# Obstacle Avoidance Method for Highly Redundant Robotic Arms

# *Obstacle Avoidance Method for Highly Redundant Robotic Arms*

**Firas S. Hameed[1, 2], Dr. Hassan M. Alwan[1], Dr. Qasim A. Ateia[1]**

1 University of Technology, Mechanical Engineering Department, Baghdad, Iraq.

2 Email: firassubhyhameed@gmail.com

## Abstract

Obstacle avoidance is an important concept to be considered when a robotic system is installed in an environment. Obstacles within the working space of a robotic system can prevent the robot from performing the tasks assigned to it properly; hence, the designer of the robotic system must program the robot to follow an emergency strategy that enables it to avoid any contact with a probable obstacle anywhere anytime within the working space of the robot. For mobile robots this task is relatively easier to accomplish since this type of robots have the flexibility to change their routes and take alternatives that are free of obstacles. For fixed robotic arm manipulators the situation is much more complex. In this paper we are only concerned with obstacle avoidance techniques and present a real-time obstacle avoidance method for fixed robotic arms that combines the benefits of both global and local techniques in that it can be used for both static and dynamic environments with a reduced computational effort.

**Key words:** Robotics, Inverse Kinematics, Obstacle Avoidance, Critical Points, Triangulization, Echo Sensors, Arduino Board.

## Introduction

Obstacle avoidance techniques are classified into two categories, namely; global and local techniques. Global techniques are applied on the robotic arm joint space in such a way that they guarantee a collision-free path following performance within the robotic arm's working environment. The drawback of such techniques is that they assume a static working environment in which all the obstacles are fixed and pre-located resulting in a minimized computation effort to avoid them.

Local techniques, on the other hand, are applied when the environment is not static, i.e. the existence of the obstacles is not known in advance. These methods rely on a set of sensors to locate the proximity of those obstacles to the mechanism. These methods require high computational effort to re-adjust the configuration of the robotic arm while trying to avoid an obstacle.

Many researchers were interested in the aspect of obstacle avoidance for robotic manipulators. D. Puiu and F. Moldoveanu [1] presented a joint trajectory planning strategy for a redundant manipulator. The controller modifies the joint configuration as a moving object (an obstacle) is getting closer to the robotic arm. The object proximity was detected by a visual system around the robotic manipulator. E.J. Solteiro Pires et al [2] used a multi-objective genetic algorithm (MOGA) to optimize the configurations of the robotic arm such that the optimized configuration ensures obstacle avoidance while performing the task. Jing Liu et al [3] used Cylindrical Bounding Box to simplify the geometry of the obstacle and, by using optimization, they can consider the major axis of the box and its radius as the dimensions of the obstacle to avoided. Tse-Ching Lai et al [4] proposed an obstacle avoidance method based on Non Uniform Rational B-splines (NURBS). A safe distance between the end-effector of robot arm and obstacle is considered to avoid the collision. Tianjian Hu et al [5] developed what is termed a backward quadratic search algorithm (BQSA) as another option for solving IK problems in obstacle avoidance. The BQSA detects possible collisions based on the root property of a category of quadratic functions, which are derived from ellipse-enveloped obstacles and the positions of each link's end-points. The algorithm executes a backward search for possible obstacle collisions, from the end-effector to the base, and avoids obstacles by utilizing a hybrid IK scheme, incorporating the damped least-squares method, the weighted least-norm method and the gradient projection

method. In this paper, an obstacle avoidance method was applied on highly redundant robotic arms. The method relays mainly on the existence of a solution to the inverse kinematics problem for that type of robotic arm.

## Description of the Method

In a previous work [6] we presented a new method to solve the inverse kinematics problem for redundant robotic arms. It is based on the idea of configuring the robotic arm to adjust its joints on a selected polynomial such that all the links are aligned on the polynomial starting from the base up to the end effector. When the robotic arm works in a free environment the second order polynomial of the form of equation (1) is the best choice based on which the links of the robotic arm are to be configured. This is demonstrated in Figure 1.

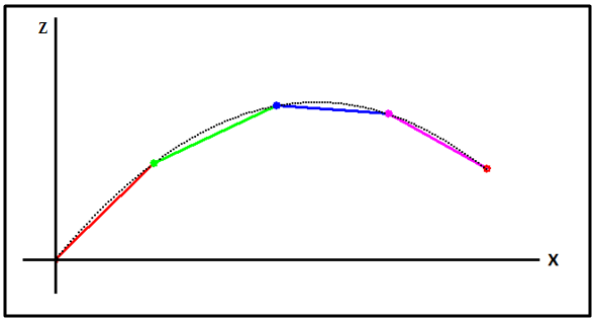


*Figure (1): Robotic arm links (the colored lines) are fit on a second order polynomial (the dotted curve) in an obstacle-free working space.*

$$Z(X) = a_o + a_1 X + a_2 X^2 \quad (1)$$

When an object gets closer to the arm, a set of sensors installed at certain locations on the robot body, will measure the distance of the robotic arm from that object. Based on the information acquired by the sensors the controller determines a number of "critical" points that lie closest to the arm referenced to the location of the detected obstacle. These critical points locate inflexion points on the suggested curve hence determine the overall shape (the degree) of the configuration polynomial that enables the robotic arm to avoid a possible collision with the object. Figure 2 shows the effect of one object proximity to the robotic arm. Point A represents a critical point that suggests a third-order polynomial expressed in equation (2) as the best configuration of the robotic arm that enables it to avoid this obstacle.

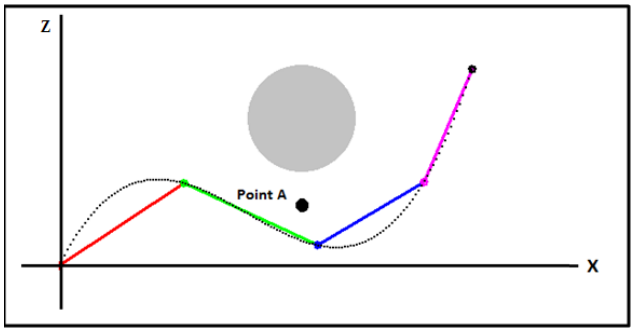


*Figure (2): Working Space with One Obstacle. A third order polynomial is generated based on the position of the critical point* A.

$$Z(X) = a_o + a_1 X + a_2 X^2 + a_3 X^3 \quad (2)$$

If the number of objects that are close to the robotic arm increases, the degree of the configuration polynomial increases as well. Figure (3) shows a fourth-degree configuration polynomial used to configure the robotic arm as

it gets close enough to two objects simultaneously. The sensors generate a critical point for each object and the degree of the polynomial is decided accordingly by the controller.

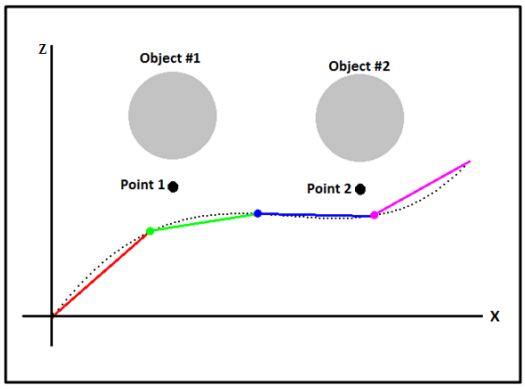


*Figure (3): Working Space with Two Obstacles.*

Let's take an example to best describe how the method works. Consider the working space shown in Figure (4) in which a robotic arm is to follow a semicircular 3D path that happens to pass a region in which a column (an obstacle) is suspended from the ceiling. As long as the obstacle is far enough (at regions R1 and R3), the second order polynomial is used to solve the inverse kinematics of the robotic arm. Once the obstacle gets closer to the arm by entering region R2, the sensors locate the critical point (which will be defined later) and trigger the controller to switch into using the third order polynomial that enables the robot to avoid the collision. Figure (5) shows the three regions R1, R2 and R3 that the robotic arm should move into while performing the task. The obstacle is represented as a solid black circle inside the region R2.

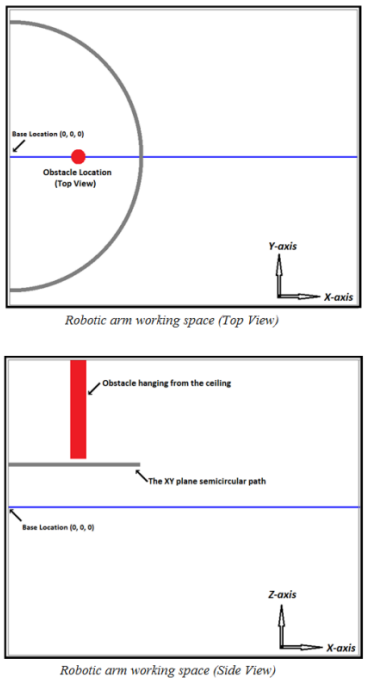


*Figure (4): Top and side views of the robotic arm working space.*

The position profile of the robotic arm as it tracks the path is shown in Figure (6). The figure represents a record of angle variations of each link which is derived from solving the inverse kinematics problem of the robotic arm during the motion. Since the path chosen is a semicircular and centered at the base, the joint angles of all the links, but not the base, are constant within R1 and R3.

As Figure 6 shows, the position profile of each link is smooth and continuous within each of the defined regions. However, at the instants of transition between the adjacent regions, the profile shows discontinuities that are large enough to cause very high torque variations of all the joints at these transition instances. These high torque spikes can cause damage to one or more joints due to high inertia forces resulting from sudden change of the links' angular position in a short period of time not to mention the possibility of losing the arm stability during the motion.

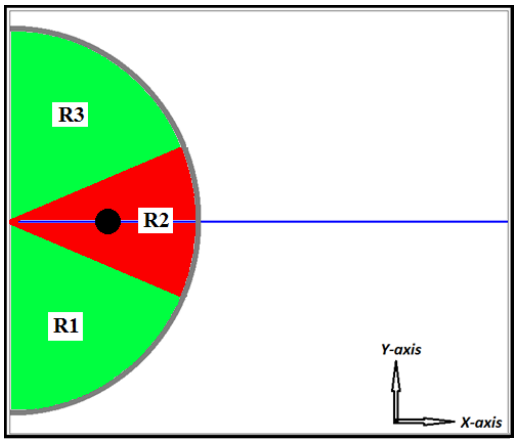


*Figure (5): Working space regions. R1 and R3 are the regions where the arm is far from the obstacle. R2 is the region of obstacle proximity to the robotic arm.*

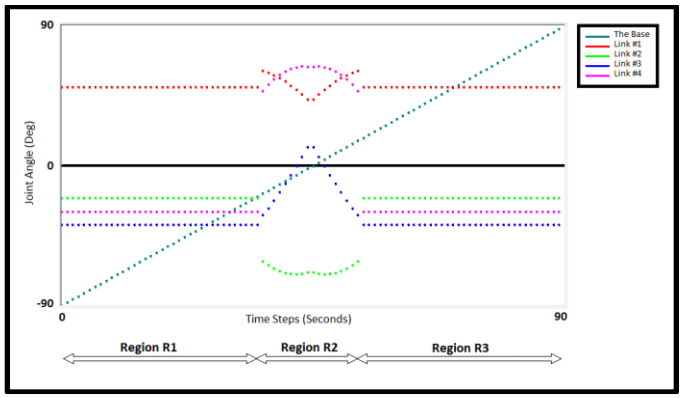


*Figure (6): Position profile of joint angle of each link while the robot tracks the given semicircular path.*

In order to guarantee a stable performance of the robotic arm during its motion, the controller must deal with those profile discontinuities generally in one of two ways:

1- The controller can increase the time step at the instant of any region-to-region transition in such a way that the speed of the corresponding motor(s) is reduced. This can highly lower the acceleration at the joint which, accordingly, lowers the inertia forces at the joints. This approach is followed when the task assigned to the robotic arm requires that the path should be preserved.

2- The position profile can be linearized to get rid of the discontinuity points and re-gain a smooth position profile which guarantees a stable arm performance during the motion. However, following this linearizing process will definitely affect the end-effector positioning which makes it impossible for the robotic arm to follow the path assigned to it. This approach is followed when the task is path-independent. Figure (7) shows the modified position profile after applying the linearizing process.

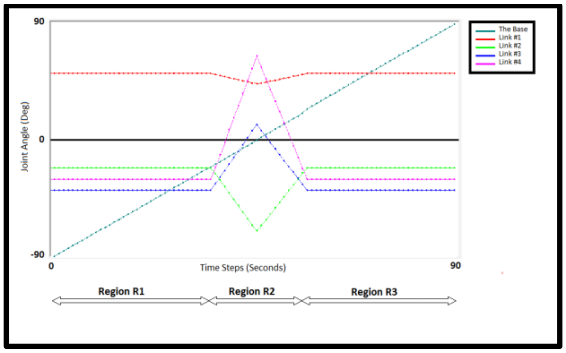


*Figure (7): Linearized position profile*

To linearize the position profile in R2, we make use of the fact that the variations in all of the joint angles are related to the angle of the base $\theta_o$. In the example, R2 is divided into two sub regions, the first starts at $\theta_o = -20^o$ and ends at $\theta_o = 0^o$(in the direction of the obstacle). The second part starts at $\theta_o = 0^o$ and ends at $\theta_o = +20^o$. Let $\theta_{-20}^{(n)}$ be the displacement of the $n^{th}$ joint at $\theta_o = -20^o$ and $\theta_{20}^{(n)}$ be the displacement of the $n^{th}$ joint at $\theta_o = 20^o$. Also let $\theta_{Obs}^{(n)}$ be the displacement of the $n^{th}$ joint at that value if the base angular position in the direction of the obstacle (which equals to zero in our example). The linearization equation is:

$$\theta^n = \frac{(\theta_{obs}^n - \theta_{-20}^n)}{20}\theta_o + \theta_{obs}^n \qquad (3)$$

$$\theta^n = \frac{(\theta_{20}^n - \theta_{obs}^n)}{20}\theta_o + \theta_{obs}^n \qquad (4)$$

Equation (3) is valid for $-20 \leq \theta_o \leq 0$, and equation (4) is valid for $0 \leq \theta_o \leq 20$. n=1 to 4 for both equations.

The effect of the linearization process on the original semicircular path is shown in Figure (8) below. Since the modification, or the linearization, process is applied only for the points within region R2, the effect of the process appears only in that region. The other regions did not change in shape after the linearizing process was applied.

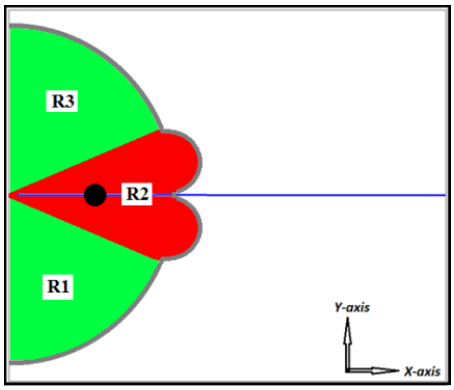


*Figure (8): The semicircular path after applying the linearizing process.*

## Detection of an Obstacle in 3D Space

The ultrasound echo sensor with effectual angle of 15° is the most commonly used for distance measurement. It works by sending a sequence of ultrasound waves directed in front of the sensor and if an object exists it will reflect them back to the sensor. The time period between the instant of sending the ultrasound waves and that when they reflect back to the sensor is a measure of the distance since the sound speed is previously known. The distance

measured by a single sensor does not give information about the location of the object in the 3D space, which makes the sensor useful only as a proximity indicator. Due to the effectual angle of 15°, the sensor view is a cone whose head lies on the sensor itself with its base lying away from the sensor.

In order to locate an object in the 3D space, we propose a method in which a set of sensors located at previously known points referenced to a certain origin making simultaneous measurements for the same obstacle and use the information to locate the object in the 3D space. Consider the coordinate system shown in Figure (9).

In Figure (9) the points A, B and C are the locations of three echo sensors that constitute the 3D distance measurement set. The plane X = 0 of the frame coincides with the body of the robot and the points A(0, $d_1$, -$d_2$), B(0, -$d_1$, -$d_2$) and C(0, $d_1$, $d_2$) are the locations of three distance sensors and are referenced to one origin point O(0, 0, 0). Because of the small value of the sensors' effectual angle the distances $d_1$ and $d_2$ must be chosen carefully such that all the three sensors make a successful measurement.

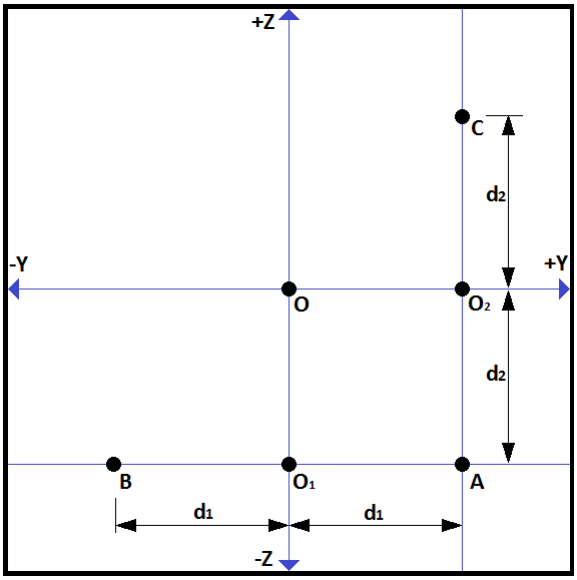


*Figure (9): The frame on which the echo sensors are fixed.*

Point P(X, Y, Z) is the 3D spatial position of an object detected simultaneously by the three sensors and it is required to calculate X, Y and Z in order to know where that objects is located with respect to the robotic arm. Choosing point O as the reference for the measurement has the advantage that the distance OP is the true distance between the obstacle and the link on which the sensors are attached regardless of the orientation of that link with respect to the obstacle.

In order to make the calculations easier, let $d_1 = d_2 = d$. When an object gets close enough from the sensors, each sensor will "see" the obstacle from its own angle, which means that each sensor will measure a different distance from the other two in the set. The object allocation procedure begins as follows:

1- The sensor at A measures the distance AP = $R_a$.
2- The sensor at B measures the distance BP = $R_b$.
3- The sensor at C measures the distance CP = $R_c$.
4- Now, point P(X, Y, Z) is a common point for three spatial scalene triangles, namely, Δ APB, Δ APC, and Δ BPC. See Figure (10).

5- For the scalene APB of which AB is the base, the median that starts at P and ends at $O_1$(0, 0, -d) which is the middle point of AB has a length found by the equation [7]:

$$R_1 = \overline{PO_1} = \frac{\sqrt{2R_a^2+2R_b^2-4d^2}}{2} \qquad (5)$$

6- For the scalene APC with AC is the base, the median that starts at P and ends at $O_2$(0, d, 0), the middle point of AC, has a length given by the equation:

$$R_2 = \overline{PO_2} = \frac{\sqrt{2R_a^2+2R_c^2-4d^2}}{2} \qquad (6)$$

7- For the scalene BPC with BC is the base, the median that starts at P and ends at O(0, 0, 0), the middle point of BC, has a length given by the equation

$$R_3 = \overline{PO} = \frac{\sqrt{2R_b^2+2R_c^2-8d^2}}{2} \qquad (7)$$

8- Since $R_1$ represents the distance between point $O_1$ and point P, the following expression for the $R_1$ also applies, that is:

$R_1^2 = X^2 + Y^2 + (Z + d)^2$

Or,

$R_1^2 = X^2 + Y^2 + Z^2 + 2dZ + d^2$ (8)

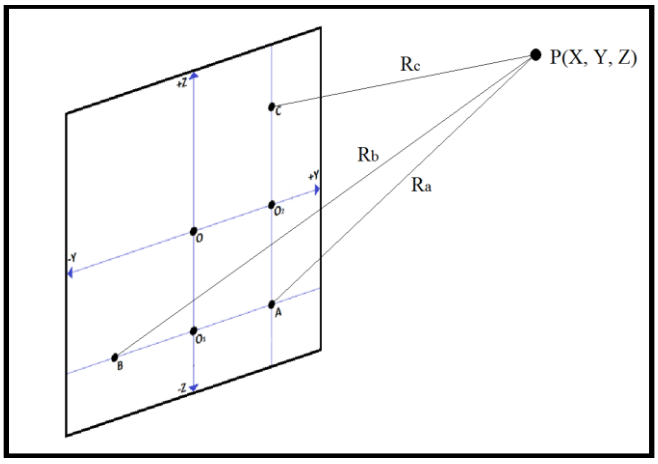


*Figure (10): The frame on which the echo sensors are fixed in a 3D view.*

9- Similarly for $R_2$, which is the distance between point $O_2$ and point P, whose length can be expressed as:

$R_2^2 = X^2 + (Y - d)^2 + Z^2$

Or,

$R_2^2 = X^2 + Y^2 + Z^2 - 2dY + d^2$ (9)

10- Since $R_3$ originates at O(0, 0, 0), its length can be expressed as:

$R_3^2 = X^2 + Y^2 + Z^2$ (10)

11- Equation (10) can be used to simplify equations (8) and (9) yielding:
$R_1^2 = R_3^2 + 2dZ + d^2$, and

$R_2^2 = R_3^2 - 2dY + d^2$

from which

$$Y = \frac{R_3^2 + d^2 - R_2^2}{2d} \quad (11)$$

$$Z = \frac{R_1^2 - R_3^2 + d^2}{2d} \quad (12)$$

And by using equation (10),

$$X = \sqrt{R_3^2 - Y^2 - Z^2} \quad (13)$$

This way, the exact position of the object, represented by point P(X, Y, Z), is determined. When the distance between O(0, 0, 0), which is located on the robot body, and the object becomes smaller than a predefined threshold, a critical point is created. A *critical point* is defined as that point located half way along the line connecting point P on the object and the local origin of the sensors set O, (See Figure 10).

The creation of a critical point is the trigger for the kinematics of the robot to turn from the parabolic configuration (Eq 1) to the third-degree-polynomial (Eq 2) passing through those three points (the base of the robot, the created critical point and the required end effector position) to avoid hitting the object while the robotic arm is in motion.

In certain cases, however, the size of the obstacle may be too large compared to the size of the measuring set (like when the end effector moves towards a wall for example). In such cases, the measurement set will saturate because all the sensors will measure the same distance making the above equations inapplicable. Then, the only alternative for the controller to avoid hitting that obstacle is to "freeze" or stop the robotic arm when it gets dangerously close to that kind of obstacles.

## Conclusion

Obstacle avoidance is an emergency strategy taken by the robotic arm to prevent any chance of collision between a robotic arm and an obstacle that happens to enter the working space of the robot. Many techniques are available to accomplish obstacle avoidance each with its own advantages and disadvantages. The method presented in this work enables a highly redundant robotic arm to reconfigure its overall geometrical shape passing around the obstacle and continue performing the task safely. A set of three echo sensors in certain arrangement where used to make the distance measurement. The sensors are connected to an Arduino board as the micro controller.

## References

[1] D. Puiu and F. Moldoveanu (2011) "Real-time Collision Avoidance for Redundant Manipulators". 6th IEEE International Symposium on Applied Computational Intelligence and Informatics, May 19–21, 2011.

[2] E.J. Solteiro Pires, P.B. de Moura Oliveira and J.A. Tenreiro Machado (2007) "Manipulator Trajectory Planning Using a MOEA". Applied Soft Computing 7 (2007) 659–667.

[3] Jing Liu, Ruimin Liu, Xin Shen and Linmin Meng (2018) "Research on Obstacle Avoidance of Space Manipulators Based on Cylindrical Bounding Box Model". Proceedings of 2018 IEEE International Conference on Mechatronics and Automation August 5 - 8, Changchun, China

[4] Tse-Ching Lai, Sheng-Ru Xiao, Hisayuki Aoyama, and Ching-Chang Wong (2017) "Path Planning and Obstacle Avoidance Approaches for Robot Arm". Proceedings of the SICE Annual Conference 2017, September 19-22, 2017, Kanazawa University, Kanazawa, Japan

[5] Tianjian Hu, Tianshu Wang, Junfeng Li and Weiping Qian (2016) "Obstacle Avoidance for Redundant Manipulators Utilizing a Backward Quadratic Search Algorithm". International Journal of Advanced Robotic Systems, 2016

[6] Firas S. Hameed, Hasan M. Alwan and Qasim A. Ateia (2019) "Novel Approach to Solve the Inverse Kinematics Problem for a Multi-Degree-of-Freedom Robotic Arm. Journal of Engineering and Applied Sciences (2019 Volume 14 Issue 13).

[7] Nathan Altshiller-Court, "College Geometry: An Introduction to the Modern Geometry of the Triangle and the Circle" Dover Publications, Inc. Mineola, New York 2007.

[8] Mehmet Ismet Can Dede, Omar W. Maaroof and Enver Tatlicioglu (2016) "A New Objective Function for Obstacle Avoidance by Redundant Service Robotic Arms". International Journal of Advanced Robotic Systems (2016).

[9] Haobin Shi , Jialin Chen, Wei Pan, Kao-Shing Hwang and Yi-Yun Cho (2018) "Collision Avoidance for Redundant Robots in Position-Based Visual Servoing". IEEE SYSTEMS JOURNAL (2018).