Int J Adv Manuf Technol (2010) 48:251–265
DOI 10.1007/s00170-009-2269-3

ORIGINAL ARTICLE

On-machine 3D vision system for machining setup modeling

Xi Zhang · Xiaodong Tian · Kazuo Yamazaki

Received: 5 March 2009 / Accepted: 12 August 2009 / Published online: 2 September 2009

Abstract In computer numerical control machine tools, using machining simulation to prevent collision becomes more and more popular due to its efficiency and low cost. However, if the entire digital model of the machining setup does not exist, the simulation is not applicable. As a result, the operator has to manually check the numerical control program, which is a time-consuming and error-prone process. In this paper, an on-machine vision system is presented to quickly construct the digital model based on the actual machining setup. The total construction for a complex setup can be done within a few minutes. Several key technologies have been developed. First, a 2D edge feature detection algorithm has been designed which will extract the edges of the object of interest by processing both the real and virtual images. Second, a stereo vision system is developed which will obtain the three-dimensional (3D) edge data of the object of interest. A new algorithm is presented to solve correspondence, which is the key problem of the stereovision system. Furthermore, the 3D object recognition algorithm is developed to let the system intelligently search for the matched solid model in the database and import it into the virtual environment with an accurate pose. Finally, experiments are carried out to test the developed system.

X. Zhang (✉) · K. Yamazaki
IMS-Mechatronics Laboratory,
Mechanical and Aerospace Engineering Department,
University of California,
Davis, CA, USA
e-mail: xizhang@ucdavis.edu

X. Tian
Digital Technology Laboratory Corporation,
Sacramento, CA, USA

Keywords Machining setup · Edge detection · Correspondence · Object recognition

1 Introduction

In the entire process of machining, simulation plays a more and more important role. Originally, the machining simulation only simulates the cutting motion between a cutter and a workpiece, but current simulation includes workpiece, fixtures, cutter, and even the whole machining center structure. In order to fully utilize the merits of simulation, the accuracy of the digital model of the machining setup becomes critical. Currently, the machining setup is conducted by the operator following the setup planning. Owing to different factors, the final physical setup, after the installation, may not be the same as the planned model.

The operator has to spend plenty of time to check the setup and verify the numerical control code manually. It not only consumes time but also increases the chance to fail the machining process or even cause a collision with the machine tool. Therefore, the identicalness of the machining setup and its digital model is an important issue. In order to guarantee the identicalness of the setup and model, there are two types of solutions based on the current research activities. One is the forward solution, which focuses on how to guarantee and adjust the physical setup to comply with its digital model. The other is the reverse solution, which focuses on how to guarantee and adjust the digital design to comply with the accomplished physical layout.

For the forward solution, one of the promising research works is the automatic fixture layout design. It gains more popularity [1–3] since it increases the accuracy and productivity of the machining process. Commercially available modular fixturing systems typically include a

lattice of holes with precise spacing and an assortment of locating and clamping modules that can be rigidly attached to the lattice. It will definitely ensure the positioning accuracy when operators install fixtures and workpieces according to the designed layout. However, the high cost of the modular fixture limits its wide application.

The reverse solution can be based on two kinds of approaches. The first one is to use three-dimensional (3D) object measurement. The second is to use 3D object recognition.

For the 3D object measurement, both the contact and non-contact probes are used. However, the direct output of either contact or non-contact probe is the points cloud, which cannot be directly used for machining simulation, because it is a discrete model. Meanwhile, the whole measurement process is time-consuming.

3D object recognition plays a prominent role in the robot area since it entails a number of fundamental problems in computer vision. During the past two decades, 3D object recognition has had a great deal of activity, as pointed out in the surveys [4–6]. Traditionally, a model-based recognition system would include the following sequence of tasks: sensory data acquisition, low level processing of the sensory input, feature extraction, perceptual organization (or feature grouping), scene-to-model hypothesis verification, and pose estimation. The 3D object recognition is able to quickly recognize and position the part if pre-knowledge of the part is known, such as the digital model of the part.

Nowadays, many industrial parts are designed in 3D solid modeling software. Their solid models are all made as the design data. Figure 1a and b show the solid models of a machining center and a fixture. Furthermore, in order to precisely control the machining costs, the raw workpiece also has its solid model before it is machined (Fig. 1c). From this point of view, it is more efficient and accurate to choose the 3D object recognition technology in order to construct the digital model of the machining setup.

Therefore, the 3D object recognition method is chosen in this paper. A 3D vision system for quick machining setup modeling is proposed. First, the images of components of machining setup are taken by an on-machine stereovision system. By using the virtual images of the worktable as the background, the 2D edges of the object of interest can be extracted from images reliably. Then, the 2D edges obtained from left and right images are matched in order to reconstruct the 3D edges. At last, the reconstructed 3D edges are used to recognize the matched solid model in the database and estimate its position and orientation.

The paper is organized as follows. Section 2 proposes the conceptual design of the quick machining setup modeling system. In section 3, the critical algorithms involved in system implementation are developed and verified, which include composite background subtraction and 2D edge extraction, edge correspondence, as well as object recognition algorithm. In section 4, a prototype system is built to test the feasibility of the proposed machining setup modeling system.

2 Conceptual design

Before giving the proposed solution, we formulate the problem in a more specific way. Given a physical machining setup P inside a machine tool, its accurate digital model D should be constructed within several minutes without or with little human assistance. Based on the analysis in previous section, the concept of this system is illustrated in Fig. 2. The system assumes that the final setup model can be represented as an assembly of the components, and each component is stored in a computer database as a 3D solid model. These components can be the machine tool, cutting tool assembly, workpiece, pallet, jigs, and fixture modules. The assumption is supported by the fact that most of the mechanical design process is carried out using 3D CAD design systems today. In order to construct the digital model of the machining setup, a 3D vision system without assistant lighting is proposed as shown in Fig. 3. Notice two charge-coupled device (CCD) cameras (left and right) are installed into a machine tool to

(a) Machine tool (b) fixture (c) Work piece

Fig. 1 Solid model for industrial product

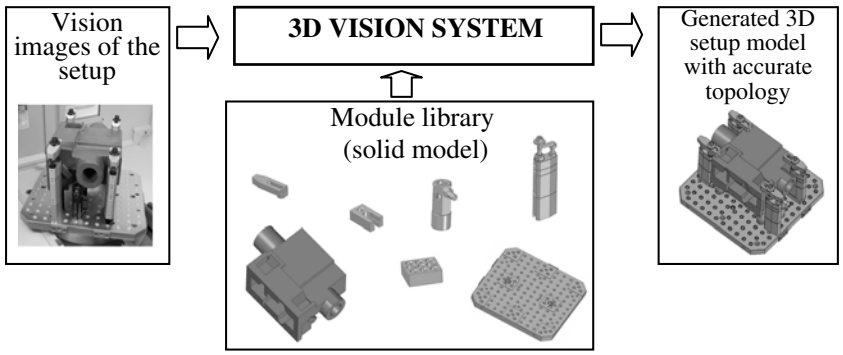


Fig. 2 System concept

take pairs of pictures of the machining setup. The captured images are processed by the 2D feature extraction module to extract the 2D edge data of the part in the machining setup. These 2D edges from the left and right images are inputted into the 3D feature generation module to generate the 3D edge data in the 3D space. Thereafter, these 3D edges will be compared with the solid model database to find out the best match in the object recognition module. The matched solid model is imported into the virtual environment to update the entire digital model. Solutions to several key problems will be elaborated on in the following section. For example, in the 2D feature extraction module, a composite background subtraction is designed using both real and virtual images to automatically extract the 2D edges of the part from the real image. One robust correspondence engine is designed to accurately find the matched edges between the images captured from the left and right cameras. In the object recognition module, 3D edges are retrieved from the solid model and compared with system-generated 3D edges. The matched solid model is imported into the virtual environment with an optimized pose.

3 System implementation

3.1 Composite background subtraction and 2D edge extraction

For a better explanation, the problem is formulated as: given an image I which contains the object of interest, extract the 2D edges of that object only. To solve the above problem, the key is in knowing how to segment the object of interest from the others. Background subtraction is a technique that is frequently used. Generally, most research focuses on comparing color or intensities of pixels in the foreground image, which contains the object of interest, with a background image which does not contain the object of interest. This technique requires a static background view [7–9]. However, in a machine tool environment, the work table is often moved during the process of installation so that the background keeps changing. Thus, with the assumption of geometrically static backgrounds, the above method is rendered unsuitable for the machine tool environment.

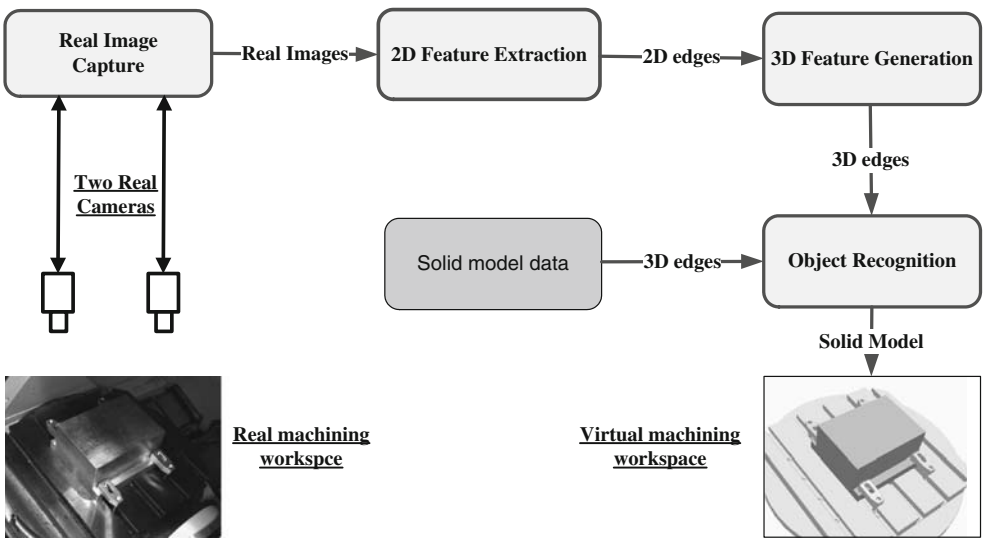


Fig. 3 Stereo vision system for machining setup modeling

An accurate virtual CCD camera is presented, whose parameters are the same as the real one. By the virtual CCD camera, a virtual image can be generated, which is the projection of the 3D solid model onto the virtual CCD camera [10]. Although the real work table moves or rotates during the machining setup, the motion synchronizer [10] will automatically synchronize the motion of the virtual work table with the real work table. Therefore, the virtual image is always the same as the real one in spite of the motion of the real work table. Using this virtual image as the background, the background subtraction can be accomplished dynamically. However, since the illumination and texture in the machine tool environment is unpredictable and difficult to model in the virtual image, only geometric features of background objects in the virtual image are used to match their counterparts in the real image. Therefore, using the virtual image as the background image and then the real image as the foreground image, a composite geometry-based edge extraction method is designed and implemented as shown in Fig. 4.

- Input. The inputs are the real and virtual images, either from the left side CCD camera or the right side. The real image is the foreground image which contains the object of interest, such as a workpiece or fixture. The virtual image is the background which only reflects the background objects. The virtual image is in wire frame format. All visible edges of the background objects in the virtual environment are known.
- Corner detection. The corner detection aims at an accurate extraction of the corners in the real image, both for the object of interest and other objects. In this paper, Harris corner detector [11] is adopted. All the detected corners are stored in the set $C=\{(x_i, y_i)\}$, where (x_i, y_i) are the corner location in the image.
- Corner suppression. Obviously, set C contains all the corners which come from the object of interest, background object, and other noises, such as lighting reflection and surface scratch. Those corners from the background object should be eliminated as much as possible. The virtual image contains all the visible edges of the background objects. The rule to overlap corners in set C onto the virtual image is given as: if the corner is located on the edge of the virtual image, this corner is considered as a corner from the background object, which should be eliminated from C.
- Edge hypothesis. All the remaining corners in C can be classified into the object of interest corners and the noise corners. We notice that the noise corners are often isolated and do not have the edge connection between each other. Therefore, the following assumption is made: there is no obvious edge between any two noise corners. Then, a hypothesis is made that an edge L_{ij} exists between every two corners (x_i, y_i) and (x_j, y_j) in C. If the hypothesis of the L_{ij} can be validated, it means that the two corners are the corners on the object of interest, and an edge does exist between these two corners.
- Edge validation. To validate the above hypothesis, an automatic validation has been designed as follows:

  (a) Obtain the direction, length in pixel unit of L_{ij}, as well as all pixels which are locate on L_{ij}.
  (b) For each pixel on L_{ij}, find its eight neighboring pixels along the normal direction of the L_{ij}. The normal direction is orthogonal to the direction of L_{ij}. All the neighboring pixels form a striped region. In the region, the magnitude of the gradient is calculated. The local gradient maximum pixels are determined if its gradient magnitude is greater than its two continuous neighbors along the direction of L_{ij}.
  (c) Determine the gradient threshold T based on gradient magnitudes of all pixels on the edge L_{ij} and their neighbors. In this paper, T is set as the average value of the gradient magnitudes.
  (d) For each pixel on L_{ij}, add one to a counter in the following two cases: case 1, if this pixel is located close enough to the local gradient maximum pixel and if its local gradient maximum is higher than the threshold T; case 2, if at least one of its eight neighbors has gradient magnitude higher than the threshold T. All these pixels are named as edge pixels.
  (e) If the ratio of the current number of the counter to the length of L_{ij} is less than a preset value, for

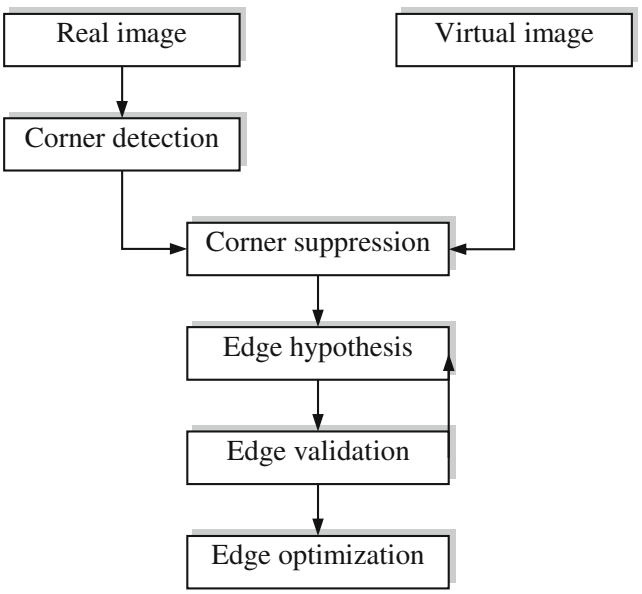


Fig. 4 Composite background subtraction and edge extraction

example 0.8 in this paper, it indicates the edge L_{ij} hypothesis is wrong. Otherwise, L_{ij} does exist between these two corners.

(f) If more than one edge share the same corners and these edges have the same direction, they are combined as one edge.

(g) For each validated edge L_{ij}, its edge is refined by least-squares method based on all its edge pixels.

Here are several advantages of this algorithm. First of all, it effectively suppresses the noise corners. Secondly, it is a unified process for both edge detection and noise corner suppression. Finally, since the threshold T is determined by the local image property, it is not necessary to manually select it. On the other hand, this algorithm requires that the difference between the foreground and background image is only one object of interest. For example, the background image contains the work table. The foreground image contains the work table and only a workpiece or a fixture. If the foreground image contains the work table, a workpiece, and a fixture, the extracted edges will be the combination of the workpiece and the fixture. These combined edges will cause complications for the object recognition process outlined in this paper. In such a case, human assistance is introduced to select the corners for one object of interest. This system uses these corners to continuously conduct the edge hypothesis and edge validation, which is explained above.

The proposed method is tested on the Mori Seiki NV-5000 machine tool. Two SONY XCD-710 CCD cameras are installed inside the machine tool. On the work table, a workpiece is installed. After performing the camera calibration [10] and real and virtual image compensation, the captured real image and virtual CCD camera-generated virtual images from the left side images are shown in Fig. 5a and b. For every pixel in the real image shown in Fig. 5a, the Harris corner detector is applied to obtain its corner response by considering the 9×9 neighboring pixels. The detected corners are marked by red circular points on the image as shown in Fig. 5c. Some of the corners are caused by light reflection and surface scratches. All visible background edges shown in Fig. 5b are obtained as shown in Fig. 5d. Then, all the corners are processed by the algorithms of corner suppression and edge validation described above. The final result is shown in Fig. 5e as edges. In these edges, most of the edges do correspond to the integral edge of the workpiece, some of them only correspond to the partial edge.

3.2 Edge correspondence engine

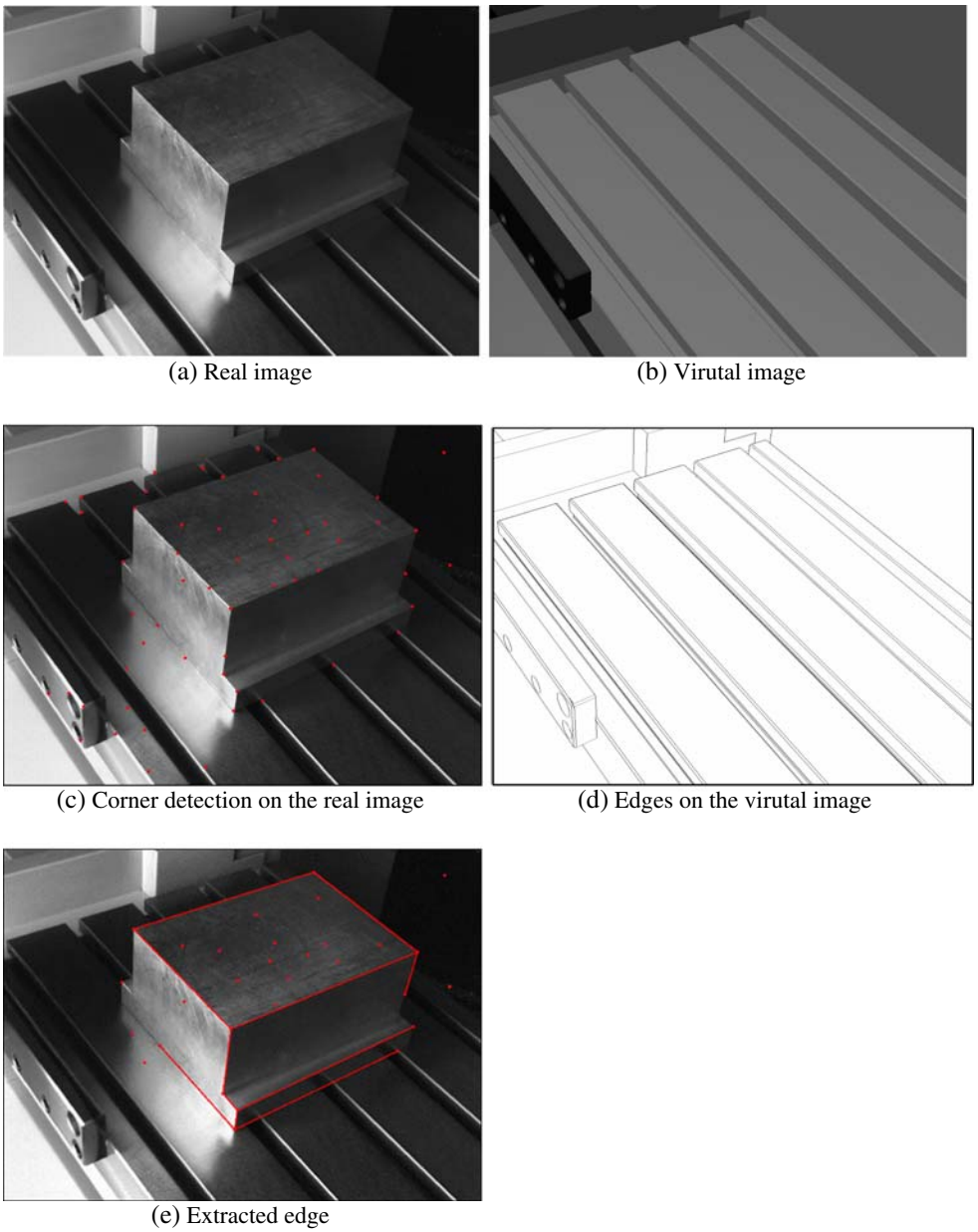
In a stereovision system, given one point (edge or object) in the left image, its correspondent point (edge or object) needs to be identified in the right image, or vice versa. Since the correspondence is a reverse problem, it is considered to be greatly ill-posed. One current solution to the correspondence problem is named image block matching. It estimates the disparity at a point in the left image by comparing a small region about that point with a series of small regions extracted from the right image [12, 13]. It is also well known that this method is sensitive to depth discontinuity regions as well as the regions of uniform texture. However, in the machining setup, all the parts usually have the uniform metallic texture. Therefore, a correspondence method which can work in such a case is designed in this paper.

All the extracted edges from the left image are considered as one set, labelled as LES. All the extracted edges from the right image are considered as another set, labelled as RES. All the edges in LES are sorted as the incremental order of the x value of the middle point of the edge. The x value is based on the corresponding image coordinate. Likewise, the edges in the RES are sorted as the same way.

The LES is classified into three groups. In the first group, each edge connects to other edges at both of its end points. This group is named as “closed edge”. In the second group, each edge connects to other edges only at one end point. The other end point does not connect to any other edges. This group is referred to as “half-closed edge”. In the third group, each edge does not connect to any other edges. Its two end points are all open. This group is referred to as “open edge”. The pseudo-code to classify the LES into three groups is shown below.

```
For each edge Li (i=1~n) extracted from the image,
where n is the number of the edges
  If the first end point connect to other edges
    Then Flag1=1
  Else Flag1 =0;
  If second end point connect to other edges
    Then Flag2=1
  Else Flag2 =0;
  If flag1=1 and flag2=1
      Li is a closed edges
  If flag1=0 and flag2=0
      Li is an open edge
  If (flag1=1 and flag2=0) or (flag1=0 and flag2=1)
      Li is a half-closed edge
```

For RES, all the edges are classified into three groups in much the same way. The overall algorithm is shown in



(a) Real image (b) Virutal image

(c) Corner detection on the real image (d) Edges on the virutal image

(e) Extracted edge

Fig. 5 Hypothesis-based edge extraction

Fig. 6. There are two characteristics in this algorithm. First, it uses a divide-and-conquer strategy. We notice that the closed edge is the highest reliable edge in the image, which means the closed edge has the least possibility to be the edge caused by shining and reflection. Meanwhile, it also means the closed edges have the highest possibility to be detected on both images. The half-closed edges are the second grade. The open edges are the third grade. Therefore, the correspondence search is classified into three phases starting with the closed edges. Second, once some correspondences between edges are found, these established edges will form a global correspondence structure, which assists the rest of the unmatched edges to find the correct match.

The correspondence algorithm for each phase is illustrated in Fig. 7. For two edges, their absolute similarity is checked by the absolute evaluation functions. In this paper, these functions are presented as the Epipolar evaluation function, the length evaluation function, and the absolute orientation evaluation function. Moreover, the tested two edges are also evaluated by checking the relative relationship between them and the already matched edges. It is named as the relative evaluation functions, which includes the relative orientation evaluation function and the relative angular evaluation function in this paper. After all the edges are evaluated, the values are used as input into the dynamic programming to optimize an overall best correspondence result.

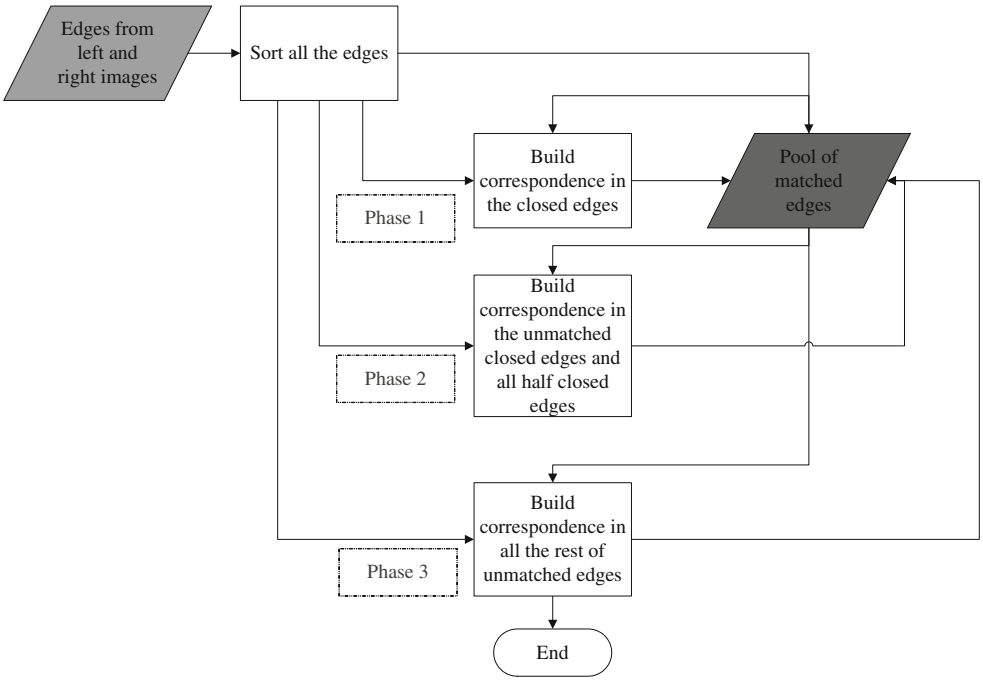


Fig. 6 Overall flow of correspondence algorithm

3.2.1 Epipolar evaluation function (EEF)

Since the two CCD cameras are accurately calibrated, the natural Epipolar line EL_i for the middle point C_{L_i} of L_i can be explicitly calculated, shown as the horizontal edge in Fig. 8. Given one edge from the right side as R_j and its middle point C_{R_j}, their EEF (L_i, R_j) is:

$$\mathrm{EEF}(L_i,R_j)=\begin{cases}0, & \text{if} \quad \mathrm{Dist}(C_{R_j},EL_i)>T_{d\_n}, \quad n\in[0,5)\\ 1-\mathrm{Dist}(C_{R_j},EL_i)/T_{d\_n}, & \text{else}\end{cases} \tag{1}$$

where $\mathrm{Dist}(C_{R_j},EL_i)$ is the distance between C_{R_j} and EL_i in pixel unit, and T_{d_n} is the threshold in five levels. In the first level, T_{d_n} is set as 5. T_n increases by 2 in each following level.

3.2.2 Length evaluation function (LEF)

Given two edges, L_i and R_j, their LEF (L_i, R_j) is:

$$\mathrm{LEF}(L_i,R_j)=\begin{cases}0, & \text{if} \quad \dfrac{\min(\mathrm{Len}(L_i),\mathrm{Len}(R_j))}{\max(\mathrm{Len}(L_i),\mathrm{Len}(R_j))}<T_{l\_n}, \quad n\in[0,5)\\ \dfrac{\min(\mathrm{Len}(L_i),\mathrm{Len}(R_j))}{\max(\mathrm{Len}(L_i),\mathrm{Len}(R_j))}, & \text{else}\end{cases} \tag{2}$$

where Len (L_i) is the length of L_i, and Len (R_i) is the length of R_i. T_{l_n} is the threshold in five levels. In the first level, T_n is set as 0.9. T_{l_n} decreases by 0.2 in each following level.

3.2.3 Orientation evaluation function (OEF)

Given two edges, L_i and R_j, the angle between each of them to the x-axis are α and β, respectively. Their OEF (L_i, R_j) is:

$$\mathrm{OEF}(L_i,R_j)=\begin{cases}0, & \text{if} \quad |\alpha-\beta|>T_{o\_n}, \quad n\in[0,5)\\ 1-|\alpha-\beta|/T_{o\_n}, & \text{else}\end{cases} \tag{3}$$

where T_n is the threshold in five levels. In the first level, T_{o_n} is set as $\pi/32$. T_{o_n} increases by $\pi/64$ in each following level.

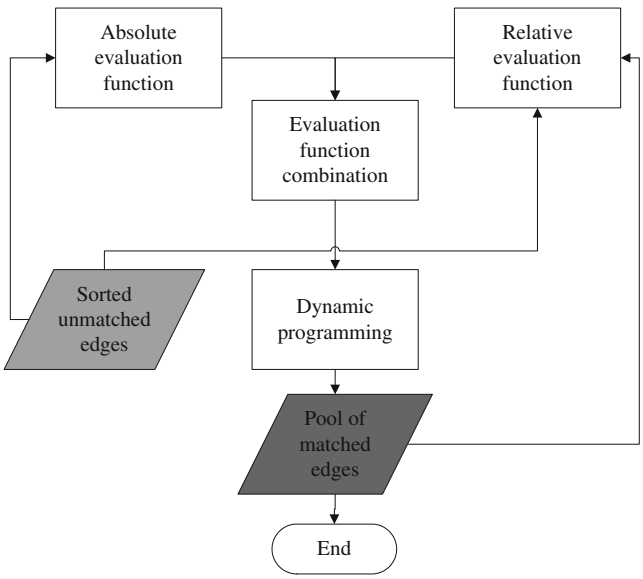


Fig. 7 Correspondence method

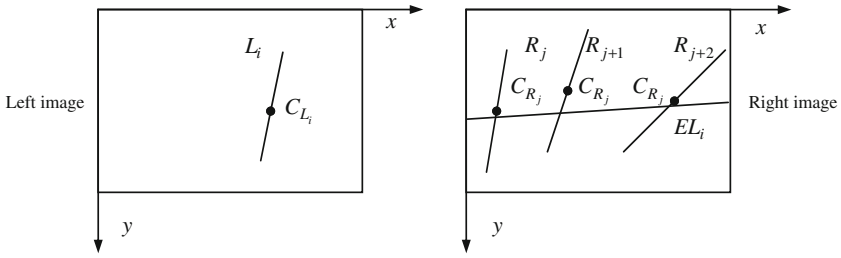


Fig. 8 Epipolar evaluation function

3.2.4 Relative orientation evaluation function (ROEF)

Once the matched edges are found, ROEF is considered for finding the next correspondence. Given two edges, L_i and R_j, L_m and R_m which are the matched edges, the ROEF is:

$$\mathrm{ROEF}(L_i,R_j)=\begin{cases}0, & \text{if} \quad \frac{1}{k}\sum_{m=1}^{k}\left(\left|\theta_l(L_i,L_m)-\theta_r(R_j,R_m)\right|\right)>T_{ro\_n}, \quad n\in[0,5)\\ 1-\frac{1}{k}\sum_{m=1}^{k}\left(\left|\theta_l(L_i,L_m)-\theta_r(R_j,R_m)\right|\right)/T_{ro\_n}, & \text{else}\end{cases} \tag{4}$$

where θ_l (L_i, L_m) is the angle between these two edges in the left image, and θ_r (R_j, R_m) is the angle between these two edges in the right image. k is the total number of the matched edge pairs. T_{ro_n} is the threshold in five levels. In the first level, T_{ro_n} is set as $\pi/32$. T_n increases by $\pi/64$ in each following level.

3.2.5 Relative angle evaluation function (RAEF)

Once the matched edges are found, RAEF is considered for the next correspondence finding. Given two edges, L_i and R_j, L_m and R_m which are the matched edges, the RAEF is:

$$\mathrm{RAEF}(L_i,R_j)=\begin{cases}0, & \text{if} \quad \frac{1}{k}\sum_{m-1}^{k}\left(\left|\phi_l(L_i,L_m)-\phi_r(R_j,R_m)\right|\right)>T_{ra\_n}, \quad n\in[0,5)\\ 1-\frac{1}{k}\sum_{m=1}^{k}\left(\left|\phi_l(L_i,L_m)-\phi_r(R_j,R_m)\right|\right)/T_{ra\_n}, & \text{else}\end{cases} \tag{5}$$

where ϕ_l (L_i, L_m) is the angle between the edge which connects the middle point of L_i to the middle point of L_m, and x-axis in the left image. ϕ_r (R_j, R_m) is the angle between the edge which connects the center of R_r to the middle point of R_m and x-axis in the right image. T_{ra_n} is the threshold in five levels. In the first level, T_{ra_n} is set as $\pi/32$. T_{ra_n} increases by $\pi/64$ in each following level.

The total evaluation function of L_i and R_j, written as TEF (L_i, R_j), is the linear combination of all five evaluation functions if several correspondences of edges are already found. If not, it is the linear combination of the first three evaluation functions. The value of TEF (L_i, R_j) is filled in a matrix, named the dynamic programming (DP) matrix. Afterwards, the DP traces back the matrix and determines the local maximum in the matrix. The corresponding L_i and R_j at the local maximum are the matched edges.

$$\mathrm{TEF}(L_i,R_j)=\begin{cases}\left(\mathrm{RAEF}(L_i,R_j)+\mathrm{ROEF}(L_i,R_j)+\mathrm{OEF}(L_i,R_j)+\mathrm{LEF}(L_i,R_j)+\mathrm{EEF}(L_i,R_j)\right)/5,\\ \quad \text{if correspondence was found}\\ \left(\mathrm{RAEF}(L_i,R_j)+\mathrm{ROEF}(L_i,R_j)+\mathrm{OEF}(L_i,R_j)\right)/3, \quad \text{else}\end{cases} \tag{6}$$

The proposed algorithm is tested by the extracted edges from the left and right images of a workpiece as shown in Fig. 9. First of all, all the edges are sorted and labelled according to their middle point position along the x-axis on

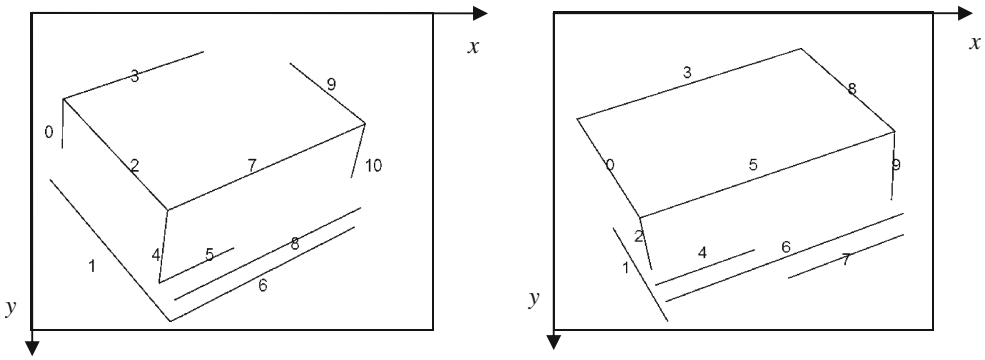


Fig. 9 Extracted edges for correspondence

(a) Edges from left image (b) Edges from right image

the image. In the left image, edges 2, 4, and 7 belong to the first group, in which both end points connect to the other edges. Edges 0, 1, 3, 5, 6, 9, and 10 belong to the second group, in which only one end point connects to the other edges. Edge 8 belongs to the third group, in which both of the end points are open. In the right image, edges 0, 3, 5, and 8 belong to the first group. Edges 2 and 9 belong to the second group. Edges 1, 4, 6, and 7 belong to the third group. Table 1 shows the result of the matching. The unmatched edges include edges 0 and 1 from the left side and edge 1 from the right side. The edge 0 does not have the actual match from the right image, which is correct. The edge 1 from the left side should match with the edge 1 from the right side since both of them represent the same edge of the workpiece. However, since the edge 1 from the right side is only about half as long as the actual, the algorithm considers that it does not correspond to edge 1 from the left side. The design intention of this correspondence method is to find all the correct matches rather than all the matches with some error. Therefore, the designed method ensures a good correspondence.

3.3 Object recognition

Object recognition includes scene-to-model hypothesis verification and pose estimation. These two tasks can be formulated as:

- Given a 3D scene edge set E_s generated by stereovision and a 3D model edge set E_m obtained from the solid model of one part, verify whether or not they match.
- Once the E_s matches with E_m, determine the pose of the solid model of E_m.

3.3.1 Hypothesis verification

All the solid models referred to in this paper including workpiece, jigs, and so on are in the Parasolid file format. A solid model parser is designed in this paper to retrieve only 3D edges from the Parasolid file. For example, E_m is the set of the retrieved 3D edges of a workpiece. E_s is the set of 3D edges generated by stereovision shown in Fig. 9.

Table 1 Edge correspondence result

	Edge number in left	Edge number in right	Total evaluation value	Matched phase	Matched level
Matched edges	7	5	0.98345	1	1
	2	0	0.67891	2	1
	10	9	0.95342	2	2
	9	8	0.80808	2	3
	4	2	0.4360	2	4
	3	3	0.67829	2	5
	8	6	0.76125	3	1
	5	4	0.30718	3	2
	6	7	0.30619	3	2
Unmatched edges	0	N/A	N/A	N/A	N/A
	1	N/A	N/A	N/A	N/A
	N/A	1	N/A	N/A	N/A

The verification process of E_s and E_m includes the following two steps:

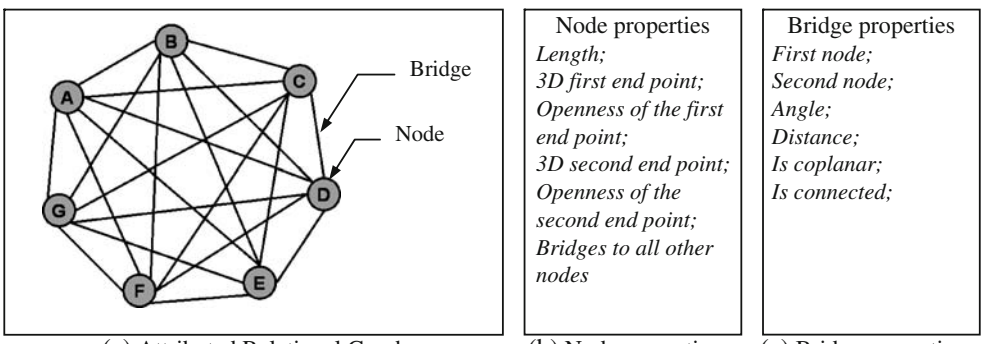
Logical relation building In order to verify the match of E_s and E_m, the edges in each set should be organized together to indicate the edge's individual property as well as its logical relationship with the other edges. In this paper, the attributed relational graph (ARG) is used, which is shown in Fig. 10a. In the graph, each node stands for one individual edge, which contains the properties shown in Fig. 10b. It includes the edge's length, the two end point positions, the openness status of the two end points, and the bridges connecting to all other nodes. Each bridge stands for the relationship between any two edges, which contains the properties shown in Fig. 10c. These properties include the two nodes, the angle between the two edges, the distance of two edges, if applicable—in other words, if the two edges are coplanar and if the two edges are connected to each other.

Verification The flowchart of the algorithm for verifying the match of E_s and E_m is shown in Fig. 11. The inputs for this algorithm are two ARGs for both the scene and the model edges. In the scene nodes pool, the closed edges are processed first, which is similar to the previous section. This algorithm is described as follows:

(a) Pick the first scene node S_1 and a model node M_i ($0<i<$ the total number of model nodes).
(b) If the edge's length in S_1 matches with the edge's length in M_i, the principle matrix shown in Fig. 11 is calculated by aligning the two edges together. These two edges are considered as matched and are pushed into the pool of matched nodes. The node M_i is excluded from the model nodes pool.
(c) Pick the second scene node S_2 and a model node M_j, if the edge's length in S_2 matches with the edge's length in M_j, and if the logical relationship between S_1 and S_2 matches with the logical relationship between M_i and M_j, then the principle matrix is updated by adding the condition of aligning the two edge's middle points together. Afterwards, the M_j is transformed to a new position by this principle matrix. If the new position of M_j matches with the position of S_2, they are considered as matching and are pushed into the pool of matched nodes. The node M_j is excluded from the model nodes pool. As long as there are at least two pairs of matches in the pool of matched nodes, the principle matrix does not need to be updated anymore.
(d) Pick the third scene node S_3 and a model node M_k, if their lengths match, and their logical properties and positions when transformed by the principle matrix are all valid, they are considered as a matched pair. If there is no matched node in the model pool for the scene node S_3, that means that the previous match could be wrong. Therefore, the previously matched pair nodes are popped from the pool of the matched nodes, and then for S_2, the match search is re-conducted in the model node pool but not M_j since it leads the match search in the wrong direction.
(e) If all the scene nodes find their corresponding matched model nodes, the match of E_s and E_m is verified. Otherwise, it is not. Another E_m of another solid model should be tested.
(f) The validated solid model is the recognized model which E_s stands for.

3.3.2 Pose estimation

A common need in the computer vision field is to compute the 3D rigid body transformation that aligns two sets of digital data together as their best match. In metrology field, two or more sets of scanning data of the different portion of one part usually needs to be merged together, known as the registration problem. Since the corresponding relationship between these data sets is unknown, an iteration method is used, such as the iterative closest points which uses the quaternion method [14]. In the industrial quality control



(a) Attributed Relational Graph (b) Node properties (c) Bridge properties

Fig. 10 Attributed relational graph and properties

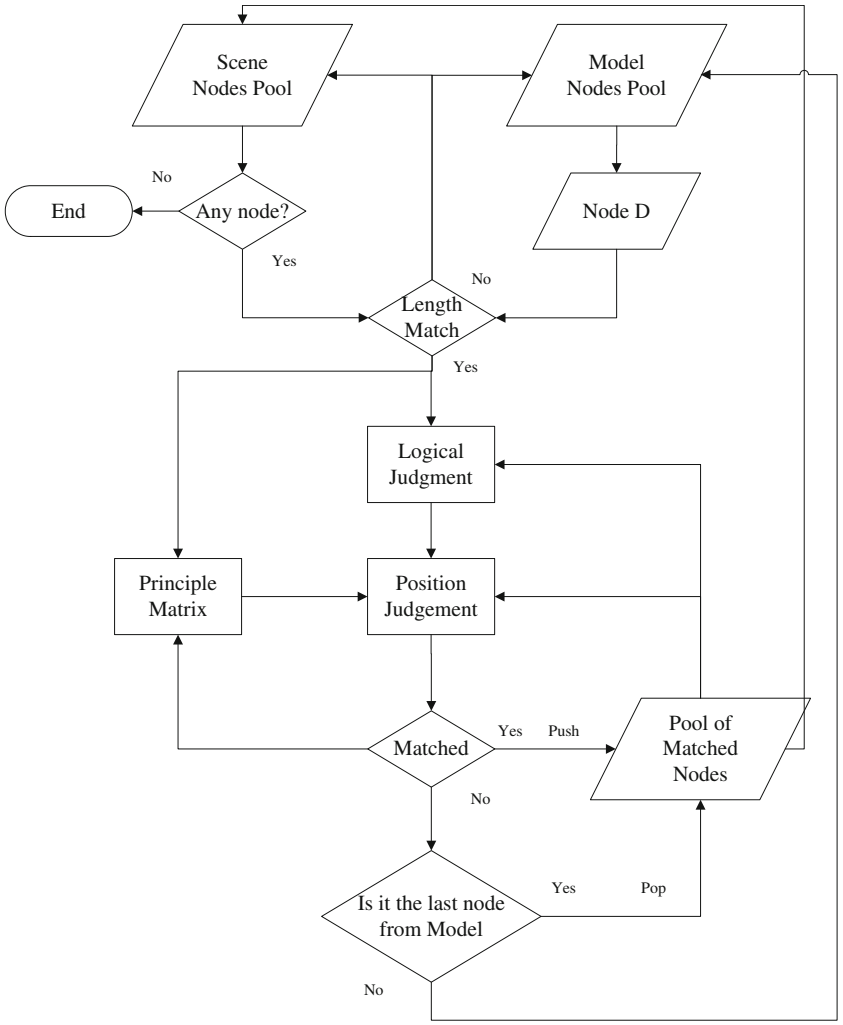


Fig. 11 Flowchart of object recognition

field [15], the scanned digital data of the manufactured part needs to be compared with the design model to evaluate its quality. It involves aligning the scanned data (scattered point cloud) to the pre-designed solid model. Usually, the corresponding relationship between the scanned data and the design model is also unknown. Similar to the solution in metrology field, the iteration method is also used to find the correspondence. In the robotics field, once the object is recognized, the correspondence between the 3D scene data and model data becomes known. Therefore, the translational and rotational components of the transformation can be solved in a closed form without iteration. The most popular method in this field is to align two sets of points [16]. The points could be the corners of a part taken from both the actual object and the solid model. In this paper, using corners to align the solid model to the 3D scene data is not applicable since some of the 3D scene edges may not be the integral edge of the actual object. Therefore, using these corners for alignment will bring significant error to the final position of the solid model. Moreover, from a practical point of view, it is often advantageous to use edges, in which edge detection is more accurate and more reliable than point detection, and edge correspondence is more robust and reliable than point correspondence with respect to noise, partial occlusions, and so on. Therefore, in this paper, the pose of the recognized solid model is estimated based on the edges which have a known correspondence which is built in section 3.3.1. The criterion for the best rotation and translation of the recognized solid model is to minimize the distance of the correspondent edges. The definition of the distance of the correspondent edges in this paper is the distance from the scene edge's corner to the model edge. It can be written as:

$$\begin{aligned}&\text{Minmize}(\text{Dist})\\&= \text{Minmize}\left(\sum_{i=1}^{k}\left(\text{Dist}(\text{StartP}_{E_s}, E_m) + \text{Dist}(\text{EndP}_{E_s}, E_m)\right)\right)\end{aligned} \tag{7}$$

where k is the total number of the correspondent edge pairs, StartP_{E_s} is the start point of the scene edge E_s, and EndP_{E_s} is the end point of the scene edge E_s.

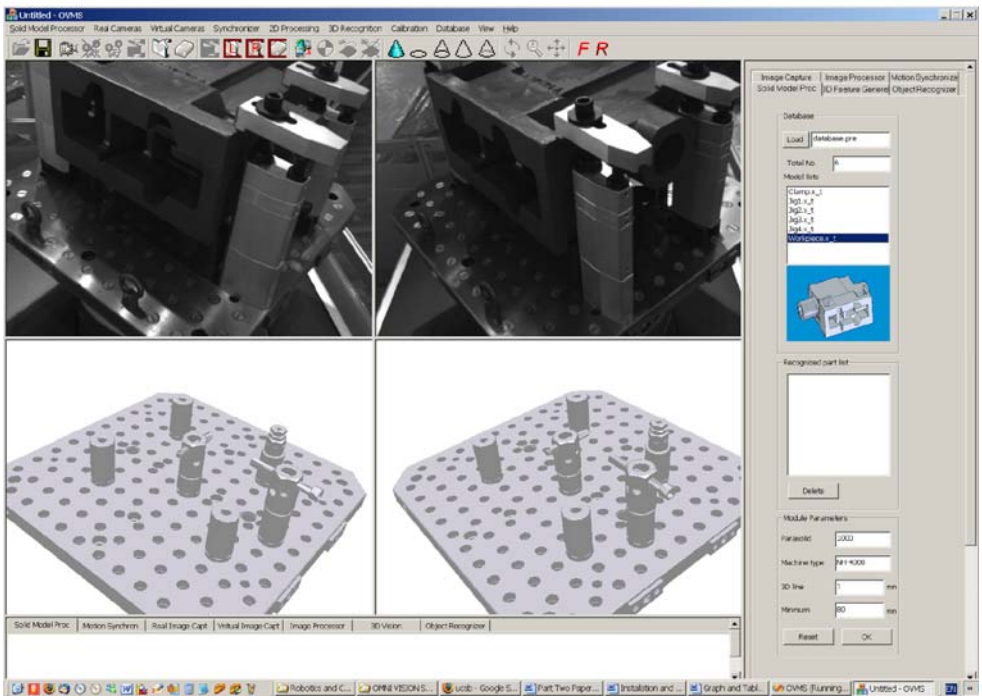
The present algorithm is a closed form optimization using quaternion method. It basically has four steps: self-centered coordinate transform, rotation matrix calculation, translation matrix calculation, and import the solid model into the virtual environment. For details, see references [14] and [17].

The designed pose estimation method in this paper is able to accurately locate the pose of the recognized solid model. As pointed out at the end of section 3.1, some detected edges are only partial edges. These partial edges will affect the pose estimation if their 3D corners are directly used to match with the corners from the solid model. In Eq. 7, the optimization objective is defined to minimize the distance from the corners of the scene edge to the model edge. Even though the scene edge is a partial edge, the pose of the solid model can still be accurately located.

4 Feasibility study

A prototype of this vision system is built, which includes hardware and software. Considering the requirement of collision detection, the accuracy for the inspection is determined as 1 mm, 1,024 × 768 pixels monochrome CCD is the final choice. Limited by the space of machining center, the distance ranges from 0.8–1.2 m. After calculation, the proper focal length is about 8 mm. The magnetic

(a) Setup of the vision system



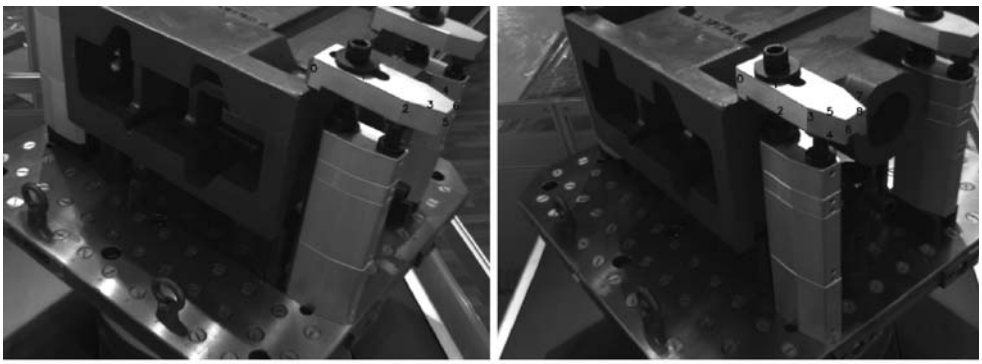
(b) Software of the vision system

Fig. 12 Prototype of the vision system

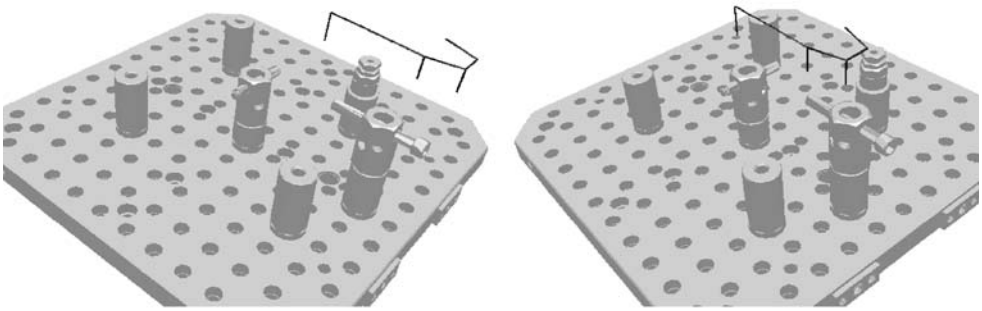
base and swivel joint have been used to increase the portability and adaptability. The final setup of two CCD cameras into a five-axis machining center is shown in Fig. 12a. Figure 12b is a screenshot of the user interface of the designed software. It has the image capturing and processing region, which includes left real image, right real image, left virtual image, and right virtual image. The two real images are aligned at the upper row and the two virtual images are aligned at the lower row. All the solid models of the workpiece and jigs are stored in the database, which is shown at the right portion of Fig. 12b.

As the two real images indicate, there is one complicated machining setup sitting on the rotary setup station of the Mori Seiki NH-4000 machine tool. On the virtual images, there are only a pallet, several supporters, and some locators. The workpiece, jigs, and clamps are missing. In this tested setup, the digital models for one workpiece, one clamp, and four jigs are stored in the database.

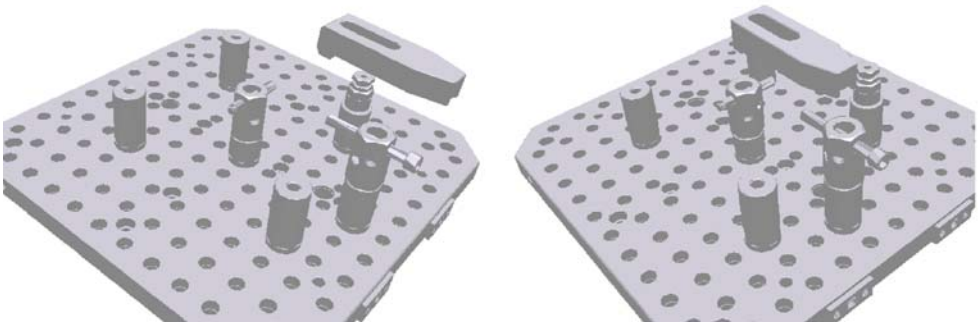
In Fig. 13, the rotary setup station is first located at 340.00° (B-axis) for the image capture. On the captured real images (Fig. 13a, b), the edges of a clamp are extracted with human assistance by just clicking on several corners of the image. There are seven edges on the left and ten edges on the right. After edge correspondence, a total of seven pairs of edges are found. Since the two CCD's positions are known, these seven edges' 3D positions are easily calculated as shown in Fig. 13c and d. The system uses these edges to search in the database and find that the matched model is clamped. After the pose estimation, the solid model of the clamp is imported into the virtual environment as Fig. 13e and f show. A similar operation is done to other clamps and jigs. For the occluded parts, the



(a) Edges of clamp from left side (B=340.00) (b) Edges of clamp from right side (B=340.00)

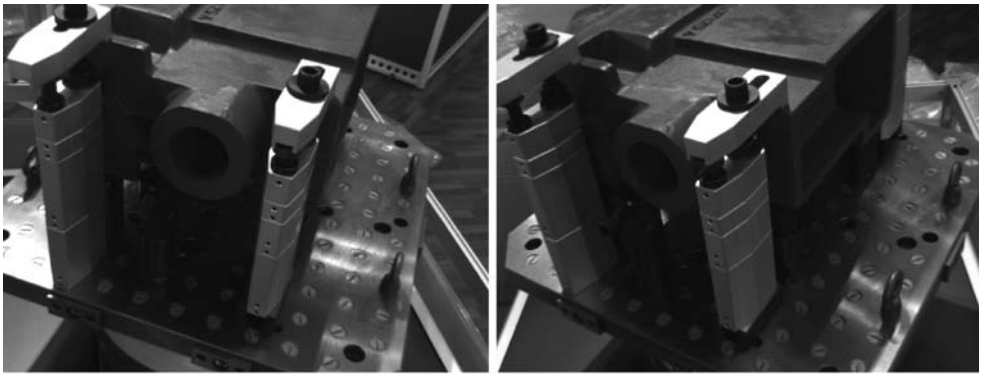


(c) Generated 3D edges in left virtual image (d) Generated 3D edges in right virtual image



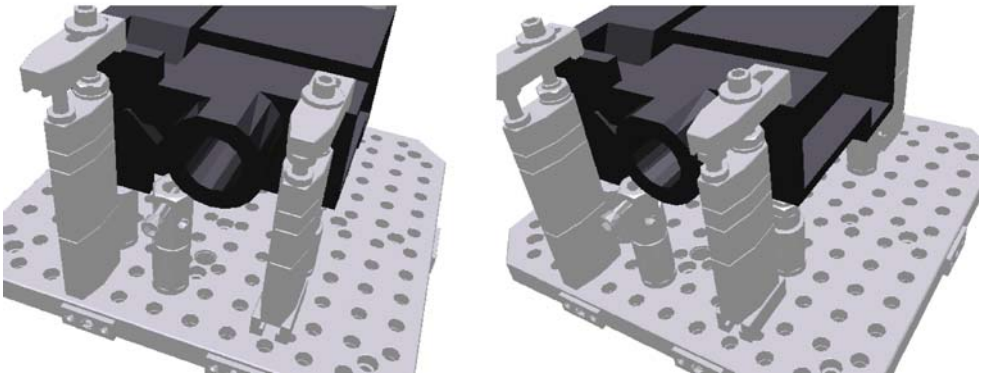
(e) Recognized clamp in left virtual image (f) Recognized clamp in right virtual image

Fig. 13 Recognition and positioning of clamp



(a) Left image of actual setup (B=270.00) (b) Right image of actual setup (B=270.00)

Fig. 14 Recognition of the machining setup

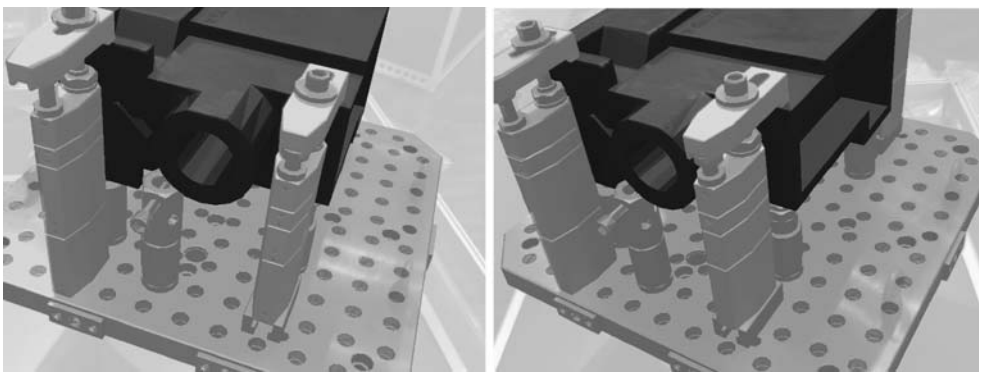


(a) Left image of constructed model (b) Right image of constructed model

setup station rotates to the appropriate view against the two CCD cameras for image capture and object recognition. Figure 14 is the final result of the constructed digital model of the setup when the B-axis of the setup table is located at 270.00°. The total time for constructing this entire setup is about 5 min. After the image compensation, four images in Fig. 14 are superimposed, respectively, for the left and right side. From the superimposed images (Fig. 15), we can evaluate the correctness of the constructed digital model. They indicate the virtual images are exactly overlapped with the real images for the left and right side. The correctness of the position and orientation of the actual setup in accordance with the designed digital model has been confirmed. Using this digital model for the computer numerical control-program simulation, a manual check and trial run are not necessary.

5 Conclusion

The paper describes a 3D vision-based modeling system. It can quickly construct solid models of a machining environment including the workpiece setup with jigs and fixtures on the table or pallet which is as accurate as the



(a) Verification of real and virtual setup from left view (b) Verification of real and virtual setup from right view

Fig. 15 Constructed digital model verification

actual environment. Using this system, the time-consuming manual numerical control-program check and trial run can be completely replaced by simulation so as to significantly increase the productivity and safety of the process. The concept and architecture of the system are introduced first. Rather than scanning all the parts in the machining setup, the object recognition method is introduced so as to quickly construct the model. Afterwards, the key technologies used in the development of the system, such as edge extraction, edge correspondence, and object recognition and positioning have been clarified. In the edge extraction, the composite background image subtraction is proposed by using dynamically changed virtual images. For the edge correspondence, an accurate edge correspondence engine is designed to find the correct matched edges from the left to the right side. In the object recognition, the edge-based pose estimation method is presented so as to increase the positioning accuracy. The prototype of the developed system has successfully been verified through experiments conducted in an actual industrial environment. The feasibility of computer vision technology into the machine tool environment for inspection and modeling applications has been verified in this paper. For future work, more experiments will be done to test this vision system's reliability, accuracy, and usability, especially in an actual machining environment.

Acknowledgments The authors wish to express their sincere appreciation for the generous support from Mori Seiki Corp. which makes this research possible. We also extend the thanks to the IMAO Co. for providing the modular fixture and machining setup.

References

1. Kumar AS, Fuh JYH, Kow TS (2000) An automated design and assembly of interference-free modular fixture setup. Comput Aided Des 32(10):583–596
2. Wu Y, Rong Y, Ma W, LeClair SR (1998) Automated modular fixture planning: geometric analysis. Robot Comput -Integr Manuf 14(1):1–15
3. Brost RC, Goldberg KY (1996) A complete algorithm for designing planar fixtures using modular components. IEEE Trans Robot Autom 12(1):31–46
4. Jain AK, Dorai C (2000) 3D object recognition: representation and matching. Stat Comput 10(3):167–182
5. Pope AR (1994) Model-based object recognition—a survey of recent research. Technical Report, 94-04. University of British Columbia, Vancouver, Canada
6. Arman F, Aggarwal JK (1993) Model-based object recognition in dense-range images—a review. ACM Comput Surv 25(1):5–43
7. Javed O, Shafique K, Shah M (2002) A hierarchical approach to robust background subtraction using color and gradient information. In: Proceedings of the Workshop on motion and video computing, Orlando, FL, 5–6 Dec 2002, pp 22–27
8. Jabri S, Duric Z, Wechsler H, Rosenfeld A (2000) Detection and location of people using adaptive fusion of color and edge information. In: Proceedings of International Conference on Pattern Recognition, Barcelona, Spain, 3–8 Sept 2000, pp 4627–4631
9. Fiala M, Shu C (2005) Background subtraction using self-identifying patterns. In: Proceedings of the 2nd Canadian Conference on Computer and Robot Vision, Victoria, Canada, 9–11 May 2005, pp 558–565
10. Tian X, Zhang X, Yamazaki K, Hansel A (2009) A study on three-dimensional vision system for machining setup verification. Robot Comput -Integr Manuf (in press)
11. Harris C, Stephens M (1988) A combined corner and edge detector. In: Proceedings of the 4th Alvey Vision Conference, Manchester, UK, 31 Aug–2 Sept 1988, pp 147–151
12. Bhat DN, Nayar SK (1998) Ordinal measures for image correspondence. IEEE Trans Pattern Anal Mach Intell 20 (4):415–423
13. Faugeras O et al (1993) Real time correlation-based stereo: algorithm, implementations and applications. INRIA, Technical Report 2013
14. Tian X, Zhou X, Ruan X (2002) Object modeling of multiple views using dual quaternion in reverse engineering. Int J Adv Manuf Tech 20(7):495–502
15. Boukebbab S, Bouchenitfa H, Boughouas H, Linares JM (2007) Applied iterative closest point algorithm to automated inspection of gear box tooth. Comput Ind Eng 52(1):162–173
16. Eggert DW, Lorusso A, Fisher RB (1997) Estimating 3-D rigid body transformations: a comparison of four major algorithms. Machine Vision and Applications 9(5–6):272–290
17. Horn BK (1987) Closed-form solution of absolute orientation using unit quaternions. J Opt Soc Am 4(4):629–642