Genetic Algorithm Based Pick and Place Sequence Optimization for a Color and Size Sorting Delta Robot

[1]H.A.G.C. Premachandra, [2]H.M.A.N. Herath, [3]M.P. Suriyage, [4]K.M. Thathsarana, [5]Y.W.R. Amarasinghe, [6]R.A.R.C. Gopura, [7]S.A. Nanayakkara
Department of Mechanical Engineering
University of Moratuwa
Katubedda, Sri Lanka 10400
e-mail: [1]charithprem@gmail.com

***Abstract*—Delta Robots are used in industry for light weight material handling and sorting. This paper presents a sequence optimizing methodology for a color and size sorting delta robot. It finds the optimum path in the task space to perform an industry emulated scenario. An OpenCV-Python program was developed to sort objects according to their colors and sizes. The static positional coordinates of the objects in the robot workspace are obtained using the program. Genetic algorithm is used for pick-and-place sequence optimization to ensure that the sorting process is performed in the shortest possible path. The static positional coordinates are used to calculate the fitness. Single point crossover and mutation are applied with elitist selection when the current generation evolves to the next generation. The genetic algorithm ensures that the sequence of pick-and-place converges to the highest fitness in minimum number of generations reducing the computational time.**

Keywords-delta robot; image processing; path planning; genetic algorithm; sequence optimization

I. INTRODUCTION

Parallel manipulators are widely used in industrial applications compared to serial manipulators [1]. Serial manipulators are susceptible to bending at high loads and have high vibration resulting in lack of precision. Delta robots are parallel type of robots which are capable of performing high-speed movements accurately in a limited workspace. This makes them highly suitable for handling light objects [2]. They are used in packaging and drilling applications mainly aimed at food, pharmaceutical and electronics industry [3]. Vision based sorting systems are widely used in food industry for quality grading and in packaging industry for classifying similar products with different colors and shapes [4].

Sorting is a labor-intensive process. Automated sorting systems reduce the time to accomplish a certain task with minimum human intervention and contamination to the products. Vision guided parallel robots are currently being used in the industry [5] to increase the productivity. There have been several studies [6], [7] using open source computer vision (Open CV) to perform industrial sorting operations.

Time-optimal trajectory planning has been achieved for the pick-and-place operation path of a delta robot [8], [9]. Lee et al [10] used Genetic Algorithm (GA) based trajectory planning considering the most significant optimality criteria for the fitness function. Only a few researches have been conducted in the industrial context for sequence optimization problems. PCB assembly lines in electronic industry uses GA for optimization of the assembly process [11], [12]. The Travelling Salesman Problem (TSP) approach has been followed in most of the cases [13] including PCB assembling mentioned above. However, the existing work are either task specific or omitting the requirement for sequence optimization to enhance productivity of the robot. Therefore, a sequence optimization algorithm is proposed for a more generalized emulated scenario to overcome above issues.

This paper introduces a methodology with image processing and sequence optimization to perform a pick-and-place sorting task for an emulated industrial scenario. Section II describes the sorting process and the specifications of the hardware system. Section III describes the image processing algorithm. Section IV and V discuss the reason for implementing sequence optimization while explaining the GA in detail. In Section VI, experimental results are presented. Section VII concludes the paper.

II. HARDWARE SYSTEM

A delta robot is used to pick-and-place stationary objects into two conveyors as shown in Fig. 1. The objects are on a table inside the workspace of the robot. Those objects are to be sorted and placed into two conveyors (i.e. two categories).

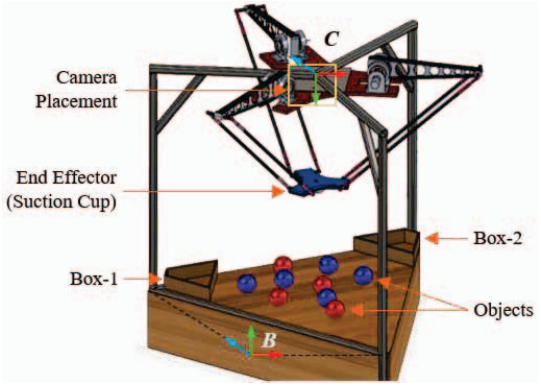

Figure 1. The emulated scenario with camera coordinate frame ***C*** and base coordinate frame ***B*** of the robot. In this figure, two boxes are replacing the conveyors at either side of the delta robot.

978-1-7281-6139-6/20/$31.00 ©2020 IEEE

TABLE I. SPECIFICATIONS OF THE DELTA ROBOT

Description	Parameter	Value
Geometrical parameters	Base radius	170 mm
	Bicep length	270 mm
	Forearm length	500 mm
	End effector radius	55 mm
Workspace (cylindrical)	Base to floor distance	660 mm
	Radius	315 mm
	Height	330 mm
Payload		200 g

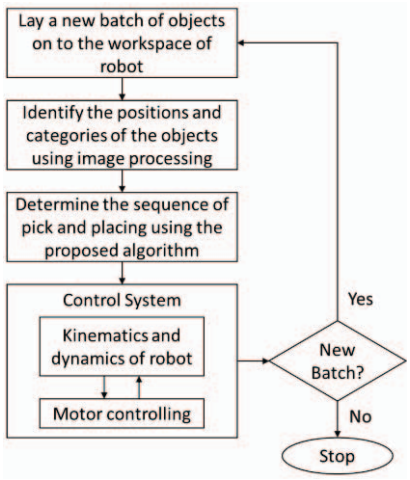

Figure 2. Flow chart of the sorting process.

Figure 2 shows the sorting process of the delta robot after implementing the proposed methodology. Python 2.7.16 with OpenCV 3.4.2 library was used for programming the proposed system on a laptop (with 2.50 GHz intel® core™ i7 6th generation processor), installed with Windows® 10. The methodology consists of two parts: object detection and planning the pick-and-place sequence. A camera is mounted under the robot's base (Fig. 1). An A4Tech full HD camera was used to carry on object detection which gives 640 × 480 resolution images of its field of view.

III. IMAGE PROCESSING FOR COLOR AND SIZE SORTING

An image processing algorithm was developed using python-OpenCV to find the positional coordinates of objects based on color and sizes. This algorithm achieves more flexibility allowing the user to choose any sample color through the camera. First, the program shows the histograms (both RGB and HSV) of the user-input colors so that the user can specify the lower and upper bounds for the HSV values. Then the image processing algorithm provides the coordinates of the duo-chrome objects. A calibration pane was used for the purpose of calibrating the HSV values and the threshold size of the detected objects during the program execution.

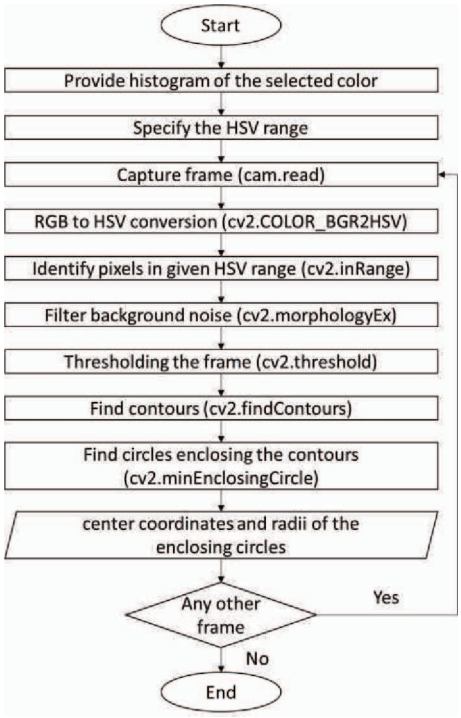

Figure 3. Flow chart of the image processing algorithm for object detection using color recognition with python-OpenCV.

Algorithm in Fig. 3 is for monochrome object detection. The emulated scenario of duo-chrome objects detects two colors and provide two sets of positional coordinates. It was assumed that the centroid of the object coincides with the center of its minimum enclosing circle (obtained using the cv2.minEnclosingCircle method). The radius of the minimum enclosing circle allows to further categorize the objects based on their sizes. A threshold value can be defined by the user such that objects exceeding the threshold is categorized as large and vice versa.

Since object detection is done for stationary objects, a single camera frame is enough per batch of objects. In addition, looping through camera frames enables to get positional coordinates of moving objects by using the same algorithm. This leads to another research of dynamic object tracking which is not discussed in this paper.

IV. SEQUENCE OPTIMIZATION

After obtaining the positional coordinates of n number of objects on the table, the end effector must travel to each object coordinate to pick and place it on the relevant conveyor. The conveyors have fixed known coordinates. It can be noticed that the total number of possible sequences is given by the number of permutations of all the objects.

$$P(n,r) = \frac{n!}{(n-r)!} \quad (1)$$

where n is the total number of objects and r is the number of objects taken at a time. In this case, $n = r$ since all the objects are permutated at a time. Equation (1) then simplifies to,

$$P(n,n) = n! \quad (2)$$

The obvious method to find the shortest path is brute-force (i.e. going through all possible sequences and find the shortest path). This is possible for a small number of items.

TABLE II. TOTAL NUMBER OF POSSIBLE PATHS

Number of objects (n)	Total paths ($n!$)
4	24
5	120
6	720
7	5040
8	40320
9	362880
10	3628800

According to Table II, total possible paths for pick-and-place sequence have a factorial growth with the number of items thus increases the computational time. Implementing Genetic algorithm (GA) for sequence optimization overcomes this problem by finding the shortest path utilizing a lower computational cost.

V. GENETIC ALGORITHM IMPLEMENTATION

GA which belongs to evolutionary algorithms, is based upon the "suvival of the fittest" as in Darwinian evolutionary theory and natural selection. This section provides an overview of the steps followed in developing the GA for the sequence optimization problem in the emulated scenario.

Step1. Generate an initial population of possible random sequences (genotypes or sometimes called as chromosomes) using cartesian coordinates of the conveyors and items. The population size is determined according to the number of items (i.e. the population size is always less than the total number of paths for a given number of objects as in Table II to avoid duplication of individuals inside the population).

Step2. Select parents from the previous population using tournament selection method [13]. The fitness is calculated as,

$$F = -T \tag{3}$$

where F is fitness and T is travelling distances. Note that the calculation deviates from the TSP when calculating the travlling distance. Leu *et al* [12] model their PCB component placement problem as TSP since they are placing the same type of component. Hence, the distance from one component to the other remains constant. However, in this case, traveling distance depends on the color of the previous object since the end effector of robot has to travel to the respective conveyor before picking the next object.

Step3. Crossover is the process of generating the offspring from the parents after exchanging their genes. Even though several conventional crossover methods such as single-point crossover, two-point crossover and uniform crossover are available, single-point crossover was used in this algorithm.

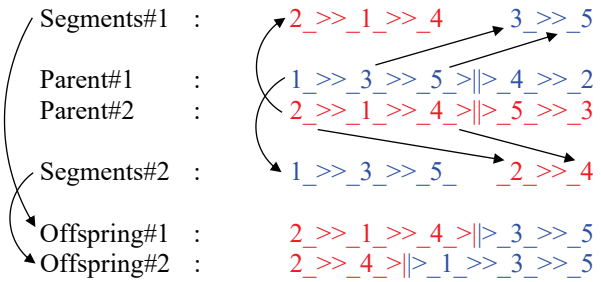

Figure 4. Crossover operator.

In Fig. 4, 1_>>_3_ depicts that picking no. 3 item after no. 1 item. >||> depicts the crossover point. Crossover rate determines whether two paths can be crossed over as shown above (i.e. crossover probability). Chromosomal segment of Parent#2 before the crossover point is directly transferred to the offspring#1. The missing genes are exchanged from Parent#1 as shown in Fig. 4 such that the order is preserved. High crossover rates result in omitting good solutions.

Step4. The mutation is done by swapping two consecutive pickings at a random place on the genotype.
e.g. 1 >> 2 >> 3 >> 4 after mutation: 1 >> 3 >> 2 >> 4

Mutation is not important when there is a small number of items since above mutated individual may have already been included in the random generated population (i.e. when randomly generating paths, both 1 >> 2 >> 3 >> 4 and 1 >> 3 >> 2 >> 4 would have already been included in the population). However, when the number of items increases, mutation plays a vital role to avoid local minima.

Step5. A group of individuals with best fitness is sent to the next generation without any change (i.e. elitist selection). Offspring is generated from the rest. The new generation is the best individuals from the previous generation and the new offspring.

Step6. Follow these steps for several iterations. The individual having the best fitness is the most optimized solution in the final population.

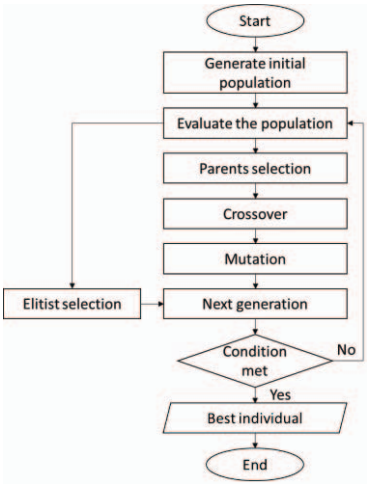

Figure 5. Flow chart of the proposed genetic algorithm.

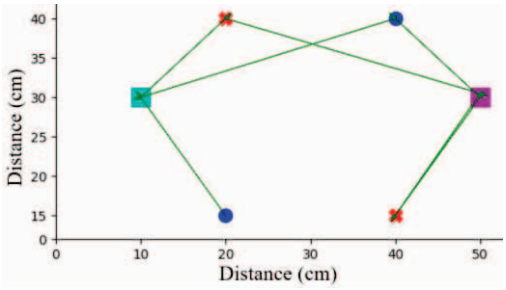

Figure 6. Best path using GA to pick four objects. Red crosses and blue dots are object positions while squares are depicting conveyors on the Cartesian plane. Green arrows show the pick-and-place sequence.

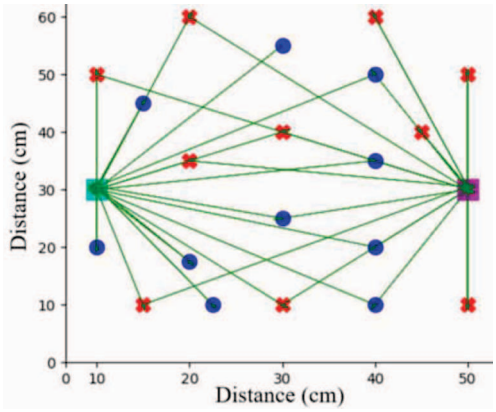

Figure 7. Best path using GA to pick multiple objects. Red crosses and blue dots are object positions and squares are depicting conveyors on the Cartesian plane. Green arrows show the pick-and-place sequence

VI. Experiments and Results

Two experiments were conducted to evaluate the image processing algorithm and the proposed genetic algorithm. Randomly placed duo-chromatic objects with different sizes were processed to identify the objects and to acquire their positional coordinates. Those objects were sorted into groups based on their color and size accordingly. The positional coordinates of the objects were fed to the proposed genetic algorithm for sequence optimization.

A. Image Processing Results

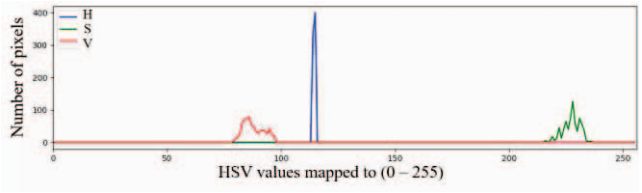

Figure 8. HSV histogram of detected blue.

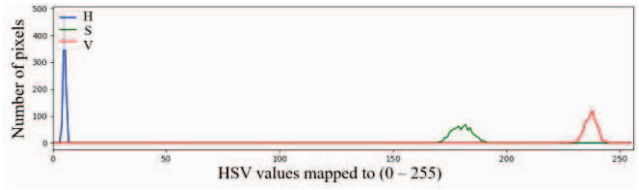

Figure 9. HSV histogram of detected red.

Before specifying the HSV range, the color histograms of the duo-chromatic objects were analyzed. The reason was that the HSV values may vary due to camera quality, lighting condition and the surface texture of object. Hence, the lower and upper bounds for the HSV values were chosen (Table III) such that the range is within the histogram distribution.

TABLE III. Selected HSV Values

Color		H	S	V
Blue	Lower Bound	100	200	70
	Upper Bound	130	250	120
Red	Lower Bound	0	170	190
	Upper Bound	20	220	240

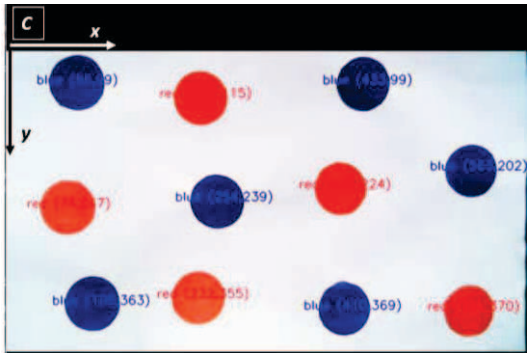

Figure 10. Final output from the camera with the centroidal positions on the xy plane with respect to ***C***.

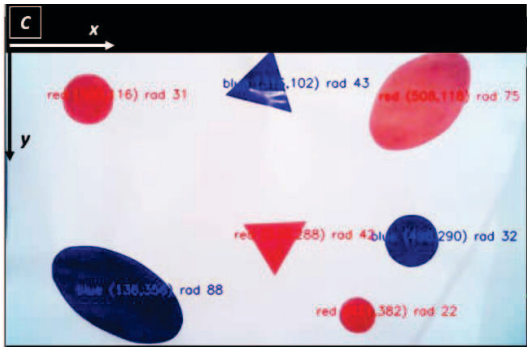

Figure 11. Final camera output with the centroidal positions and objects categorized based on both color and size on the xy plane with respect to ***C***. The size is measured by radius of the circumcircle bounding each object.

B. Genetic Algorithm Results

The highest fitness value of every generation is expected to converge when the population evolves. The individual in the final generation with highest fitness is the optimum sequence according to (3). Considering the four duo-chrome items in the Fig. 6, the fitness has converged to (-145.63) which is same as the shortest distance for the end effector to travel. The solution converged as early as in the third generation (refer Fig. 12). This is a common phenomenon for a smaller number of items.

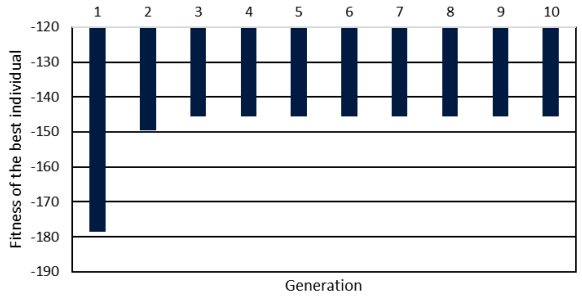

Figure 12. Convergence of maximum fitness for four items in Fig. 6.

However, the convergence of fitness is more critical with larger number of items as in Fig. 7. Although the solution may not be the shortest path, it is the most optimized sequence with minimum computations. According to the convergence graph in Fig. 13, it took around 400 generations to increase its fitness to (-916.434) which is the optimized traveling distance for the end effector.

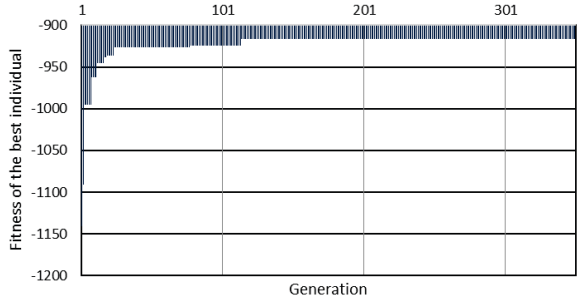

Figure 13. Convergence of maximum fitness for multiple items in Fig. 7.

The test case in Fig. 6 has a computational time of 0.4*s* and the test case in Fig. 7 takes around 1.5*s*. The converging time has increased with number of items. However, this algorithm demonstrates enough capability in the sorting application. Table IV depicts the genetic parameters which were used to produce results in Fig. 13.

TABLE IV. GENETIC PARAMETER COMBINATION

Parameter	Value	Remarks
Population size	30	Can increase depending on number of items.
Selection method	-	Tournament selection; efficient and simple.
Probability of crossover	80%	Requires additional research to determine a suitable value. Depends on the distribution of objects in the workspace.
Probability of mutation	5%	Requires additional research.
Elitist selection	8	Number of best individuals passed to the next generation unaltered. Requires additional research.
Number of generations	1000	Stopping condition.

It is required to determine suitable parameters used in the GA in order to get an early convergence thus reducing the computational time. It is noticed that computational time is proportional to the number of objects and their distribution in the workspace of the robot.

VII. CONCLUSION

A color and size sorting algorithm was proposed using OpenCV-Python for an industry emulated scenario. The algorithm can be used for real time dynamic object tracking. The color accuracy of the camera affects the overall performance of image processing while evenness of lighting condition in the field of view of camera gives better results. Distortion effects have not been considered. A proper calibration procedure must be followed to decide the scale factors, translation vectors and rotation angles when defining the transformation matrix from camera coordinate frame to the base coordinate frame of the robot.

The developed sequence optimization algorithm can be easily adapted to pick-and-place sequence optimizing problems in the industry irrespective of a specific application.

REFERENCES

[1] Y. D. Patel and P. M. George, "Parallel Manipulators Applications - A Survey," *Mod. Mech. Eng.*, vol. 02, no. 03, pp. 57–64, 2012.

[2] A. Kosinska, M. Galicki, and K. Kedzior, "Designing and Optimization of Parameters of Delta-4 Parallel Manipulator for a given Workspace," *J. Robot. Syst.*, vol. 20, pp. 539–548, Sep. 2003.

[3] Z. Affi, L. Romdhane, and A. Maalej, "Dimensional synthesis of a 3-translational-DOF in-parallel manipulator for a desired workspace," *Eur. J. Mech. - ASolids*, vol. 23, no. 2, pp. 311–324, Mar. 2004.

[4] Z. Liu, B. Zhao, and H. Zhu, "Research of Sorting Technology Based on Industrial Robot of Machine Vision," *International Symposium on Computational Intelligence and Design*, vol. 1, pp. 57–61, 2012.

[5] V. Bulej, J. Stanček, and I. Kuric, "Vision guided parallel robot and its application for automated assembly task," *Adv. Sci. Technol. Res. J.*, vol. Vol. 12, no 2, 2018.

[6] Y. Jia, G. Yang, and J. Saniie, "Real-time color-based sorting robotic arm system," *IEEE International Conference on Electro Information Technology (EIT)*, pp. 354–358, 2017.

[7] J. B., S. V., V. Purohit, D. Oswald Tauro, and V. J., "Design and Development of Automated Intelligent Robot Using OpenCV," *International Conference on Design Innovations for 3Cs Compute Communicate Control*, pp. 92–96, 2018.

[8] T. Su, L. Cheng, Y. Wang, X. Liang, J. Zheng, and H. Zhang, "Time-Optimal Trajectory Planning for Delta Robot Based on Quintic Pythagorean-Hodograph Curves," *IEEE Access*, vol. 6, pp. 28530–28539, 2018.

[9] Y. Zhang, R. Huang, Y. Lou, and Z. Li, "Dynamics based time-optimal smooth motion planning for the delta robot," *IEEE International Conference on Robotics and Biomimetics*, pp. 1789–1794, 2012.

[10] Y. Dae Lee, B. Hee Lee, and H. Gyoo Kim, "An evolutionary approach for time optimal trajectory planning of a robotic manipulator," *Inf. Sci.*, vol. 113, no. 3, pp. 245–260, Feb. 1999.

[11] M. Ancău, "The optimization of printed circuit board manufacturing by improving the drilling process productivity," *Comput. Ind. Eng.*, vol. 55, no. 2, pp. 279–294, Sep. 2008.

[12] M. C. Leu, H. Wong, and Z. Ji, "Planning of Component Placement/Insertion Sequence and Feeder Setup in PCB Assembly Using Genetic Algorithm," *J. Electron. Packag.*, vol. 115, no. 4, pp. 424–432, Dec. 1993.

[13] H. M. Pandey, "Performance Evaluation of Selection Methods of Genetic Algorithm and Network Security Concerns," *Procedia Comput. Sci.*, vol. 78, pp. 13–18, Jan. 2016.