Efficient Integration of Template Matching, Calibration and Triangulation for Automating Peg Hole Insertion Task Using Two Cameras

Andres Sauceda Cienfuegos[#1], Baidya Nath Saha[*2], Jesus Romero-Hdz[#3], David Ortega[*4]

[#1]*Universidad Autónoma de Nuevo León, San Nicolás de los Garza, México*
[*2]*Centro de Investigación en Matemáticas (CIMAT), Monterrey, México*
[#3,*4]*Centro de Ingeniería y Desarrollo Industrial (CIDESI), Monterrey, México*
[#1]andres_sau.cien@hotmail.com
[*2]baidya.saha@cimat.mx
{[#3]jaromero,[*4]ortega.a}@posgrado.cidesi.edu.mx.

Abstract *— This paper integrates template matching, calibration and triangulation algorithms in an efficient way to automate peg-hole insertion task using a pair of cameras. First we implement a fast template matching (fast correlation based block matching) algorithm for automatically finding the peg and hole using two cameras. We exploit the templates of the peg and hole at different orientation and illumination to improve the accuracy of the template matching algorithm. Then we implement the Direct Linear Transform (DLT) method based calibration algorithm to find the intrinsic and extrinsic parameters of the camera. We then refine the camera calibration parameters through Levenberg-Marquardt (LM) based non-linear optimization method. We used two cameras to prevent the occlusion of peg and hole occurred due to robot movement and to reduce calibration error. Finally we implement a DLT based triangulation method to find the three dimensional world coordinates of the peg and hole from the images captured by two cameras. We use square and circular grids to reduce triangulation error. For triangulation method similar feature points of two images are matched through Harris corner detection for square and sift features for circular grids. Optimum camera parameters for triangulation method are determined based on minimum rectification based calibration error. We conducted the experiment on gantry robot. Experimental results demonstrate that efficient integration of template matching, calibration and triangulation method can successfully automate peg hole insertion task.*

Keywords *— Direct Lineal Transformation, Template Matching Algorithm, Harris corner detection, SIFT, Levenberg-Marquardt Algorithm, Triangulation method, peg-hole insertion task, gantry robot.*

I. *INTRODUCTION*

Peg-hole insertion task is a topic largely addressed and long standing problem in robotic research. Peg-hole assembly is the most basic and benchmark problem in which a peg is inserted into a hole [18]. The popularity of peg-hole insertion task is not only due to its importance in many industrial assembly tasks, but also for its complexity as a control problem that requires both position and force regulation. Human may achieve this task very easily, because we have the ability to perceive naturally all the factors that this process involves, nevertheless for a robot this task can be very complex. On the other hand, it will be a great benefit if robots can learn the human skill and apply it autonomously. Because automation of peg-hole insertion tasks increases productivity, costs reductions, and manual repetitive task reduction. Computer vision can automate this peg-hole insertion task.

This paper presents the automation of peg hole insertion task using computer vision technique. Towards achieving this goal, we mounted two cameras: one at the ceiling looking downwards and the other capturing the side view to avoid the occlusion of the peg and hole due to robot movement as shown in Fig. 1. We first implement the fast correlation based block matching algorithm for automatically finding the position of the peg and hole in the images captured by two cameras. We exploit the templates of the peg and hole at different orientation and illumination to automatically find the peg and hole as shown in Fig. 2. Then we implement the Direct Linear transform (DLT) based calibration method to find the intrinsic and extrinsic parameters of the camera. We then refine the camera calibration parameters through Levenberg-Marquardt (LM) based non-linear optimization method. Finally we implement triangulation method using Direct Linear Transform (DLT) method to find the world coordinate of the peg and hole from their coordinates in the images captured by two cameras. We find the optimum camera parameters for triangulation method using pairwise rectification error based calibration error. To compute rectification error we used square and circular grids. Feature matching for square grids are performed through Harris Corner and for circular grids are performed through sift features. We conducted the peg-hole insertion task using Gantry robot. Gantry robot is widely used in different industry operations that possess several advantages such as it can capture large work areas, it has the ability to reach at different angles and it achieves better positioning accuracy [17]. The target application of the presented system is to autonomously execute assembly operations of parts with complex geometry. To this end, we propose a computer vision system in a

combination of image processing tasks such as template matching, calibration and triangulation able to fulfill the steps required for such a high-level task, namely, visual object identification, fine robot positioning, picking and insertion strategies.

The outline of the remaining of the paper is as follows. Section II discusses brief literature review. Section III presents the fast template matching algorithm. Section IV explains calibration algorithm. Section V mentions triangulation methods. Section VI determines optimum camera parameters for triangulation. Section VII demonstrates experimental results and discussions. Section VIII concludes this research.

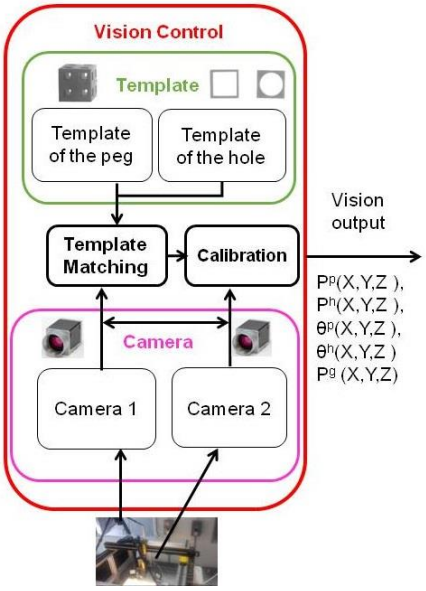

Fig. 1 Computer vision tasks for peg hole insertion using two cameras.

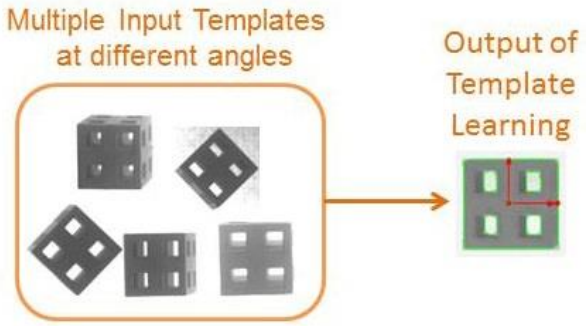

Fig. 2 Multiple templates at different angles and illumination. Output includes the object having the highest matching score with the available templates of different angles and illumination.

II. *BRIEF LITERATURE REVIEW*

Different computer vision based automated peg-hole insertion tasks are available in literature. A few of them are discussed below.

Choi et al. [3] automated an assembly task by using different sensor data. For gross motion control, vision and proximity sensors were used, and for fine motion control the force/torque sensor was used.

Kim and Cho [4] used vision sensors to solve for peg misalignments due to deformation. An algorithm was developed and tested, and proved to be effective in detecting misalignments and fixing the errors.

Xue *et al*. [5] introduced and used a new type cell of Self Organizing Manipulator for dual-peg-in-hole insertion. A camera and a force/torque sensor is used to complete this process, using the results obtained from an analysis of geometric and force conditions in performing that work.

To deal with an assembly task, Newman et al. [6] implemented intelligent methods which are faster and better than blind search. Sensor data is gathered and interpreted; furthermore, pattern matching can be used for some cases to improve greatly the time required for completion.

In this research, we integrated fast template matching, accurate calibration and triangulation method to automate the "peg-hole" insertion task. In the subsequent sections, different well-known methods of template matching, calibration and triangulation algorithm are discussed.

III. *FAST TEMPLATE MATCHING ALGORITHM*

Template matching algorithm are used to match the peg-hole pair which provides the image coordinates of the center position and rotation of the peg, hole and the position of the gripper. We exploited different templates at different orientations and angles to improve the performance of the template algorithm as shown in Fig. 2. In this section we discussed two different fast template matching algorithms, fast correlation based block matching [1] and zero mean normalized cross correlation [2]. These two methods made some modifications on normalized cross correlation method to make it faster. First we define the normalized cross correlation and then describe the two modified version.

A. *Normalized Cross Correlation [12]*

For template matching, cross correlation is a method related to the difference of the distance between images.

$$d^2_{f,t}(u,v) = \sum_{x,y}[f(x,y) - t(x-u, y-v)]^2$$

Expanding d^2

$$d^2_{f,t}(u,v) = \sum_{x,y}[f^2(x,y) - 2f(x,y)t(x-u)(y-v)^2 + t^2(x-u, y-v)]$$

The term $\sum_{x,y} t^2(x-u, y-v)$ is constant. If the term $\sum_{x,y}[f^2(x,y)$ is approximately constant, then

$$c(u,v) = \sum_{x,y} f(x,y)t(x-u)(y-v)$$

is a measure of the similarity between the image and the feature. Normalized cross correlation overcomes problems

with cross correlation like energy variation by normalizing the image and template vectors to unit length.

$$\gamma(u,v)=\frac{\sum_{x,y}[f(x,y)-\bar{f}_{u,v}][t(x-u,y-v)-\bar{t}]}{\{\sum_{x,y}[f(x,y)-\bar{f}_{u,v}]^2[t(x-u,y-v)-\bar{t}]^2\}^{0.5}}$$

Where $\bar{t}$ is the mean of the template and $\bar{f}$ is the mean of $f(x,y)$ in the region under the template.

B. Fast Correlation Based Block Matching [1]

Mahmood *et al.* [1] proposed two early termination criteria for normalized cross correlation to accelerate its computation. The first criterion is on growth based, starting with a perfect value and then decreasing, and when this partial value is lower than the yet known maxima, the remaining calculations are skipped. The second criterion used bounds to limit the minimum value that a region should have, skipping the calculations when this limit is not met.

Correlation coefficient between two blocks is commonly interpreted as the covariance of the two blocks normalized by the individual image standard deviations:

$$\rho_{bb'}=\frac{\sigma^2_{bb'}}{\sigma_b\sigma_{b'}}$$

A monotonic form of $\rho_{bb'}$ is proposed for early termination algorithms, while proceeding from the traditionally used form of ρ:

$$\rho_{bb'}=\sum_{i=1}^{n}\sum_{j=1}^{n}\tilde{b}(k,i_o+i,j_o+j)\cdot\tilde{b}(k',i'_o+i,j'_o+j)$$

Where:

$$\tilde{b}(k,i_o+i,j_o+j)=\frac{b(k,i_o+i,j_o+j)-\mu_b}{\sigma_b},$$

μ_b is the mean of block b. Since $\tilde{b}(k,i_o,j_o)$ has zero mean and unit variance therefore

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\{\tilde{b}^2(k,i_o+i,j_o+j)\cdot\tilde{b}^2(k',i'_o+i,j'_o+j)\}=2.$$

From these equations , we have:

$$\rho_{bb'}=1-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\Delta^2_{bb'}(i,j)$$

Where $\Delta_{bb'}(i,j)$ is given by:

$$\Delta_{bb'}(i,j)=\{\tilde{b}(k,i_o+i,j_o+j)-\tilde{b}(k',i'_o+i,j'_o+j)\}.$$

As the summation process proceeds, $\rho_{bb'}$ either decreases or remains same.

The monotonic decreasing growth pattern of ρ, provides the opportunity of early termination. For a specific search location, the partial value of ρ calculated over $(p\le n,q\le n)$ pixels provides an upper bound on the final value of $\rho_{bb'}$ to be obtained at that location:

$$\rho_{bb'}(p,q)\ge\rho_{bb'}\forall p\le n,q\le n.$$

As a result, if at a specific search location, the current value of correlation coefficient $\rho_{bb'}(p,q)$ once falls below the yet known maxima $\rho_{\max}$, it cannot improve after processing the remaining pixels. Therefore, further calculations at that search location becomes redundant and can be skipped. The percentage of pixels skipped at each search location depends upon the dissimilarity between the blocks b,b' and the magnitude $\rho_{\max}$.

An upper bound of ρ is derived as a function of the contents of the two blocks to be matched. Then that upper bound is used to filter out the search locations, exhibiting large dissimilarities with the block b. This upper bound is given by:

$$\rho^u_{bb'}=1-\frac{1}{2n^2}(\sum_{i,j}|\tilde{b}(i,j)|-\sum_{i,j}|\tilde{b}(i,j)|)^2.$$

The bound $\rho^u_{bb'}$ can be used for early termination of $\rho_{bb'}$ calculation as follows: at a specific search location $\rho^u_{bb'}$ is calculated before starting the correlation process and compared with the yet known best maxima, $\rho_{\max}$. If $\rho^u_{bb'}$ is found to be lower than the $\rho_{\max}$, correlation calculation at that specific search location becomes redundant and can be skipped.

C. Zero mean Normalized Cross Correlation (ZNCC) [2]

Stefano *et al.* [2] proposed an algorithm to skip calculations in the process of Normalized Cross Correlation. Here, two boundary conditions are checked at each image position. These two conditions check whether the current correlation can improve the best correlation found so far, if it cannot, the calculations are skipped.

Let I be the image under examination, of size $W\times H$ pixels, T the template sub-image, of size $M\times N$ pixels, and $Ic(x,y)$ the sub-image of size $M\times N$ located at pixel coordinates (x,y). Denoting as $\mu(T)$ and $\mu(Ic(x,y))$ the mean intensity value of T and $Ic(x,y)$. The Zero Mean Normalized Cross-Correlation (a more robust version of the NCC) between T and I at pixel position (x,y) can be written as

$$ZNCC(x,y)=\frac{\sum_{j=1}^{N}\sum_{i=1}^{M}[I(x+i,y+j)-\mu(Ic(x,y))]\cdot[T(i,j)-\mu(T)]}{\sqrt{\sum_{j=1}^{N}\sum_{i=1}^{M}[I(x+i,y+j)-\mu(Ic(x,y))]^2}\cdot\sqrt{\sum_{j=1}^{N}\sum_{i=1}^{M}[T(i,j)-\mu(T)]^2}}$$

Denoting with $\psi(x,y)$ the dot product between $Ic(x,y)$ and T and with $\|\cdot\|$ the ℓ_2 norm, simple algebraic manipulations allows the equation to be written as

$$ZNCC(x,y)=\frac{\psi(x,y)-M\cdot N\cdot\mu(Ic(x,y))\cdot\mu(T)}{\sqrt{\| Ic(x,y)\|^2-M\cdot N\cdot\mu^2(Ic(x,y))}\cdot\sqrt{\| T\|^2-M\cdot N\cdot\mu^2 T}}$$

Calling ψ(x, y) the numerator of and split T and Ic into two portions (rows from 1,…,n denoted with $|_1^n$ and rows n+1,…,N denoted with $|_{n+1}^N$) in order to express $\psi(x,y)$ as a sum of two contributions:

$$\psi(x,y)=\sum_{j=1}^{n}\sum_{i=1}^{M}[I(x+i,y+j)-\mu(Ic(x,y))]\cdot[T(i,j)-\mu(T)]+$$
$$\sum_{j=n+1}^{N}\sum_{i=n+1}^{M}[I(x+i,y+j)-\mu(Ic(x,y))]*[T(i,j)-\mu(T)]$$
$$=\psi_Z(x,y)|_1^n+\psi_Z(x,y)|_{n+1}^N$$

Starting from (3), two different bounding conditions of the term $\psi_Z(x,y)$ can be devised. These yield two sufficient conditions to be used during the matching process in order to skip rapidly those pixel positions that cannot improve the current maximum ZNCC score.

To get the first condition, Cauchy-Schwarz inequality is applied to the latter term of the equation, and we obtain an upper-bound of the term $\psi_Z(x,y)|_{n+1}^N$

$$\beta_Z'(x,y)|_{n+1}^N=\frac{\sqrt{(\| T\|_{n+1}^N)^2+(N-n)\cdot M\cdot\mu^2(T)-2\cdot\mu(T)\cdot\xi(T)|_{n+1}^N}}{\sqrt{(\| Ic(x,y)\|_{n+1}^N)^2+(N-n)\cdot M\cdot\mu^2(Ic(x,y))-2\cdot\mu(Ic(x,y))\cdot\xi(Ic(x,y))|_{n+1}^N}}$$

Calling $\eta_{Z\max}$, the maximum ZNCC score found so far, and by replacing $\psi_Z(x,y)|_{n+1}^N$ with $\beta_Z'(x,y)|_{n+1}^N$ in we obtain the following condition:

$$\frac{\psi_Z(x,y)|_1^n+\beta_Z'(x,y)|_{n+1}^N}{\sqrt{\| Ic(x,y)\|^2-M\cdot N\cdot\mu^2(Ic(x,y))}\cdot\sqrt{\| T\|^2-M\cdot N\cdot\mu^2 T}}\le\eta_Z\max$$

A second upper bounding function and associated sufficient condition can be obtained by algebraically manipulating the $\psi_Z(x,y)|_{n+1}^N$ term appearing in *(3)* before the application of the Cauchy-Schwarz inequality. So, we first rewrite $\psi_Z(x,y)|_{n+1}^N$ *as*

$$\psi_Z(x,y)|_{n+1}^N=\psi(x,y)|_{n+1}^N-\mu(T)\cdot\zeta(Ic(x,y))|_{n+1}^N-$$
$$\mu(Ic(x,y))\cdot\zeta(T)|_{n+1}^N+(N-n)\cdot M\cdot\mu(T)\cdot\mu(Ic(x,y))$$

Applying the Cauchy-Swarz inequality to the dot product term $\psi(x,y)|_{n+1}^N$ we obtain the upper bound $\beta_Z''(x,y)|_{n+1}^N$

$$\beta_Z''(x,y)|_{n+1}^N=\| Ic(x,y)\|_{n+1}^N\cdot\| T\|_{n+1}^N-\mu(T)\cdot$$
$$\zeta(Ic(x,y))|_{n+1}^N-\mu(Ic(x,y))\cdot\zeta(T)|_{n+1}^N+(N-n)\cdot$$
$$M\cdot\mu(T)\cdot\mu(Ic(x,y))$$

That yields the sufficient condition

$$\frac{\psi_Z(x,y)|_1^n+\beta_Z''(x,y)|_{n+1}^N}{\sqrt{\| Ic(x,y)\|^2-M*N*\mu^2(Ic(x,y))}*\sqrt{\| T\|^2-M*N*\mu^2 T}}\le\eta_Z\max$$

The algorithm to follow is [2]:

(1)Consider the next position $(x,y)\in 1$.

(2)Compute $\psi(x,y)|_{n+1}^N$, $\beta_Z'(x,y)|_{n+1}^N$ *and* $\beta_Z''(x,y)|_{n+1}^N$.

(3)If both boundary conditions are true go to step 1, else compute $\psi(x,y)|_{n+1}^N$.

(4)If $ZNCC(x,y)>\eta_{Z\max}$ update $\eta_{Z\max}$ together with the current best matching position $(x_{\max},y_{\max})$.

(5) Go to step 1

IV. *CALIBRATION ALGORITHM*

Calibration algorithm converts the image coordinates to the world coordinates which are used as an input to the robot kinematics algorithm. For calibration, we first implement Direct Linear Transform (DLT) method for three dimensional (3D) world coordinate (*x*, *y*, *z*). When we get the initial values of the camera parameters (both intrinsic and extrinsic), we refine these values using Levenberg – Marquardt (LM) based nonlinear optimization. These two techniques are given below.

A. Direct linear Transform (DLT) method for 3D [15]

It is based on the collinearity between a point expressed in world frame (x,y,z), its equivalent in image frame coordinates (u,v) and the central projection point of the camera.

Eleven coefficients $L_1,\ldots,L_{11}$ are needed to establish the relationship between the points in the world reference system and their equivalents in the image reference system, according to DLT method.

$$u=\frac{L_1x+L_2y+L_3z+L_4}{L_9x+L_{10}y+L_{11}z+1}$$
$$v=\frac{L_5x+L_6y+L_7z+L_8}{L_9x+L_{10}y+L_{11}z+1}$$

DLT calibration consists of calculating the eleven parameters, and since each point provides two equations, a minimum of six points to calibrate is necessary.

$$\begin{bmatrix} \frac{x}{R} & \frac{y}{R} & \frac{z}{R} & \frac{1}{R} & 0 & 0 & 0 & 0 & \frac{-ux}{R} & \frac{-uy}{R} & \frac{-uz}{R} \\ 0 & 0 & 0 & 0 & \frac{x}{R} & \frac{y}{R} & \frac{z}{R} & \frac{1}{R} & \frac{-vx}{R} & \frac{-vy}{R} & \frac{-vz}{R} \end{bmatrix} \begin{bmatrix} L_1 \\ L_2 \\ L_3 \\ L_4 \\ L_5 \\ L_6 \\ L_7 \\ L_8 \\ L_9 \\ L_{10} \\ L_{11} \end{bmatrix} = \begin{bmatrix} \frac{u}{R} \\ \frac{v}{R} \end{bmatrix}$$

$$R = L_9 x + L_{10} y + L_{11} z + 1$$

To improve the results obtained by this method is necessary to include in the above equations the correction of errors caused by optical lenses distortion and deviation of the optical center.

$$u = \frac{L_1 x + L_2 y + L_3 z + L_4}{L_9 x + L_{10} y + L_{11} z + 1} + \Delta u$$

$$v = \frac{L_5 x + L_6 y + L_7 z + L_8}{L_9 x + L_{10} y + L_{11} z + 1} + \Delta v$$

$$\Delta u = \xi(L_{12} r^2 + L_{13} r^4 + L_{14} r^6) + L_{15}(r^2 + 2\xi^2) + L_{16}\xi\eta$$

$$\Delta v = \eta(L_{12} r^2 + L_{13} r^4 + L_{14} r^6) + L_{15}\xi\eta + L_{16}(r^2 + 2\eta^2)$$

$$\xi = u - u_0$$

$$\eta = v - v_0$$

$$r^2 = \xi^2 + \eta^2$$

$$\begin{bmatrix} \frac{x}{R} & \frac{y}{R} & \frac{z}{R} & \frac{1}{R} & 0 & 0 & 0 & 0 & \frac{-ux}{R} & \frac{-uy}{R} & \frac{-uz}{R} & \xi r^2 & \xi r^4 & \xi r^6 & (r^2+2\xi^2) & \xi\eta \\ 0 & 0 & 0 & 0 & \frac{x}{R} & \frac{y}{R} & \frac{z}{R} & \frac{1}{R} & \frac{-vx}{R} & \frac{-vy}{R} & \frac{-vz}{R} & \eta r^2 & \eta r^4 & \eta r^6 & \xi\eta & (r^2+2\eta^2) \end{bmatrix} \begin{bmatrix} L_1 \\ L_2 \\ L_3 \\ L_4 \\ L_5 \\ L_6 \\ L_7 \\ L_8 \\ L_9 \\ L_{10} \\ L_{11} \\ L_{12} \\ L_{13} \\ L_{14} \\ L_{15} \\ L_{16} \end{bmatrix} = \begin{bmatrix} \frac{u}{R} \\ \frac{v}{R} \end{bmatrix}$$

Where L_{12} , L_{13} , and L_{14} correspond to the distortion correction, and L_{15} and L_{16} to the deviation of the optical center. By increasing the number of parameters in order to solve the equations system, it is necessary to increase the minimum calibration points up to eight. Once obtained the coefficients of DLT method, it is possible to calculate intrinsic and extrinsic parameters of the calibrated camera. With the above coefficients, whether the optical defects have been corrected or not, the following projection matrix can be created:

$$P = \begin{bmatrix} L_1 & L_2 & L_3 & L_4 \\ L_5 & L_6 & L_7 & L_8 \\ L_9 & L_{10} & L_{11} & 1 \end{bmatrix}$$

Terms (L_9, L_{10}, L_{11}) of P matrix have correspondence with the terms of the rotation matrix R , (r_{31}, r_{32}, r_{33}) , except an scale factor. It must meet that:

$$\sqrt{r_{31}^2 + r_{32}^2 + r_{33}^2} = 1$$

Therefore, one can calculate the scale factor as:

$$\sqrt{L_9^2 + L_{10}^2 + L_{11}^2} = |\lambda| \sqrt{r_{31}^2 + r_{32}^2 + r_{33}^2}$$

$$|\lambda| = \sqrt{L_9^2 + L_{10}^2 + L_{11}^2}$$

$$\begin{bmatrix} p'_{11} & p'_{12} & p'_{13} & p'_{14} \\ p'_{21} & p'_{22} & p'_{23} & p'_{24} \\ p'_{31} & p'_{32} & p'_{33} & p'_{34} \end{bmatrix} = \frac{1}{|\lambda|} \begin{bmatrix} L_1 & L_2 & L_3 & L_4 \\ L_5 & L_6 & L_7 & L_8 \\ L_9 & L_{10} & L_{11} & 1 \end{bmatrix}$$

$$\begin{bmatrix} -fx & 0 & u_0 \\ 0 & -fy & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} = \begin{bmatrix} p'_{11} & p'_{12} & p'_{13} & p'_{14} \\ p'_{21} & p'_{22} & p'_{23} & p'_{24} \\ p'_{31} & p'_{32} & p'_{33} & p'_{34} \end{bmatrix}$$

Once P matrix is normalized, the parameters of the camera can be calculated.

$$t_z = \text{sgn} \cdot p'_{34}$$

Where sgn is a sign to determine according to the position of the camera regarding the world reference frame in Z axis.

$$u_0 = p'_{11}\, p'_{31} + p'_{12}\, p'_{32} + p'_{13}\, p'_{33}$$

$$v_0 = p'_{21}\, p'_{31} + p'_{22}\, p'_{32} + p'_{23}\, p'_{33}$$

$$f_x = \sqrt{(p'^2_{11} + p'^2_{12} + p'^2_{13}) - u_0^2}$$

$$f_y = \sqrt{(p'^2_{21} + p'^2_{22} + p'^2_{23}) - v_0^2}$$

$$f = f_x d_x = f_y d_y$$

Where d_x and d_y are the distances between the centers of the camera sensor elements.

$$r_{1t} = \text{sgn} \cdot \frac{p'_{1t} - u_0 p'_{3t}}{f_x} \quad t = 1,2,3$$

$$r_{2t} = \text{sgn} \cdot \frac{p'_{2t} - v_0 p'_{3t}}{f_y} \quad t = 1,2,3$$

$$r_{3t} = \text{sgn} \cdot p'_{3t} \quad t = 1,2,3$$

$$t_x = \text{sgn} \cdot \frac{p'_{14} - u_0 t_z}{f_x}$$

$$t_y = \text{sgn} \cdot \frac{p'_{24} - v_0 t_z}{f_y}$$

The decomposition of projection matrix P , does not guarantee that rotation matrix R is orthogonal, therefore its

orthogonality must be ensured. An easy way to do this is through SVD decomposition.

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} = U \cdot D \cdot V^T$$

It is only necessary to replace D matrix by the identity l_{3x3} matrix to ensure that R is orthogonal.

B. *Levenberg-Marquardt (LM) Algorithm for Refining Camera Calibration Parameters [8]*ERROR! REFERENCE SOURCE NOT FOUND.

When initial camera intrinsic and extrinsic parameters of the linear model value are obtained by direct linear transformation method, nonlinear optimization is required to optimize parameter. For n calibration points, optimization to the intrinsic and extrinsic parameters can be performed by minimizing the objective function as shown below.

$$F(x) = \sum_{i=1}^{n} f(M_1, k_1, k_2, p_1, p_2, R_i, t_i)$$

$$= \sum_{i=i}^{n} (u_{di} - u_i)^2 + (v_{di} - v_i)^2$$

$$= \sum_{i=1}^{n} (\alpha_x x_{di} + u_0 - u_i)^2 + (\alpha_y x_{di} + v_0 - v_i)^2$$

(u_{di}, v_{di}) is the model coordinates of i points, (u_i, v_i) is the actual coordinates of i points. Levenberg-Marquardt algorithm is a typical algorithm for solving nonlinear optimization problems, combining the advantages of the steepest descent method and Gauss-Newton method. Assuming that x_k is the current solution, for equation, the trial step of Gauss-Newton method for solving the problem is:

$$d_k = -(J(x_k)^T J(x_k))^{-1} J(x_k) F(x_k)$$

Levenberg-Marquardt algorithm improves the Gauss-Newton method by adding a positive definite matrix $u_k I$ to $J(x_k)^T$ to make d_k a positive matrix. Step length of Levenberg-Marquardt algorithm is:

$$d_k = -(J(x_k)^T J(x_k) + uI)^{-1} J(x_k) F(x_k)$$

Among them, I is the unit matrix. u is called Levenberg-Marquardt parameter. When $u \to 0$, Levenberg-Marquardt algorithm tends to Gauss –Newton method; when $u \to \infty$, Levenberg-Marquardt algorithm tends to be the steepest descent method. Let the intrinsic and extrinsic parameters obtained by direct linear transformation method as the initial value. Initial values of k_1, k_2, p_1, p_2 are set as zero. Iteration algorithm steps are shown below:

1) Enter the in initial value. The initial parameters u is 0.01. Set the accuracy ε as 1e-7;

2) Calculate $F(x)$ and $J(x)$;

3) Calculate Levenberg Marquardt step d_k, then $x^{k+1} = x^k + d_k$;

4) If $F(x^{k+1}) < F(x^k)$, and $\| d_k \| < \varepsilon$, stop the iteration, and output result; otherwise, set $u = u/10$, go to Step (2);

5) If $F(x^{k+1}) \geq F(x^k)$, set $u = 10u$, recalculate d_k, return to step (4).

V. *TRIANGULATION THROUGH DLT [7]*

Since two cameras were mounted, one at the top (at ceiling looking downward) and the other gives the side view to avoid the occlusion of the peg and hole due to robot movement. We get the camera matrix *P* and *P'* for these two cameras using the calibration method discussed in section IV and then implement the triangulation method using Direct Linear Transform (*DLT*) method to get the three-dimensional world co-ordinate from the pixel position of the center of the peg and hole peg using *P* and *P'* that is discussed below.

For each input image we have a measurement $x = PX$, $x' = P'X$ where x is the 2D camera-space coordinates of a world point, x' is the same point projected into the camera-space coordinates of a second camera. X represents the 3D world coordinate that we would like to recover. These two equations can be combined into the form AX = 0, which is an equation linear in X. The homogeneous scale factor is eliminated by a cross-product to give three equations for each image point visible in more than one of the cameras in the system. As an example the equation derived for a point in the first image would be given as x × (PX) = 0. Expanded, this gives the following set of three equations:

$$x(p^{3T}X) - (p^{1T}X) = 0$$

$$y(p^{3T}X) - (p^{21T}X) = 0$$

$$x(p^{2T}X) - y(p^{1T}X) = 0$$

Combining equations from both cameras to produce an equation in the form AX = 0 gives us:

$$A = \begin{bmatrix} xp^{3T} - p^{1T} \\ yp^{3T} - p^{2T} \\ x' p'^{3T} - p'^{1T} \\ y' p'^{3T} - p'^{2T} \end{bmatrix}$$

Solving for A using SVD allows us to estimate the value of X and thus the 3D coordinate of any point for which we know the camera-space coordinates from two cameras for which the projection matrix has already been determined.

In this triangulation method *x* and *x'* are two different image coordinates generated from the same world coordinate. In this experiment we find the Harris corner detection method for the square grid and sift feature for the circular grid that is demonstrated below.

A. *Detecting Corner Points using Harris corner [10]*

The principal of Harris corner detection depends on the formula

$$E(u,v)|_{(x,y)} = \sum w(x,y)[I(x+u,y+v) - I(x,y)]^2$$

Where E represents the gray change of the image, w (x, y) is a smooth window of Gaussian and (u,v) is expressed as minimal distance. The gray of the image changes when the images moves minimal distance. Applying the Taylor series expansion to the Formula of $E(u,v)$ we can get

$$E(u,v)|_{(x,y)} = [u,v]M\begin{bmatrix} u \\ v \end{bmatrix}$$

In which

$$M = \begin{bmatrix} A & C \\ C & B \end{bmatrix}$$

Where

$$A = w \otimes Ix^2$$
$$B = w \otimes Iy^2$$
$$C = w \otimes IxIy$$

$\otimes$ represents the convolution symbol, Ix represents the value of x direction, and Iy represents the value of y direction.

M is a 2x2 matrix, and analyzing the two eigenvalues of the matrix we can determine if it is a point of interest (both eigenvalues are large), an edge (one is large, the other is small), or a plain area (both are small).

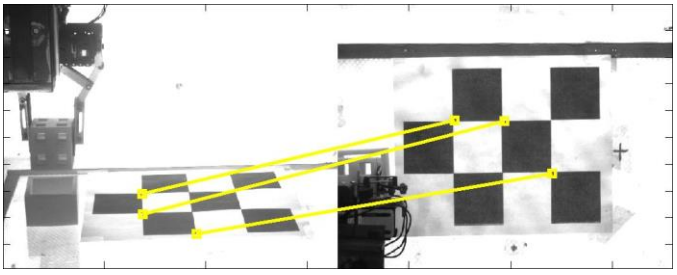

Fig. 3 Feature matching of the square grid with Harris Corner.

B. *Sift Features[13]*

The Harris Corner detection is rotation invariant, this means that the rotation of the image does not affect the ability to obtain results. But with scale, this is not the case, a corner could stop being a corner if the image is scaled, to solve this, a Scale Invariant Feature Transform (SIFT) is used. This process is composed of five steps:

1) ***Scale-space Extrema Detection***: Here a scale-space filtering is used. The Laplacian of Gaussian (LoG) is found for the image with various σ values. The LoG acts as a blob detector of various sizes due to changes of σ, in other words, σ acts a scaling parameter. Because LoG is costly, a Difference of Gaussians (DoG) is used for SIFT. The DoG is obtained by subtracting 2 images with different σ. This process is done for various octaves of the image in Gaussian Pyramid. Once DoG is found, images are searched for local extrema over scale and space. If it is a local extrema, it is a potential keypoint.
2) ***Keypoint Localization*:** Once potential keypoints locations are found, they are processed through a filter with a concept similar to Harris corner detector for results with more accuracy. The objective is to remove keypoints with low contrast and edge keypoints.
3) ***Orientation Assignment*:** An orientation is assigned to each keypoint so that the image becomes rotation invariant. The keypoint with the highest peak and keypoints with higher than 80% of the peak value are used for this step. This creates keypoints with same scale and location, but different directions. It contributes to stability of matching.
4) ***Keypoint Descriptor*:** Around each keypoint, a 16x16 neighbourhood is created. This is divided in 16 sub-blocks of 4x4. For each sub-block, an 8 bin orientation histogram is created. So total of 128 bin values are available. It is a vector to form keypoint descriptor. Even more, it also includes measures against illumination changes, rotation, etc.
5) ***Keypoint Matching*:** For the matching, their nearest neighbourhoods are identified. In the case that a second match is really close to the first, ratio of closest-distance to second-closest distance is taken. If it is greater than 0.8, they are rejected. This takes care of 90% of the false matches.

VI. Optimum Camera Parameters for Triangulation

We need to choose the optimum values of the camera parameters for obtaining better triangulation. For finding the optimum values of the camera parameters we used two different types of grid: square and circle as shown in Fig. 2 and Fig. 3 respectively. There are two different types of calibration error: (*i*) reprojection error and (*ii*) rectification error. For binocular calibration, reprojection error performs better than reprojection error [9]. Different calibration errors are defined as follows,

A. *Reprojection Error [9]*:

Let $P_i = K \cdot [R_i \,|\, t_i]$ be the projection matrix of camera c for calibration grid view i. Defining k as the number of grid points detected in the image at the coordinates x_j, corresponding to 3D planar points X_j. Then, the reprojection error for image i is defined as,

$$e^c_{rep}[i] = \frac{1}{k}\sum_{j=1}^{k} \| P_i(X_j) - x_j \|$$

A low reprojection error indicates an accurate projection matrix, at least for the points on the plane that were used to compute the projection matrix. This can be further improved if there were additional 3D points available for which we had

corresponding detected 2D pixels. One camera is not enough to get a 3D location of point Q, however, if two cameras observe the same calibration grid sequence a pair-wise calibration algorithm can be done using the rectification error.

B. Rectification Error [9]: When two cameras observe the same sequence of calibration grid locations, all grids can be used to evaluate the calibration accuracy for each individual set of extrinsic parameters. If a point, Q, on some grid is visible in both cameras then the projection of Q onto the rectified versions of the left and right images should lie in the same scanline, if the calibration of the cameras is accurate. As this is independent of the 3D location of Q, we are able to use all detected points from all grid locations that are common in both views.

From this, we can measure the rectification error for two cameras, c_1 and c_2, and calibration grid view as follows. For each calibration grid, let the k^{th} detected grid point on the image plane of c_1 corresponding to unknown 3D point Q^k be $q_1^k = (u_1^k, v_1^k)$, and on the image plane of c_2 be $q_2^k = (u_2^k, v_2^k)$. For $c \in \{1,2\}$, we denote $q_c^k[0]$ to refer to u_c^k and $q_c^k[1]$ to refer to v_c^k.

$$e_{rect}^{c_1}[i] = \frac{1}{N}\sum_{j=1}^{N}\left(\frac{1}{M_j}\sum_{k=1}^{M_j}\left| T_i^{c_1} q_1^k[1] - T_i^{c_2} q_2^k[1] \right|\right)$$

Where $T_i^{c_1}$ is the rectifying transformation for camera c_1 using calibration i, and $T_i^{c_2}$ is the same for c_2. N is the total number of grid position in the sequence, and M_j is the number of grid points that are commonly detected in both camera views for grid position j. We compute the rectifying transformations using the method of Fusiello *et al.* [11]. The rectification error in a particular point Q is defined as:

$$q'_1 = T_i^{c_1} q_1,$$

$$q'_2 = T_i^{c_2} q_2,$$

The rectification error measure can now be used to determine more accurate binocular camera calibrations than the standard method using the reprojection error.

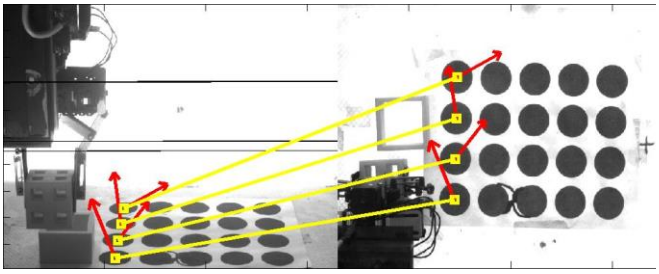

Fig. 4 Feature matching of the circle with sift feature.

C. Correction Approaches: Three different approaches are used to correct the reprojection or the rectification error.

1) Global Reprojection Error [9]: Here, let S be the set of all calibration grids visible in every camera view. Then we choose the single grid location that yields the lowest average reprojection error among all N_c cameras.

$$\min_{i \in S} \sum_{c=1}^{N_c} (e_{rep}^c[i])$$

All the cameras will be calibrated to the same coordinate system, but the problem is that the cameras will not be calibrated with the same quality.

2) Pair-wise Reprojection Error [9]: For this we use all the calibration grids (S) visible by the pair of cameras that yields the lowest average reprojection error for those two cameras.

$$\min_{i \in S} \sum_{c \in \{c_1, c_2\}}^{N_c} (e_{rep}^c[i])$$

This approach allows for more grids to choose from, resulting in lower reprojection errors, but some calibrations results will be misleading, as some will not result accurate.

3) Pair-wise Rectification Error [9]: This is the same as the Pair-wise Reprojection Error, with the difference that it uses the lowest average rectification error for the cameras. This method gives the most accurate calibrations on average.

$$\min_{i \in S} \sum_{c \in \{c_1, c_2\}}^{N_c} (e_{rect}^c[i])$$

We choose pair-wise rectification error for finding the optimum parameter values of the two cameras *P* and *P'* which were used for the calibration method described in section VI.

VII. EXPERIMENTAL RESULTS AND DISCUSSIONS

We developed and executed the matlab code for this experiment in a personal desktop of 4 GB RAM, Windows 10, AMD A6-4400M with randeon (TM) HD Graphics 2.70 GHz. This experiment was conducted with two cameras observed the same sequence of calibration grid locations, all grids can be used to evaluate the calibration accuracy for each individual set of camera parameters as shown in Fig. 3. Two cameras were used to avoid the occlusion of the peg and hole due to robot movement. Feature matching of the grid corners are matched through Harris corner as shown in Fig. 3. Feature detection of the circular grid are matched through sift features are demonstrated in Figure 4. We used two different grids, square and circular to find the optimum camera parameters required for triangulation method using pairwise rectification error based calibration error as discussed in section VI. Results of the template matching algorithm using fast correlation based block matching are demonstrated in Fig. 5. Time taken for three different template matching algorithms, such as Fast correlation based block matching, zero mean normalized cross correlation and normalized cross correlation are demonstrated in Table 1. Table 2 shows the experimental error for different cross correlation techniques and calibration

algorithms. Fig. 6 shows automated peg-hole insertion tasks by gantry robot using proposed computer vision algorithms.

We captured 18 corner points using Harris corner detection from square grid and 20 center points of the circle using sift features along with relative position of the circles automatically and recorded the corresponding world co-ordinates. For obtaining camera parameter values from DLT method for 3D, we need 8 points. We implemented a cross validation technique [16] and identified those points along with the grid having minimum pairwise rectification error gives the optimal values of the camera parameters which were used for the triangulation method discussed in section V.

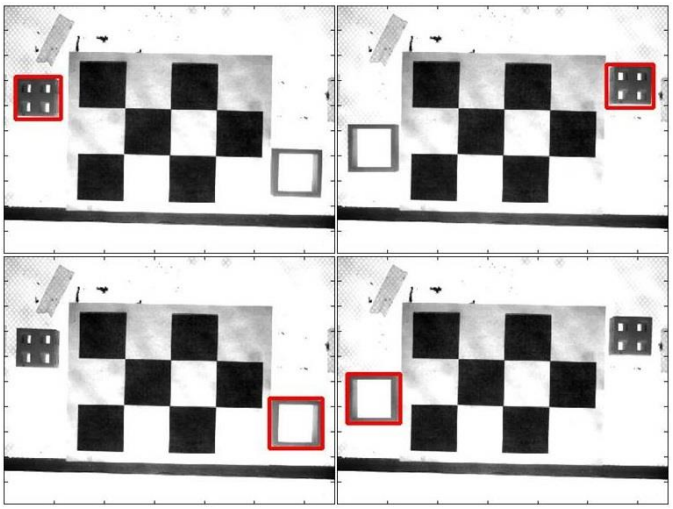

Fig.5 Results of automatic peg and hole detection using fast correlation based block matching algorithm. Top row shows two different positions of the peg in two different images. Bottom row shows two different positions of the hole in the same images. Automatic detection of the peg and hole is marked by red color in the figure.

Table 1. Time taken by template matching algorithms

Image Name	Image size	Template size	Template Matching Algorithm		
			Fast Template	ZNCC	NCC
Image1	494×659	86×93	92 seconds	128 seconds	2286 seconds
Image 2	494×659	86×93	95 seconds	120 seconds	2231 seconds

Table 2. Experimental error

Name of the algorithm	Error
Template matching	
Fast correlation based block matching	3%
Zero mean normalized cross correlation	4%
Normalized cross correlation	5%
Calibration algorithm	
Circular grid	± 2.5 m. m.
Square grid	± 1.5 m. m.

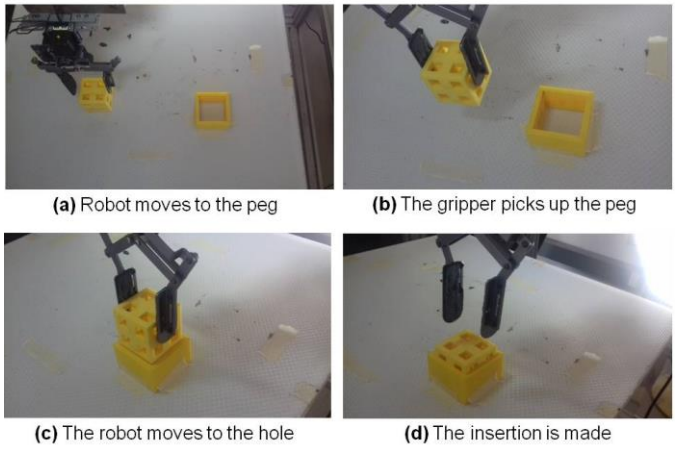

Fig. 6 Automated Peg-Hole insertion by gantry robot using proposed computer vision based algorithms (efficient integration of template matching, calibration and triangulation).

VIII. Conclusions

We have efficiently assimilated template matching, calibration and triangulation algorithms to automate the peg-hole insertion task conducted by a gantry robot using a pair of cameras. We used two cameras to avoid the occlusion of the peg and hole due to movement of the robot as well reduce the calibration error. We first implemented a fast correlation based block matching algorithm to automatically find the peg and hole using two cameras. Templates of the peg and hole at different orientation and illumination were utilized to improve the performance of the template matching algorithm. Then a Direct Linear Transform (DLT) method based calibration algorithm was implemented to find the intrinsic and extrinsic parameters of the camera. Levenberg-Marquardt (LM) based non-linear optimization method was further used to refine the camera calibration parameters. Finally we used a DLT based triangulation method to find the three dimensional world coordinates of the peg and hole captured by two cameras. Optimum camera parameters were found by pairwise rectification error based calibration error to reduce the triangulation error. We used two different grids: square and circular to compute the pair-wise rectification error. Feature matching algorithms required for triangulation method, we used Harris corner detection for square grid and sift features for circular grid. Experimental results demonstrate that proposed computer vision method by efficient integration of template matching, calibration and triangulation method can successfully automate peg hole insertion task.

Acknowledgment

We would like to acknowledge Conacyt for supporting our research.

References

[1] A. Mahmood, and S. Khan, "Early termination algorithms for correlation coefficient based block matching," In: *Proceedings of the*

International Conference on Image Processing, ICIP, pp. II 469–II472 , 2007.

[2] L. Di Stefano, S. Mattoccia,, and F. Tombari, "Zncc-based template matching using bounded partial correlation," *Pattern Recognition Letters*, vol. 26, pp. 2129-2134, 2005.

[3] Choi, J.W., Fang, T.H., Yoo, W.S., Lee, M.H., "*Sensor* data fusion using perception net for a precise assembly task." *IEEE/ASME transactions on mechatronics* 8(4), 513–516 (2003).

[4] Kim, J., Cho, H.S., "Visual sensor-based measurement for deformable peg-in-hole tasks." In*: Intelligent Robots and Systems, 1999. IROS'99. Proceedings*. *1999 IEEE/RSJ International Conference* on. vol. 1, pp. 567–572. IEEE (1999).

[5] Xue, G., Fukuda, T., Arai, F., Asama, H., Kaetsu, H., Endo, I., "Dynamically reconfigurable robotic system assembly of new type cells as a dual-peg-in-hole problem." In: *Distributed Autonomous Robotic Systems*, pp. 383–394. Springer (1994).

[6] Newman,W.S., Zhao, Y., Pao, Y.H., "Interpretation of force and moment signals for compliant peg-in-hole assembly." In*: Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference* on. vol. 1, pp. 571–576. IEEE (2001).

[7] Hartley, R.,I. and Zisserman, A., "Multiple View Geometry in Computer Vision", *Cambridge University Press*, 2004

[8] Tian Shao-xiong, Lu Shan, Liu Zong-ming, "Levenberg-Marquardt Algorithm Based Nonlinear Optimization of Camera Calibration for Relative Measurement" *Proceedings the 34th Chinese Control Conference*, 2015

[9] Bradley, Derek, Wolfgang Heidrich "Binocular Camera Calibration Using Rectification Error", In: *CRV, IEEE Computer Society*, pp. 183-190, 2010

[10] Zhang, X., He, G., & Yuan, J. (2009, October). "A rotation invariance image matching method based on Harris corner detection." In: *Image and Signal Processing, 2009. CISP'09. 2nd International Congress* on (pp. 1-5). Ieee.

[11] Fusiello, A., Trucco, E., & Verri, A. (2000). "A compact algorithm for rectification of stereo pairs." *Machine Vision and Applications*, 12(1), 16-22.

[12] Lewis, J. P. "Fast normalized cross-correlation." *Vision interface*. Vol. 10. No. 1. 1995.

[13] Lowe, David G. "Distinctive image features from scale-invariant keypoints."*International journal of computer vision 60.2* (2004): 91-110.

[14] Daehie Hong, Woo-Chang Lee, Baeksuk Chu, Tae-Hyung Kim, Woo Chun Choi "Gantry Robot with Extended Workspace for Pavement Sign Painting Operations." *International Journal of Precision Engineering and Manufacturing 9.3* (2008): 85-91.

[15] Shapiro, Robert. "Direct linear transformation method for three-dimensional cinematography." *Research Quarterly. American Alliance for Health, Physical Education and Recreation* 49.2 (1978): 197-205.

[16] T. Hastie, R. Tibshirani, and J. Friedman, "The Elements of Statistical Learning, Data Mining, Inference, and Prediction," Second Edition, 2008.

[17] J. D. Ratcliffe, P. L. Lewin, E. Rogers, J. J. Hätönen, and D. H. Owens, "Norm-Optimal Iterative Learning Control Applied to Gantry Robots for Automation Applications," *IEEE Transactions on Robotics*, vol. 22, no. 6, pp. 1303–1307, 2006.

[18] L. Balletti, A. Rocchi, F. Belo, M. Catalano, M. Garabini, G. Grioli, and A. Bicchi, "Towards variable impedance assembly : the VSA peg-in-hole." , IEEE-RAS International Conference on Humanoid Robots (Humanoid), pp. 504-508, 2012.