

# Field Programmable Gate Arrays Based Real Time Robot Arm Inverse Kinematic Calculations and Visual Servoing

Bariş Çelik<sup>1</sup> , Ayça Ak<sup>2</sup> , Vedat Topuz<sup>2</sup> 

<sup>1</sup>Department of Electric Electronic Engineering, Marmara University School of Engineering, Istanbul, Turkey

<sup>2</sup>Vocational School of Technical Sciences, Marmara University, İstanbul, Turkey

**Cite this article as:** B. Çelik, A. Ak, V. Topuz. "Field Programmable Gate Arrays Based Real Time Robot Arm Inverse Kinematic Calculations and Visual Servoing", *Electrica*, vol. 18, no: 2, pp. 143-150, 2018.

## ABSTRACT

Reliability and precision are very important in space, medical, and industrial robot control applications. Recently, researchers have tried to increase the reliability and precision of the robot control implementations. High precision calculation of inverse kinematic, color based object recognition, and parallel robot control based on field programmable gate arrays (FPGA) are combined in the proposed system. The precision of the inverse kinematic solution is improved using the coordinate rotation digital computer (CORDIC) algorithm based on double precision floating point number format. Red, green, and blue (RGB) color space is converted to hue saturation value (HSV) color space, which is more convenient for recognizing the object in different illuminations. Moreover, to realize a smooth operation of the robot arm, a parallel pulse width modulation (PWM) generator is designed. All applications are simulated, synthesized, and loaded in a single FPGA chip, so that the reliability requirement is met. The proposed method was tested with different objects, and the results prove that the proposed inverse kinematic calculations have high precision and the color based object recognition is quite successful in finding coordinates of the objects.

**Keywords:** Video processing, robot arm control, inverse kinematic, CORDIC Algorithm, field programmable gate arrays

## Introduction

Image sensors on robots are able to provide more information than other sensors. This advantage of the image processing in robot control makes it attractive [1]. Traditional robot vision systems consist of an image sensor and a single CPU-based computer [2]. However, this type of vision system is not able to process large amounts of video data because it works with one processor and sequential software. Field Programmable Gate Arrays (FPGA), or in other words fine-grained reconfigurable architecture that can perform any processing operation at a hardware level, satisfactory real-time requirements of image and video processing and take on more intensive tasks than general microprocessors [3].

Robot arms are currently used in operations which should have high accuracy, repeatability, and stability. Although many industrial and space robots can work with high precision, researchers try to develop robotic manipulators which have more precision than old ones. Especially in areas such as the design of grippers, control systems, and image processing systems for more precise robotic applications [4]. Self-moving robots have been made possible with improved technology. They are able to do work that people cannot afford or which is located in places to which people have no access [5]. The robot needs to receive information about what is happening around so that it can make decisions for itself and move itself. Image processing is more instructive than other types of sensor. However, image processing systems are expensive and also involve a great deal of image information which takes a long time to process. This causes the main system to run slowly. FPGAs are faster at image processing and cheaper than other technologies. Therefore, FPGA technology is appropriate for use in self-controlled robots.

Red, green, and blue (RGB), YUV, YCbCr and hue saturation value (HSV) have been widely used in the traditional image processing field. Commonly, within RGB space, little intensity change

## Corresponding Author:

Bariş Çelik

## E-mail:

bariscelik@marun.edu.tr

**Received:** 07.11.2017

**Accepted:** 10.05.2018

© Copyright 2018 by Electrica

Available online at

<http://electrica.istanbul.edu.tr>

**DOI:** 10.26650/electrica.2018.49877

of any colour class will result in a diagonal movement of RGB cube, and this can lead to wrong classification of colour. Several researchers have approved that the H component of HSV space is more insensitive to lighting variety and closer to the human vision system [6].

In the literature there is as yet no work which combines inverse kinematic calculation and image processing in real time in a single FPGA chip. However, both these applications have separately been built [7]. Presents digital color enhancement under non-uniform lighting conditions. Log-domain computation is used for converting RGB to HSV to eliminate all multiplications, divisions and exponentiations. Although log-domain computation requires significantly less power and space, this method is not reliable. Proposes a navigation system for an unmanned ground vehicle [8]. Four image processing nodes which are mounted on the ceiling are used. All videos in the main station are combined and locations of the mobile robots are defined. System on chip converts the RGB video to HSV color space to recognize the mobile robot CPU, or system on chip has been used to compute inverse kinematic of robot arm in [9-12]. However usage of these systems reduces the reliability of the robot control systems. Presents FPGA based fixed-point computation of robot kinematics. In this work look-up table method is used to compute trigonometric functions [13]. The proposed system works fast. Inverse kinematic computation takes only 4.5  $\mu$ s but output error of the system is greater than which computes inverse kinematic of robot manipulator with floating-point numbers [14]. Uses CORDIC algorithm for computing the inverse kinematic with small error [14]. However the proposed system contains micro-control-unit (MCU) to operate all of the joints of the robot arm at same time. FPGA can make this parallel operation, so MCU is unnecessary.

Presents a stereo computer vision algorithm intended to control a robotic arm and the final target is to have all the system running on an FPGA [15]. Proposes FPGA Based Compact and Efficient Full Image Buffering [16]. A whole framework is presented to develop dynamic visual servoing systems embedded in an FPGA using an optimizing method based on parallel processing and pipeline technique [17].

In this study, we have tried to give the sense of sight to the robotic arm which can operate in medical, space, nuclear and industrial fields. In these contexts, reliability and precision is very important. A little error in robot control application could cause a disaster. For the reasons described above, we chose double precision floating-point number format and CORDIC algorithm to compute inverse kinematic of the robot arm. To identify colours, RGB video is converted to HSV colour space which is more suitable for classification of colours than the others (RGB, YUV etc.). The parallel motor control section is also designed for controlling the servomotors of the robot arm in FPGA. The inverse kinematic calculation, the image progressing and the robot control section are realized with FPGA chip so the whole system is reliable, reconfigurable, and cheap, with the added advantage of having high performance.

The rest of this paper is organized as follows. Section 2 contains a description of experimental setup. Section 3 describes FPGA based image processing. Section 4 describes inverse kinematic calculation of the robot arm and parallel servo motor control. Section 5 presents the results of these implementations. Finally, Section 6 states the conclusions.

## Description of Experimental Setup

The hardware of the system includes DE2-115 FGPA development card, robot manipulator with 5 DOF (Degrees of Freedom), CCD (Charge Coupled Device) camera and VGA (Video Graphics Array) monitor. DE2-115 development card has Altera Cyclone IV 4CE115 FPGA chip, ADC (Analog Digital Converter) for video in DAC (Digital Analog Converter) for video out and switches which are used in this work. The manipulator has 6 servo motors to actuate 5 joints and gripper.

The RGB video, taken from the ADC, is converted to HSV colour space in FPGA, thus enabling the desired colour and its locations to be determined for every pixel. The exact medium point of the object is found by averaging all the determined pixel locations. Then the angles of the robot arm are calculated by FPGA with inverse kinematic method. The motor control unit takes the angles and generates PWM (Pulse Width Modulation) signals for servo motors of the robot arm. When all joints finish their motions, the end effector grips the object and carries it any desired location. All operations are in real time and the processed video can be seen in the monitor connected to video out of the development card. The block diagram of the system is shown in Figure 1.

## Image Processing

### Acquisition and displaying of video information

The analog camera gives us a PAL (Phase Alternating Line) video. In order to convert the video from analog to digital an ADV7180 located on the DE2-115 is used. ADV7180 auto-

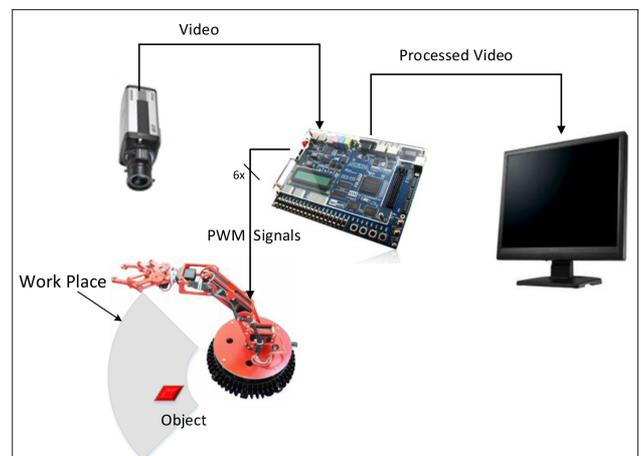


Figure 1. Experimental setup

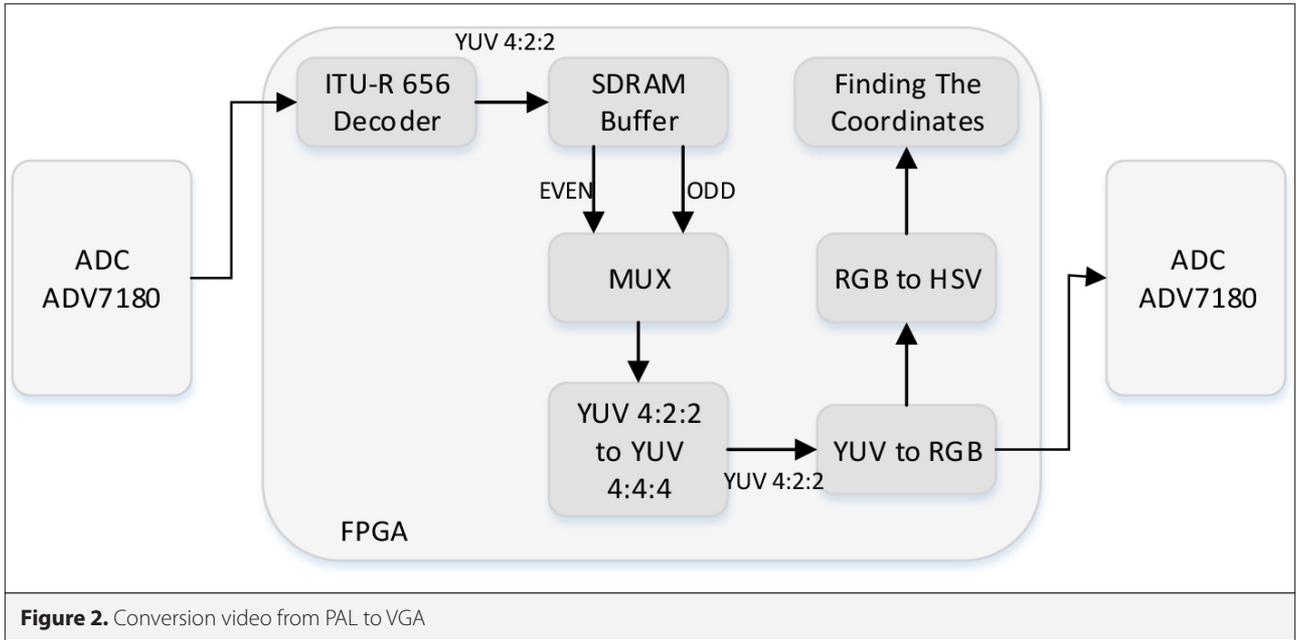


Figure 2. Conversion video from PAL to VGA

matically performs video format detection. It detects NTSC, PAL or SECAM video formats and converts them from PAL to 4:2:2 component video data compatible with the 8-bit ITU-R BT.656 interface standard. In PAL video format the even numbered lines are scanned before the odd-numbered lines. The odd-numbered lines need to be saved and then combined with the even-numbered lines. The PAL video format uses YUV 4:2:2 colour space and the VGA uses RGB colour space. It is necessary to convert these colour spaces. All conversion processes are shown in Figure 2.

### Conversion Red, green, and blue (RGB) to hue saturation value (HSV)

The object can be segmented using its colour information if it has a unique colour. We can detect the image using a suitable reference value, and the object having a unique colour can easily be distinguished from other objects in different colours. We have used the HSV colour space to determine the desired object because HSV colour space is more suitable for colour segmentation and intuitive to the human vision system.

Each pixel is represented by three components in the HSV colour space: Hue, saturation and value. The nature of colour is defined by a hue component. The brightness of the colour is represented by the 'value' component. The range of the hue component is from 0 to 360. The "Saturation" and "Value" range from 0 to 1. The advantage of the HSV is the small variability of the hue component with different lighting conditions so the illumination system does not need to have specific features. The following formulas are used to transform RGB to HSV.

$$\text{Max} = \max(R, G, B) \quad (1)$$

$$\text{Min} = \min(R, G, B) \quad (2)$$

$$\text{Value} = \text{Max} \quad (3)$$

$$\text{Saturation} = \begin{cases} 0 & , \text{Max} = 0 \\ \frac{100(\text{Max}-\text{Min})}{\text{Max}} & , \text{Max} \neq 0 \end{cases} \quad (4)$$

$$\text{Hue} = \begin{cases} 0 & , \text{Max} = \text{Min} \\ \frac{60(G-B)}{\text{Max}-\text{Min}} & , R = \text{Max} \\ \frac{60(B-R)}{\text{Max}-\text{Min}} + 120 & , G = \text{Max} \\ \frac{60(R-G)}{\text{Max}-\text{Min}} + 240 & , B = \text{Max} \end{cases} \quad (5)$$

The equations of conversion from RGB to HSV are redesigned for FPGA implementation. Hue, Saturation and Value range from 0 to 360, from 0 to 100 and from 0 to 255 respectively. They should be parallel for real time video processing. The parallel realization of the formulas can be seen in Figure 3.

### Finding the coordinates of the object

After the conversion of the colour space to HSV the desired colour is compared to each pixel colour in one picture for recognition of the object. The desired colour can be red, green, cyan, purple etc. For this study red was chosen as the desired colour. The centre of the red object should be found so that it can be grasped by the robot arm. Finding the centre of the object is expressed by the equations below:

$$M_x = \frac{x_1 + x_2 + x_3 + \dots + x_n}{n} \quad (6)$$

$$M_y = \frac{y_1 + y_2 + y_3 + \dots + y_n}{n} \quad (7)$$

In PAL video format 25 pictures are scanned in one second. After ending the line scanning for one frame the mean of red pixel locations is calculated to find the centre of the object. In the above equations  $x_{1r}$ ,  $x_{2r}$ ,  $y_{1r}$ , and  $y_{2r}$  represent the location of the red

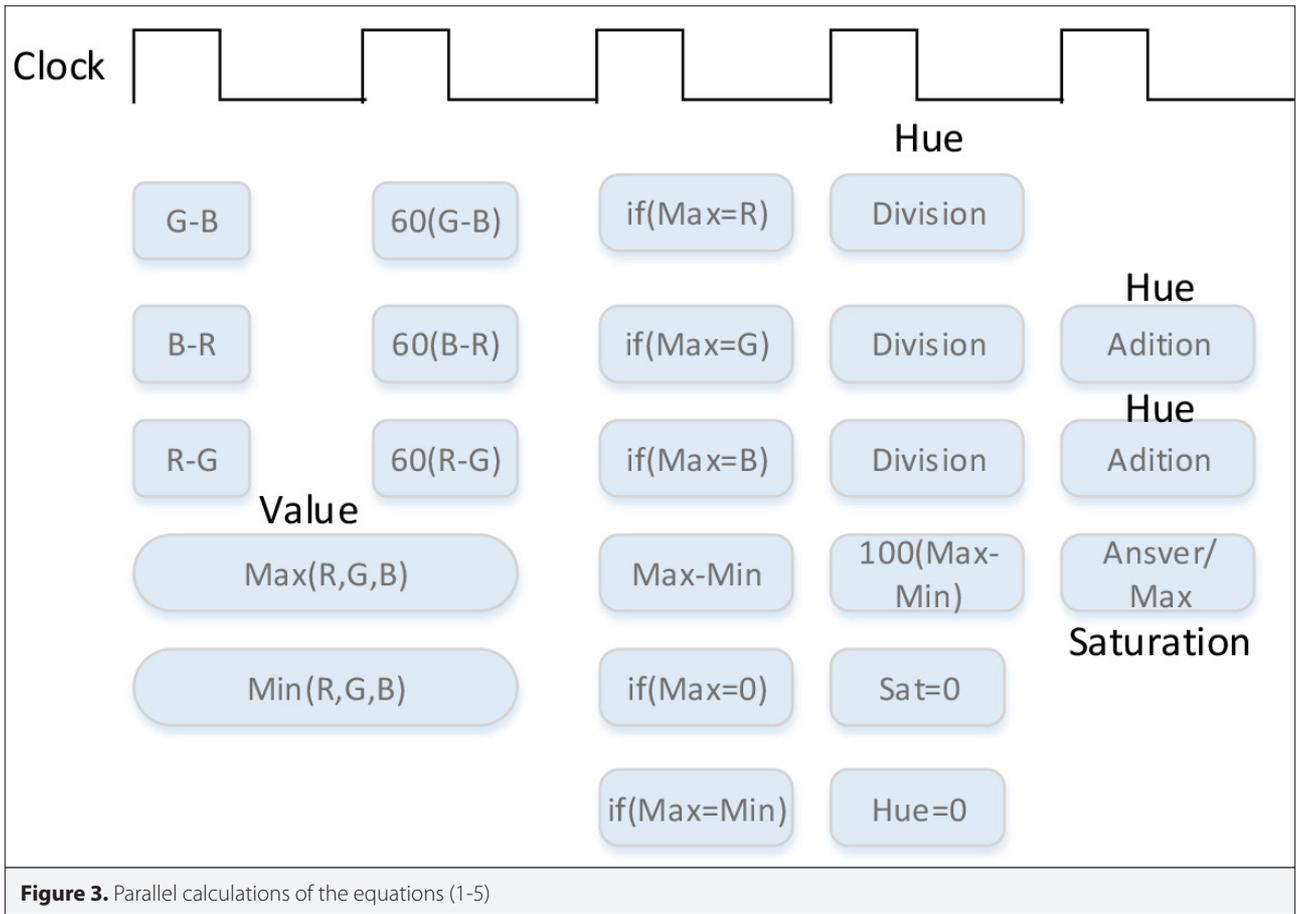


Figure 3. Parallel calculations of the equations (1-5)

pixels,  $M_x$  and  $M_y$  represent the mean of the red pixel locations. Horizontal and vertical line counters give us  $X_n$  and  $Y_n$  values, so FPGA determines the centre of the red object. If the red pixels in the picture are not sequential, they cannot be incorporated into equations (6) and (7) because two or more objects could be located in the workplace. Starting from the left side of the workplace each centre of the object is computed one by one.

The workplace can be seen in Figure 4. The z-axis is equal to zero at all times because the object is at ground level. The workplace begins at the base of the robot arm and ends at the furthest point that the robot arm can lie. The x axis starts at 11 cm and extends to 32 cm. The y-axis starts at 0 cm and extends to 14 cm in negative and positive directions.

After finding of the centre pixel location of the object  $M_x$  and  $M_y$  values are converted to x and y coordinates. To find the exact x and y coordinates of the object, the reference lines in the monitor should be matched with the edges of the work place. The reference lines in the monitor are depicted in Figure 4. If the reference lines are not matched accurately with the edges of the work place, wrong x and y coordinates will be found. The object width is found by counting the red pixels in the horizontal centre line, thus the gripper of the robot arm can take the object without being harmed.

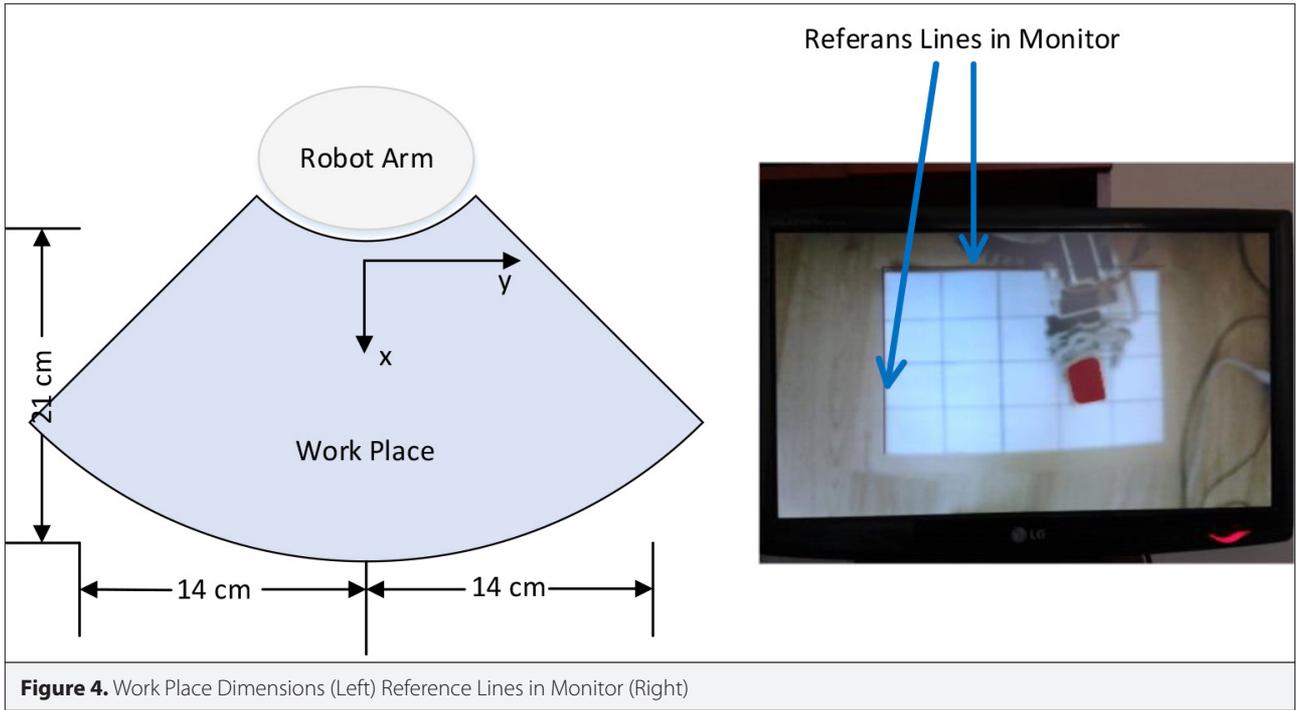
## Robot Control

### Inverse Kinematics Problem

Robot kinematics analysis can be divided into forward kinematics analysis and inverse kinematics analysis. The goal of the forward kinematic solution is to compute the position of the end-effector from specified values for the joint parameters. The inverse kinematic is used to determine the joint parameters that provide a desired position of the end-effector [18]. The forward or direct kinematics comprises the motion of the gripper of the robot according to the global coordinate system. The link coordinate system of the robot arm is shown in Figure 5 using Denavit-Hartenberg order.

Denavit-Hartenberg parameters of the robot arm are given in Table 1. In the table,  $\theta_i$  is the joint angle from the  $X_{i-1}$  axis to the  $X_i$  axis about the  $Z_{i-1}$  axis,  $d$  is the distance from the origin of the  $(i-1)$ th coordinate frame to the intersection of the  $Z_{i-1}$  axis along the  $Z_{(i-1)}$  axis,  $\alpha_i$  is the offset distance from the intersection of the  $Z_{i-1}$  axis with the  $X_i$  axis to the origin of the  $i$ th frame along the  $X_i$  axis, and  $\alpha_i$  is the offset angle from the  $Z_{i-1}$  axis to the  $Z_i$  axis about the  $X_i$  axis [19].

The inverse kinematics problem is usually more complex for redundant robots. Traditionally, three models are used to solve



**Table 1.** D-H Parameters of the Robot Arm

Joint i	$\alpha_{i-1}$	$\alpha_i$	$d_i$	$\theta_i$
1	90	0	9	$\theta_1$
2	0	8	0	$\theta_2$
3	0	8	0	$\theta_3$
4	-90	17	0	$\theta_4$
5	0	0	0	$\theta_5$

the inverse kinematics problem, these being the geometric, algebraic and iterative models [4]. In this work the geometric model is used to compute the inverse kinematic problem. A detailed explanation about the inverse kinematics of the robot arm is omitted and only the computational procedure is given as follows. The coordinates of the object x and y values are the input of the following equations and joint angles of the robot arm are the output of the equations. Also,  $\alpha_2$ ,  $\alpha_3$ ,  $\alpha_4$  and  $d_1$  are in D-H parameters and z is equal to zero because the object is always at the bottom.  $P$  is the angle of the robot arm approach to the object having been previously determined according to the distance between the robot arm and the object ( $X'$ ). Inverse kinematic equations are simplified by values and optimized for FPGA implementation. In Table 2,  $P$  are given as shown:

$$\theta_4 = P - \theta_3 \quad (17)$$

Inverse kinematic solutions contain mainly trigonometric functions (cos, sin, arctan etc.) which have to be computed

**Table 2.** Approach angles table for  $X'$  values

$X'$	11-16 cm	17-19 cm	20-22 cm	23-28 cm	29-32 cm
$P$	-80°	-70°	-60°	-40°	-30°

$$\theta_1 = \arctan \frac{y}{x} \quad (8)$$

$$X' = \sqrt{x^2 + y^2} \quad (9)$$

$$X_b = \frac{X' - a_4 \cos(P)}{2a_2} \quad (10)$$

$$Z_b = \frac{Z - d_1 - a_4 \cos(P)}{2a_3} \quad (11)$$

$$Q = \sqrt{\frac{1}{(X_b^2 + Z_b^2) - 1}} \quad (12)$$

$$P_1 = \arctan \frac{X_b - QZ_b}{Z_b + QX_b} \quad (13)$$

$$P_2 = \arctan \frac{X_b + QZ_b}{Z_b - QX_b} \quad (14)$$

$$\theta_2 = 90 - P_1 \quad (15)$$

$$\theta_3 = \theta_2 - P_2 \quad (16)$$

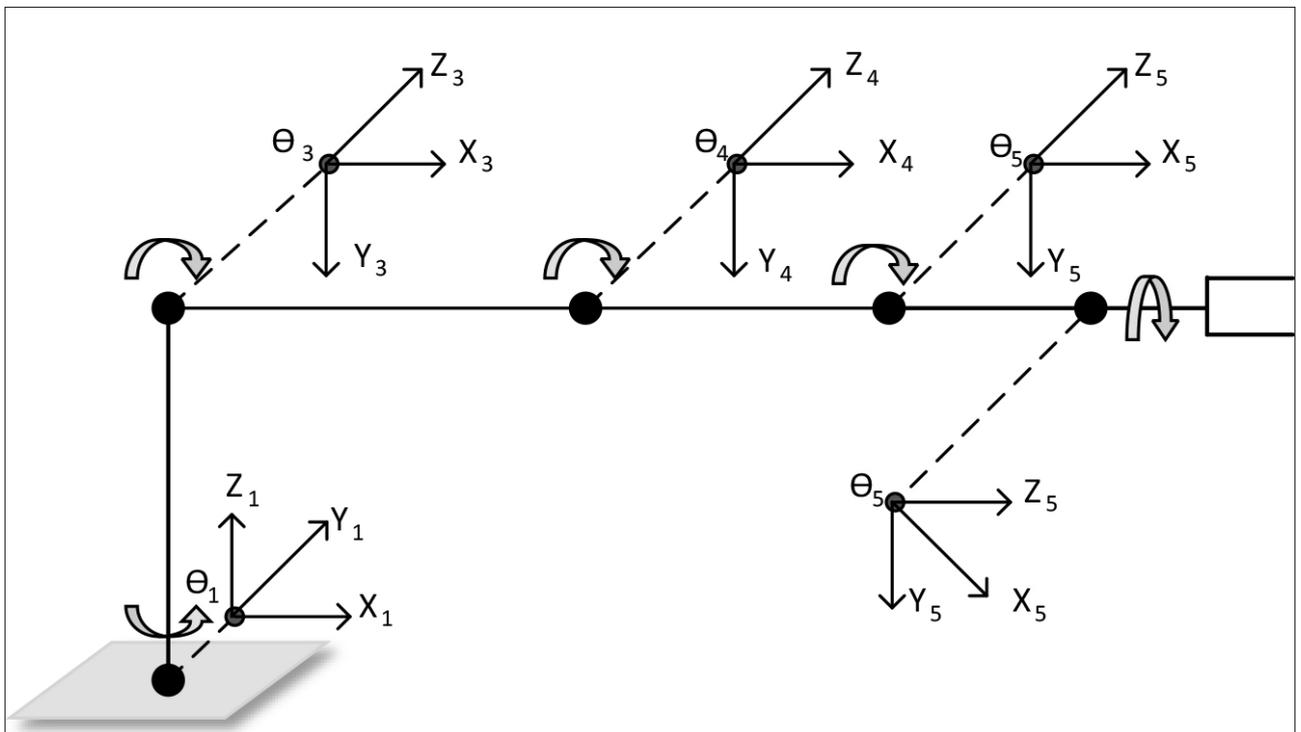
$$\theta_4 = P - \theta_3 \quad (17)$$

**Table 3.** Analysis of the inverse kinematic solution

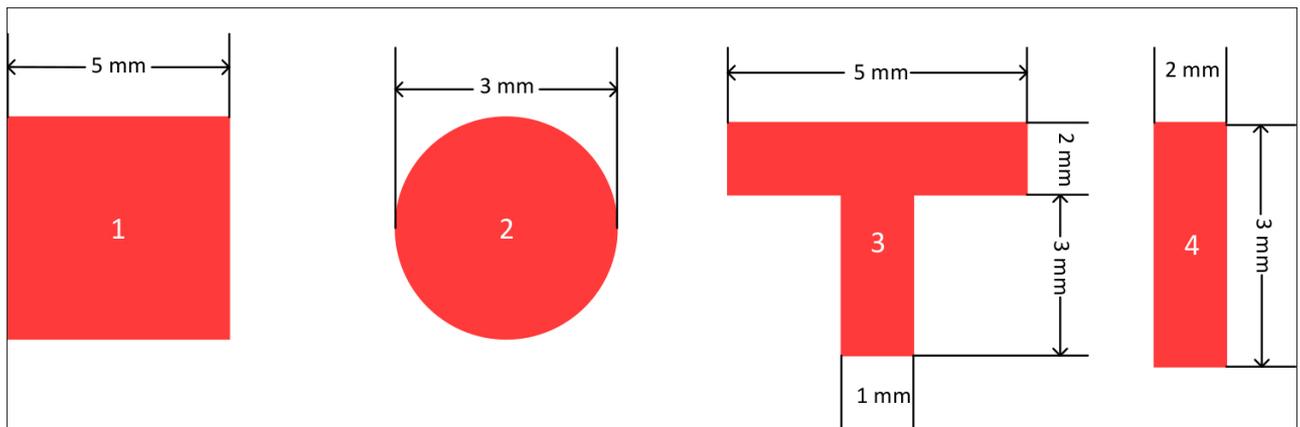
Input to In-verse Kine-matic	Angles of the Robot Arm	Calculated	Outputs of the Proposed Inverse Kinematic Calculation	Error
x=29 y=10	$\theta_1$	19,02560603756870	19,02560603777450	-2,05802E-10
	$\theta_2$	-87,80028882245060	-87,80028799234560	-8,30105E-07
	$\theta_3$	82,01028215959120	82,01028287942240	-7,19831E-07
	$\theta_4$	-24,20999333714060	-24,20999924561230	5,90847E-06
x=12 y=-5	$\theta_1$	-22,61986494804040	-22,61986494985630	-1,81587E-09
	$\theta_2$	-14,83273900666030	-14,83273993654120	9,29881E-07
	$\theta_3$	14,89228119041620	14,89228033264130	8,57775E-07
	$\theta_4$	-80,05954218375590	-80,05959565613210	5,34724E-05

**Table 4.** FPGA resource usage

	Combinational functions	Dedicated logic registers	Memory bits	Embedded multiplier 9-bit elements	PLLs
Image processing	2708	1603	42448	18	1
PWM generator	426	209	2576	0	0
Inverse kinematic calculation	54839	22151	30300	343	0
Total	57993	23963	75324	361	1



**Figure 5.** Link coordinates of the robot arm



**Figure 6.** Objects used in tests

separately for each joint angle. The CORDIC algorithm is used to compute these complicated functions because it basically allows the computation of transcendental functions using shift and add operations only and can be implemented very efficiently in simple hardware [20]. Also the implemented CORDIC architectures are based on the IEEE-754 standard and work with 64 bit double precision.

### Parallel Pulse Width Modulation Generator

The robot arm has 6 servomotors which actuate its joints. Each of the servomotors is controlled with single pulse width modulation (PWM) signal. All the PWM outputs should be parallel to enable smooth movement of the robot arm. The parallel controller is described using Verilog HDL with modules that are designed for each of the servomotors' output. The servomotors are controlled 50 Hz PWM signals which have a duty cycle of between 0.5 ms and 2.5 ms. A 32 bit counter is programmed to get precision motor motions. This variable's value changed between (666666)16 and (1FFFFFFE) 16 and the servomotor shaft varies between 0 and 180 degrees. A step of servomotors is 4, 19E-7 degrees.

### Results

The overall experimental system is depicted in Figure 1. The proposed FPGA implementation of robot arm control with image processing was coded in Verilog HDL. The system simulation was performed using ModelSim (Version 10.1d) and synthesized using Quartus II 13.1 software. The designed system implemented DE2-115 FPGA development board and was tested in varied light conditions with objects in different shapes and sizes. In Figure 6 the objects used in the test are shown. In the tests the objects were located randomly in the work place and the success of the proposed system was observed. Each object was tried out 500 times, and the first and second objects were grasped by the robot arm with a success rate of 100%. The success rates of the third and fourth objects were 95.6% and 96.8% respectively.

If two or more objects were located in the work place at the same time, the image processing unit determined this situation and sep-

arately computed centre coordinates of the objects in the work place and sent them to inverse kinematic calculation. Thus, the robot arm could take the objects from the work place one by one.

An application was developed to evaluate the precision of the proposed inverse kinematic solution. Inputs to inverse kinematic were manually entered and outputs from the inverse kinematic of the proposed application were observed. The application was loaded in the FPGA development board and the results were observed from seven segment displays on the board. In Table 3, analysis of the inverse kinematic solution is given.

The FPGA used in the proposed implementation had 114 480 logic elements. Only 3% and 1% of the logic elements were used by image processing and parallel PWM generator respectively. All implementation was realized with 57 993 (%51 of the FPGA) logic elements. FPGA resource usage is explained in table 4.

### Conclusion

In this study, a hardware implementation of robot arm control with real time image processing is presented. The proposed implementation combines high precision inverse kinematic calculation, colour based object recognition and robot arm control. The aim of this paper is to advance the precision of space and industrial robot arms, where reliability is a substantial requirement. To improve precision of the inverse kinematic calculation, CORDIC algorithm based on double precision IEEE 754 floating point number format is used. To recognize the object in different illumination, RGB colour space is converted to HSV colour space. The parallel PWM generator is also designed for the smooth operation of the robot arm. All applications are combined and implemented in FPGA development board. It was confirmed that the inverse kinematic calculation is very precise, and the tests carried out show that the proposed system works well.

**Peer-review:** Externally peer-reviewed.

**Conflict of Interest:** The authors have no conflicts of interest to declare.

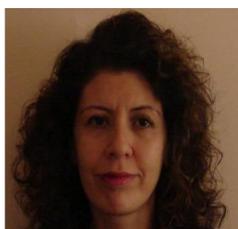
**Financial Disclosure:** The authors declared that this study has received no financial support.

## References

1. W. He, K. Yuan, H. Xiao, Z. Xu, "A high speed robot vision system with GigE vision extension", *IEEE International Conference on Mechatronics and Automation*, 2011, pp. 452-457.
2. X. Zhang, M.H. Lee, "A Developmental Robot Vision System", *IEEE International Conference on Systems, Man and Cybernetics*, 2006, pp. 2024-2029.
3. P.R. Possa, S.A. Mahmoudi, N. Harb, C. Valderrama, P. Manneback, "A Multi-Resolution FPGA-Based Architecture for Real-Time Edge and Corner Detection", *IEEE Transactions on Computers*, 2014, pp. 2376-2388.
4. R. Köker, "A genetic algorithm approach to a neural-network-based inverse kinematics solution of robotic manipulators based on error minimization", *Information Sciences*, 222, pp. 528-543.
5. Y.H. Yu, N.M. Kwok, Q.P. Ha, "Color tracking for multiple robot control using a system-on-programmable-chip," *Automation in Construction*, 20(6), pp. 669-676, 2013.
6. X. Lu, D. Ren, S. Yu, "FPGA-based real-time object tracking for mobile robot," *International Conference on Audio, Language and Image Processing*, 2010, pp. 1657-1662.
7. M. Z. Zhang, M.J. Seow, L. Tao, V.K. Asari, "A tunable high-performance architecture for enhancement of stream video captured under non-uniform lighting conditions", *Microprocessors and Microsystems*, 32(7), pp. 386-393, 2008.
8. J. Rodríguez-Araújo, J.J. Rodríguez-Andina, J. Fariña, M.Y. Chow, "Field-Programmable System-on-Chip for Localization of UGVs in an Indoor iSpace", *IEEE Transactions on Industrial Informatics*, 10(2), pp. 1033-1043, 2014.
9. X. He, Z. Wang, H. Fang, K. He, R. Du, "An embedded robot controller based on ARM and FPGA", *4<sup>th</sup> IEEE International Conference on Information Science and Technology*, 2014, pp. 702-705.
10. Y.S. Kung, H. Cheng-Ting, C. Hsin-Hung, T. Tai-Wei, "FPGA-realization of a motion control IC for wafer-handling robot", *8<sup>th</sup> IEEE International Conference on Industrial Informatics*, 2010, pp. 493-498.
11. S. Seok, D.J. Hyun, S. Park, D. Otten, S. Kim, "A highly parallelized control system platform architecture using multicore CPU and FPGA for multi-DoF robots", *IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 5414-5419.
12. Y. Zheng, H. Sun, Q. Jia, G. Shi, "Kinematics control for a 6-DOF space manipulator based on ARM processor and FPGA Co-processor", *6<sup>th</sup> IEEE International Conference on Industrial Informatics*, 2008, pp. 129-134.
13. M. K. Wu, Y. S. Kung, Y. H. Huang, T. H. Jung, "Fixed-point computation of robot kinematics in FPG", *International Conference on Advanced Robotics and Intelligent Systems (ARIS)*, 2014, pp. 35-40.
14. Y. S. Juang, T. Y. Sung, L. T. Ko, C. I. Li, "FPGA implementation of a CORDIC-based joint angle processor for a climbing robot", *International Journal of Advanced Robotic Systems*, 10(4), p. 195, 2013.
15. R. Szabo, A. Gontean, A., "Robotic Arm Control Algorithm Based on Stereo Vision Using RoboRealm Vision", *Advances in Electrical and Computer Engineering*, 15(2), pp. 65-74, 2015.
16. M. Kazmi, A. Aziz, P. Akhtar, D. Kundi, "FPGA based compact and efficient full image buffering for neighborhood operations", *Adv. Electr. Comput. Eng*, 15(1), pp. 95-104, 2015.
17. A. Alabdo, J. Pérez, G.J. Garcia, J. Pomares, F. Torres, "FPGA-based architecture for direct visual control robotic systems", *Mechatronics*, 39, pp. 204-216, 2016.
18. R. P. Paul, "Robot manipulators: mathematics, programming, and control: the computer control of robot manipulators", Cambridge, London, MIT Press, 1983.
19. J. Rigelsford, "Robotics: Control, Sensing, Vision and Intelligence", *Industrial Robot: An International Journal*, vol. 26, no. 2, 1999.
20. C. Krieger, B. J. Hosticka, "Inverse kinematics computations with modified CORDIC iterations," *IEE Proceedings - Computers and Digital Techniques*, vol. 143, no. 1, pp. 87-92, 1996.



Barış Çelik was born in Bursa, Turkey. He received the M.Sc. degree in electronic and communication education from Marmara University, Istanbul, Turkey. He is currently pursuing the Ph.D. degree in electric and electronic engineering with the University of Marmara, Istanbul, Turkey.



Dr. Ayça Ak received her B.Sc. and M.Sc. degree from the Firat University, Electrical Electronics Engineering Department, in 1993 and 1997 respectively, Ph.D degrees from the Yildiz Technical University, Control and Automation Department of Electrical Engineering in 2008. She is currently working as faculty member, phd. in Electronics and Automation Department in Vocational School of Technical Sciences, Marmara University, Istanbul-Turkey. Her research expertise is Robotic, Control, Neural Network, Fuzzy Logic, Web-Based Education and Applications. She has many international and national books, articles and project works.



Vedat Topuz received his B.Sc. and M.Sc. degrees from Marmara University, Electronics and Computer Education Department 1984 and 1995 respectively, Ph.D degrees from Marmara University Computer Systems Education Department in 2003. He is currently working as Assoc. Prof.Dr. in Computer Department in Vocational School of Technical Sciences, Marmara University, Istanbul-Turkey. Her research expertise is Artificial Intelligence, Robotics, Control, Fuzzy Logic, Web-Based Education and Applications. He has many international and national books, articles and project works.