X_e and Y_e which make joint angles θ_1 and θ_2
So the equation for Jacobian becomes,

$$dx = J^* \, \mathrm{dq} \quad (3)$$

$$\text{Where, } dx = \begin{pmatrix} dx_e \\ dY_e \end{pmatrix}, \mathrm{dq} == \begin{pmatrix} d\theta_1 \\ d\theta_2 \end{pmatrix} \text{ and} \quad (4)$$

$$J = \begin{bmatrix} -l_1 \sin\theta_1 - l_2 \sin(\theta_1 + \theta_2) & -l_2 \sin(\theta_1 + \theta a_2) \\ l_1 \cos\theta_1 - l_2 \cos(\theta_1 + \theta_2) & l_2 \cos(\theta_1 + \theta a_2) \end{bmatrix} \quad (5)$$

Equation (5) represents the Jacobian matrix in which,l_1 is link1's length and l_2 is link2's length.

Roboarm constantly moves through the end effector. Hence, X_e position and angles change continuously. This effect represents the Jacobian equation as,

$$\frac{dx_e}{dt} = J * \frac{dy}{dx} \quad (6)$$

$$\text{Or} \vartheta_e = J * \dot{q} \quad (7)$$

As per the equation (7), the Jacobian maps the relation between joint velocities and end effector velocities.

Inverse Kinematics-In Inverse Kinematic (IK) technique, the desired position of the adjunct utilizing an end-effector and every revolution joints are kept them at required area [14]. However, IK can have multiple solutions, unique solution or no solution. Multiple can occur based on the choice of the target place, DOFs of the kinematic chains, hand poster and shortest path [15].

Jacobian Inverse for Inverse Kinematics - There are many traditional methods to solve Inverse Kinematic such as PSO (Particle Swarm Optimization) [16], FABRIK (Forward and backward Inverse Kinematics) [17], CCD (Cyclic Coordinate Descent) [18] and Jacobian Inverse [19]. All the above-listed algorithms are useful but Jacobian Inverse (JI) takes less time in comparison to other methods.

The solution of the Jacobian Matrix is,

$$\frac{dx_e}{dt} = J * \frac{dq}{dt} \quad (8)$$

$$\text{Or } \vartheta_e = J * \dot{q} \quad (9)$$

Also, Inverse Kinematic is represented as,

$$q = \dot{J}^{-1} * \vartheta_e \quad (10)$$

Where $\dot{q}$ is end effector and is joint velocity.

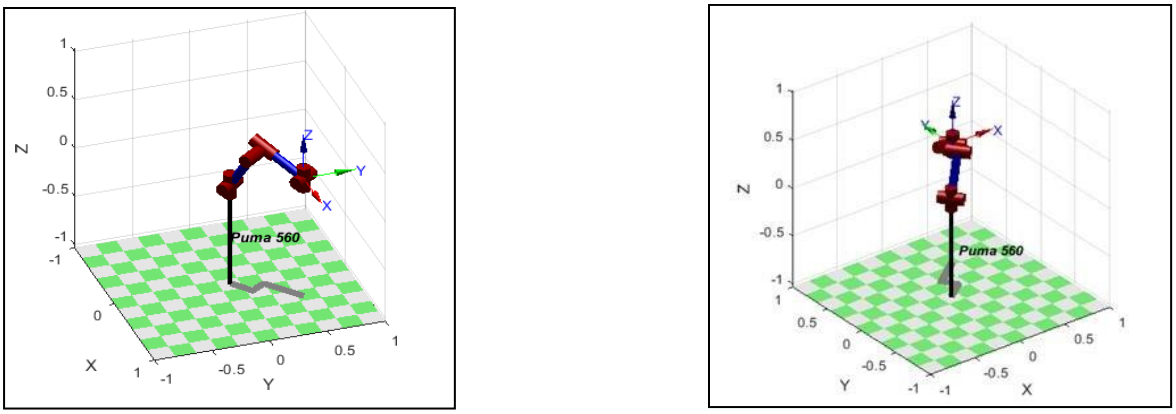
To establish persistent velocity of the robot arm, the joint velocity of all the paths is continuously estimated. In reality, all the parameters of the Jacobian change constantly as the robot and its configuration varies as it moves. This results in rapid changes in numerical values. Hence, there is a requirement to calculate the Jacobian at every possible time step. In order to calculate a sufficient number of joint velocities/sec, the process/technique should be rapid.

3. Experimental Result

In MATLAB tool mathematical and graphical representations become easily enable. The robotic toolbox of MATLAB tool provides robot techniques such as kinematics, dynamic, and trajectory generation [20].

In this section, the Inverse Jacobian algorithm is showcased using co-simulation of Robotic Toolbox and MATLAB tool.

Example: 1-In this problem, the PUMA560 object is used as a robo arm. Here, the initial position of the arm is (T_1) = (0.6,-0.5, 0.0), desired position is (T_2) = (0.4, 0.5, 0.2) and Cartesian path becomes (T_1, T_2 ,50). The roboarm has to move from its initial position to despite red position in total 50 steps.

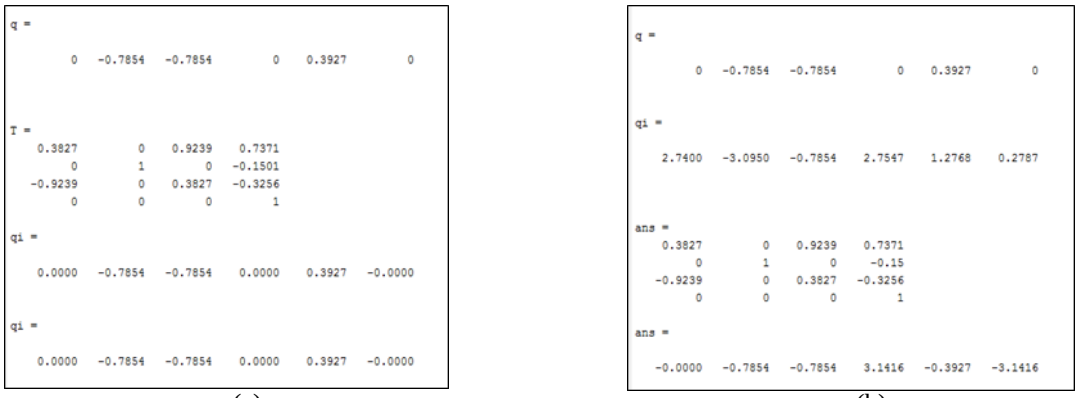


(a) (b)

Fig. 8: (a) Initial position of puma 560 (b) Desire position.

Above discussed problem has more than one solution for IK which are shown in Figure 8(a) and 8(b).

In Figure 9(a) *T* indicates Forward Kinematic solution for *q*. where *q*= [0 –pi/4 –pi/4 0 pi/8 0] and *qi* is a solution of inverse kinematics with respect to *T*. Here, *qi* is a solution of inverse kinematics. After estimate IK it converts into the Cartesian frame.



```
q =
         0   -0.7854   -0.7854         0    0.3927         0

T =
    0.3827         0    0.9239    0.7371
         0         1         0   -0.1501
   -0.9239         0    0.3827   -0.3256
         0         0         0         1

qi =
    0.0000   -0.7854   -0.7854    0.0000    0.3927   -0.0000

qi =
    0.0000   -0.7854   -0.7854    0.0000    0.3927   -0.0000
```

(a)

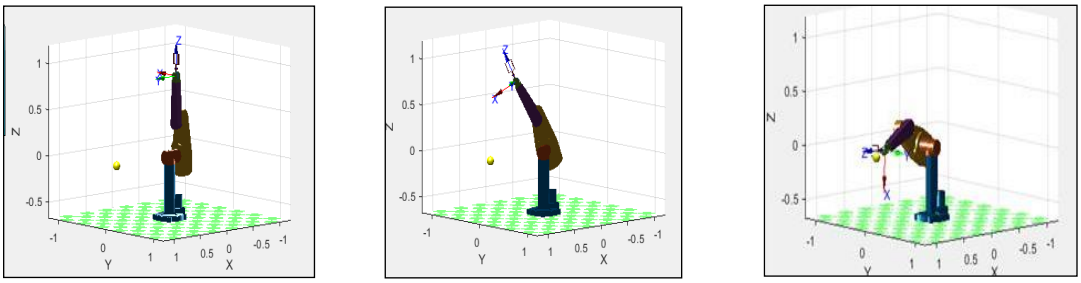
```
q =
         0   -0.7854   -0.7854         0    0.3927         0

qi =
    2.7400   -3.0950   -0.7854    2.7547    1.2768    0.2787

ans =
    0.3827         0    0.9239    0.7371
         0         1         0   -0.15
   -0.9239         0    0.3827   -0.3256
         0         0         0         1

ans =
   -0.0000   -0.7854   -0.7854    3.1416   -0.3927   -3.1416
```

(b)

Fig. 9: (a) Calculation of given position (b) Best solution for T_1 and T_2 in the Cartesian frame.

However, a simple Cartesian method gives the best shortest path for *T*1 and *T*2. In addition, matrix rows indicates time per steps of robo arm and column illustrates an angle of joint.

Example-2:



(a) (b) (c)

Figure 10: (a) Move roboarm to desire object (b) Move towards desire position (c) Y-axis reach to the destination point.

Figure 10(a) represents p560 robot arm in the Cartesian frame. This is implemented in the MATLAB tool and robot toolbox kit. example 2, P560 robot is used and implemented in MATLAB. Figure 10(a), 10(b) and 10(c) illustrate how robot arm p560 move its initial position to given ball position with the use of the DH matrix, forward kinematics, and inverse kinematics techniques.

4. Conclusion

This paper represents an overall idea of Visual Servoing along that (1) Camera Position (2) Control Techniques. For any application of Visual Servoing, both categories are mandatory. This paper highlights several of the camera as well as control techniques. The paper also shows cases result of various experiments of Visual Servoing. The Camera position is used to cover the workspace and to estimate the directions for the robot accordingly. Control technique is used for guiding a robot more accurately with the use of camera position footprint. It also checks for the obstacles within the path and chooses the more precise and short path for localizing the target. This paper gives an overall assessment of various techniques in the field of Visual Servoing.

Acknowledgments

The authors would like to thank CHARUSAT Space Research and Technology Centre (CSRTC) for providing a platform to conduct the research work.

References

[1] M. A. Putra, E. Pitowarno and A. Risnumawan, "Visual servoing line following robot: Camera-based line detecting and interpreting," *International Electronics Symposium on Engineering Technology and Applications (IES-ETA)*, Surabaya, pp. 123-128, 2017.

[2] D. Israni and H. Mewada, "Feature Descriptor Based Identity Retention and Tracking of Players under Intense Occlusion in Soccer Videos," *International Journal of Intelligent Engineering and Systems*, vol. 11, no. 4, pp. 31-41, 2018.

[3] S. V. Upadhyaya, D. Israni, K. Jasani and A. Shah, "A novel approach to precisely control linear movement of sensor by motor using microstepping," *IEEE Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER)*, Mangalore, pp. 242-246, 2016.

[4] A. Muis and K. Ohnishi, "Eye-to-hand approach on eye-in-hand configuration within real-time visual servoing," In *IEEE/ASME Transactions on Mechatronics*, vol. 10, no. 4, pp. 404-410, Aug. 2005.

[5] Piepmeier, J. Armstrong, and H. Lipkin. "Uncalibrated eye-in-hand visual servoing," *the International Journal of Robotics Research*, vol. 22, no. 10-11, pp. 805-819, 2003.

[6] R. C. Luo, S. Chou, X. Yang and N. Peng, "Hybrid Eye-to-hand and Eye-in-hand visual servo system for parallel robot conveyor object tracking and fetching," in *40th Annual Conference of the IEEE Industrial Electronics Society(IECON)*, Dallas, TX, USA, pp. 2558-2563, 2014.

[7] V. Lippiello, B. Siciliano and L. Villani, "Eye-in-Hand/Eye-to-Hand Multi-Camera Visual Servoing," in *44th IEEE Conference on Decision and Control*, Seville, Spain, pp. 5354-5359, 2005.

[8] I. Ha, D. Park and J. Kwon, "A novel Position-Based Visual Servoing approach for robust global stability with feature points kept within the Field-of-View," in *11th International Conference on Control Automation Robotics & Vision (ICARCV)*, Singapore, pp. 1458-1465, 2010.

[9] À. Santamaria-Navarro and J. Andrade-Cetto, "Uncalibrated image-based visual servoing," in *IEEE International Conference on Robotics and Automation*, Karlsruhe, pp. 5247-5252, May 2013.

[10] Khusheef, A. Shany, "Vision-Based Object Distance Measurement Using Mono and Stereo Vision," *Iraqi Journal of Mechanical and Material Engineering*, vol. 17, no. 4, pp. 734-744, 2017.

[11] N. Courty and E. Marchand, "Computer animation: a new application for image-based visual servoing," in *IEEE International Conference on Robotics and Automation (ICRA.)*, Seoul, South Korea, vol. 1, pp. 223-228, 2001.

[12] L. Li, B. Wang, B. Li, P. Xiao, W. Wang and Y. Li, "The application of image based visual servo control system for smart guard," in *10th IEEE International Conference on Control and Automation (ICCA)*, Hangzhou, pp. 1342-1345, 2013.

[13] J. Lee, G. Park, J. Shin and J. Woo, "Industrial robot calibration method usingdenavit — Hatenberg parameters,"

in *17th International Conference on Control, Automation and Systems (ICCAS)*, Jeju, pp. 1834-1837, 2017.
[14] Z. Bhatti, A. Shah, F. Shahidi, "Forward and Inverse Kinematics Seamless Matching Using Jacobian," *Sindh University Research Journal (Science Series)*, vol. 45, no. 2, pp. 387-392, 2013.
[15] Andreas Aristidou, Joan Lasenby, Yiorgos Chrysanthou, and Ariel Shamir, "Inverse Kinematics Techniques in Computer Graphics: A Survey," *Computer Graphics Forum*, vol. 37, no. 6, pp. 35-58, 2018.
[16] Nizar Rokbani and Adel M.Alimi, "Inverse kinematics using particle swarm optimization, a statistical analysis," *Procedia Engineering*, vol. 64, pp. 1602-1611, 2013.
[17] Renzo Poddighe, "Comparing FABRIK and neural networks to traditional methods in solving Inverse Kinematics," 2013.
[18] B. Kenwright , "Inverse kinematics–cyclic coordinate descent (CCD)," *Journal of Graphics Tools*, vol. 16, no. 4, pp. 177-217, 2012.
[19] Samuel R. Buss, "Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods," *IEEE Journal of Robotics and Automation*, vol. 17, no. 1, pp. 1-19, 2004.
[20] Peter I. Corke, "A computer tool for simulation and analysis: the Robotics Toolbox for MATLAB," in *Proc. National Conference Australian Robot Association*, pp. 319-330, 1995.