Research Article

3D Reconstruction of End-Effector in Autonomous Positioning Process Using Depth Imaging Device

Yanzhu Hu and Leiyuan Li

Beijing University of Posts and Telecommunications, College of Automation, Beijing 100876, China

Correspondence should be addressed to Leiyuan Li; lileiyuan1985@163.com

Received 18 May 2016; Revised 9 July 2016; Accepted 14 July 2016

Academic Editor: Jinyang Liang

Copyright © 2016 Y. Hu and L. Li. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The real-time calculation of positioning error, error correction, and state analysis has always been a difficult challenge in the process of manipulator autonomous positioning. In order to solve this problem, a simple depth imaging equipment (Kinect) is used and Kalman filtering method based on three-frame subtraction to capture the end-effector motion is proposed in this paper. Moreover, backpropagation (BP) neural network is adopted to recognize the target. At the same time, batch point cloud model is proposed in accordance with depth video stream to calculate the space coordinates of the end-effector and the target. Then, a 3D surface is fitted by using the radial basis function (RBF) and the morphology. The experiments have demonstrated that the end-effector positioning error can be corrected in a short time. The prediction accuracies of both position and velocity have reached 99% and recognition rate of 99.8% has been achieved for cylindrical object. Furthermore, the gradual convergence of the end-effector center (EEC) to the target center (TC) shows that the autonomous positioning is successful. Simultaneously, 3D reconstruction is also completed to analyze the positioning state. Hence, the proposed algorithm in this paper is competent for autonomous positioning of manipulator. The algorithm effectiveness is also validated by 3D reconstruction. The computational ability is increased and system efficiency is greatly improved.

1. Introduction

In computer vision, 3D reconstruction refers to the process of 3D scene restoration based on a single view or multiple-view images. There are four kinds of reconstruction methods: binocular stereovision, sequence image, photometric stereo, and motion view analysis. Binocular stereo method is suitable for larger objects. Sequence image method is suitable for small objects. Photometric stereo and motion view analysis methods are suitable for large and complex scene reconstruction. Because single video information is incomplete, its reconstruction needs to use the empirical knowledge. The multiview reconstruction is relatively easy. That is to calculate the pose relation between image coordinate frame and the world frame. Then, 3D information is reconstructed by using plurality of 2D images. But computational complexity is high, and the cost is more expensive. For 3D reconstruction using the simple depth image apparatus, a few scholars begin to study and have achieved certain results. Izadi et al. [1] propose a new 3D reconstruction technology based on Microsoft

Kinect. However, due to the limitation of TOF technology, accuracy of the surface texture information is not high. The significance of 3D reconstruction is to make real-time monitoring for end-effector positioning process and error. At the same time, it can lay the foundation for debugging and analysis of manipulator autonomy job task. Ma et al. [2] propose a gradual human reconstruction method based on individual Kinect. Body feature points are positioned in the depth video frames combining with feature point detection and error correction processing algorithm. And human body model is obtained by estimating the body size. Guo and Gao [3] propose a robust automatic UAV image reconstruction method under batch framework. Li et al. [4] seek multiview reconstruction method from the perspective of motion visual analysis. The sparse point cloud and initial mesh are built by each view bias model. Lü et al. [5] propose a Bayesian network model that describes body joints spatial relationship and dynamic characteristics. Golf swing process 3D reconstruction system is built by the similarities of swing movements. The problem of limb occlusion is effectively solved using

easy depth imaging device to capture the motions. Lin et al. [6] utilize adaptive window stereomatching reconstruction method based on integral gray variance and integral gradient variance. Image texture quality is determined according to integral variance size. Related calculation will be done if it is more than a preset variance threshold. And it needs to traverse the whole image to obtain dense disparity map. Izadi et al. [7] get point cloud data using a single mobile Kinect and four fixed ones. The point cloud alignment and fitting problems are also solved by iterative closest points. Kahl and Hartley [8] convert 3D reconstruction into norm minimization problem. A closure approximate solution is derived by second-order cone programming (SOCP). In the case of the known camera rotating, shifting and spacing position of the camera can be solved simultaneously. In this paper, the reconstruction process by capturing end-effector motion and recognizing the target object needs to be understood. The model is 3D point cloud fitting.

(1) Paper's whole engineering problem: the visual system is used to guide the positioning control of manipulator. Each joint trajectory of manipulator is corrected continuously according to the positioning error information. ① Internal and external parameters of Kinect are obtained by camera calibration. They provide the known conditions for solving the 3D model. ② Establishment of the kinematics model can guide the motion control of manipulator. ③ The movement of end-effector is detected and tracked using Kalman filtering by RGB images. And the motion state is estimated (including the end-effector's position and velocity). ④ The object to be positioned and TC's position are determined by target object recognition. ⑤ The manipulator's motion is corrected until the positioning is successful by the error information between EEC and TC. ⑥ Effectiveness of the algorithm can be verified by 3D reconstruction in the positioning process. Simultaneously, it also provides the convenience for visualization monitoring.

(2) Paper's research intention: firstly, we hope to improve autonomous operability of the manipulator and the adaptability to environment. Secondly, we hope that the system has the capability of visualization monitoring in the positioning process. And the positioning error can be calculated and analyzed in real time.

2. Camera Calibration

2.1. Kinect Hardware. A simple depth imaging device Kinect is used as a sensor which consists of RGB camera, IR camera, rotating motor, and speech array. IR camera consists of infrared transmitter and infrared receiver as shown in Figures 1(a) and 1(b). RGB camera outputs color images. IR camera outputs depth images directly as shown in Figures 1(c) and 1(d). Figure 1(c) denotes the depth point cloud information. Figure 1(d) denotes the mapping between depth point cloud and the distance. Figure 1(e) denotes the real-time depth value corresponding to point (320, 240). RGB camera includes three-frame rate modes: 1280 * 960 @ 12 fps, 640 * 480 @ 30 fps, 640 * 480 @ 15 fps. Frame rate mode of IR camera is 640 * 480 @ 30 fps. So far, most studies are about human detection and pose estimation based on Kinect images. New

research is about human behavior recognition. Hassine et al. [9] use Kinect to detect and identify the target object and calculate the target position in real time. Akshara et al. [10] propose a new autonomous positioning method of manipulator based on machine learning and Kinect camera. Autonomous learning algorithm (target feature set is trained) based on Kinect is studied by Cornell University [11]. In this paper, RGB images are used for object recognition, segmentation, motion capture, and state estimation. Depth images are used to calculate the target's 3D information (including EEC, TC, and 3D surface fitting).

2.2. Kinect Calibration. Internal parameters describe the transform relation between camera coordinate frame and image coordinate frame. External parameters describe the transform relation between camera coordinate frame and world frame. Thence, camera calibration is the premise of 3D reconstruction. It is convenient to use OpenCV, but the calibration results are often inaccurate and unstable. Paper uses Matlab toolbox for calibration. Then, the calibration results are applied to stereomatching and parallax calculation. The radial distortion coefficients are estimated by least squares method in Zhang's calibration [12, 13]. And internal and external parameters are estimated by a closed-form solution. External parameters of Kinect are formed by T_d and T_c . T_d is composed of rotation and translation matrix from calibration plate to the camera coordinate frame in IR camera. T_c is composed of rotation and translation matrix in RGB camera. Herrera method [14] is used for Kinect calibration. Firstly, Zhang's method is used for parameters initialization. Nonlinear minimization of cost function is used to estimate internal and external parameters by using Levenberg-Marquardt method. Perspective projection transformation describes the mapping from space point $\mathbf{X} = [X, Y, Z]^T$ to 2D point $\mathbf{x} = [x, y]^T$. It is denoted by 3×4 matrix $\mathbf{P}$:

$$\mu \tilde{\mathbf{x}} = \mathbf{P} \tilde{\mathbf{X}} = \mathbf{K} [\mathbf{R}, \mathbf{t}] \tilde{\mathbf{X}}, \quad (1)$$

where μ denotes nonzero scale factor. $\tilde{\mathbf{x}}$ denotes homogeneous pixel coordinate. $\mathbf{R}$ denotes 3×3 rotation matrix. $\mathbf{t}$ denotes 3×1 translation vector. $\mathbf{R}$ and $\mathbf{t}$ describe the camera direction and position in world frame, respectively. $\mathbf{K}$ denotes internal parameter matrix $\mathbf{K} = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}$. α and β denote conversion factors between imaging plane pixel and space physics length. In RGB image, one pixel represents $5.48 \mu\text{m}$. So, focal length is $f = \alpha * 5.48 = \beta * 5.48$. In depth image, one pixel represents $10.34 \mu\text{m}$. So, focal length is $f = \alpha * 10.34 = \beta * 10.34$. γ denotes a skew factor; here, $\gamma = 0$. (u_0, v_0) is origin coordinate between axis and the image plane. The distortion coefficients are not included in (1). But they can be obtained by Herrera method [14], where k_1 and k_2 denote the radial distortion coefficients. k_3 and k_4 denote the tangential distortion coefficients.

The length and width of calibration plate are 575 mm and 350 mm, respectively. Simulink is used to read Kinect depth video [15–17]. Twenty videos are obtained from different perspectives. And each video records the calibration plate within 2~3 seconds. Each video is transformed into images

FIGURE 1: Kinect hardware.

frame by frame, in which there are 20 images clustered from different perspectives as shown in Figure 2. The calibration can be divided into two steps: initialization and nonlinear optimization. Optimization is an accurate calculation process by iterative gradient dropping of Jacobian matrix. Color camera calibration results are shown in Table 1, its rotation matrix $\mathbf{R} = \begin{bmatrix} -0.3302 & 0.9425 & -0.0505 \\ 0.5463 & 0.1472 & -0.8244 \\ -0.7696 & -0.2999 & -0.5636 \end{bmatrix}$ and translation vector $\mathbf{t} = [-390.1895 \text{ mm} \ 271.2135 \text{ mm} \ 1392.7619 \text{ mm}]$. Depth camera calibration results are shown in Table 2, its rotation matrix $\mathbf{R} = \begin{bmatrix} -0.3302 & 0.9425 & -0.0505 \\ 0.5463 & 0.1472 & -0.8244 \\ -0.7696 & -0.2999 & -0.5636 \end{bmatrix}$ and translation vector $\mathbf{t} = [-375.1875 \text{ mm} \ 270.4205 \text{ mm} \ 1394.5524 \text{ mm}]$.

3. Kinematics Modeling

Manipulator motion control is guided by forward and inverse kinematics model. The system obtains the position of the target by Kinect. Then, the inverse kinematics model is used to get the rotation angles of each joint. Secondly, the corresponding command is sent to control the movement

TABLE 1: Internal parameters of color camera.

Focal length [pixel]			Origin [pixel]	
α	β	f [mm]	u_0	v_0
532.02	531.39	2.92	316.71	238.44
Distortion coefficients			Pixel error [pixel]	
k_1	k_2	k_3	e_x	e_y
0.2447	-0.5744	0.0029	0.630	0.882

TABLE 2: Internal parameters of depth camera.

Focal length [pixel]			Origin [pixel]	
α	β	f [mm]	u_0	v_0
587.944	589.418	6.1	313.222	259.304
Distortion coefficients			Pixel error [pixel]	
k_1	k_2	k_3	e_x	e_y
-0.307	0.729	0.048	0.630	0.882

of each joint. Thus, the end-effector can reach the target position.

FIGURE 2: Calibration plates.

FIGURE 3: Manipulator diagram.

This paper is illustrated by the example of 5-degree-of-freedom (DOF) manipulator. There are five rotation axes, six joints, and three links as shown in Figure 3. The rotation axes are waist rotation axis, arm pitching axis, forearm pitching axis, wrist pitching axis, and wrist rotation axis, respectively. Forward kinematics model is established. The end-effector motion trajectory is derived by each joint angle. Define

$${}^0T_N = {}^0T_N(\mathbf{v}) = {}^0T_1 {}^1T_2 \cdots {}^{N-1}T_N = \prod_{i=1}^N {}^{i-1}T_i$$

$$= \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

where

$$\begin{aligned} {}^{i-1}T_i &= \text{Trans}(x, a) \text{Trans}(z, d) \text{Rot}(x, \alpha) \text{Rot}(z, \theta) \\ &= \begin{bmatrix} c\theta & -s\theta & 0 & a \\ c\alpha s\theta & c\alpha c\theta & -s\alpha & 0 \\ s\alpha s\theta & s\alpha c\theta & c\alpha & d \\ 0 & 0 & 0 & 1 \end{bmatrix}. \end{aligned} \quad (3)$$

(2)

It is a homogeneous transformation matrix from $i - 1$ to i coordinate frame. N denotes the number of joints. $\mathbf{v} = [v_1^T, v_2^T, \dots, v_N^T]^T$ denotes the link parameter vector. Sine and cosine functions are abbreviated as s and c . Inverse kinematics model is established as shown in [18].

4. Manipulator End Motion Capture

The proper detection of the end-effector motion is important for postprocessing. In this paper, the changing pixel regions are detected from image sequences. And the moving object is extracted from a static background using three-frame differencing method. Consecutive three-frame differencing method [19] can better deal with environmental noise, such as weather, light, shadow, and messy background interference. It is better than two-adjacent differencing method [20] in double-shadow treatment. Then, morphology erosion and dilation are operated for the binary image to remove holes.

The motion state is estimated after moving end-effector was extracted. So, it is convenient to calculate the position error between end-effector and target object. The end-effector movement is a discrete-time dynamic system in video. State vector is $\mathbf{x}_k = [x(k), y(k), x'(k), y'(k)]^T$, where $x(k)$ and $y(k)$ denote the end-effector center coordinates on x - and y -axes, respectively. $x'(k)$ and $y'(k)$ denote the velocities on x - and y -axes, respectively. Assume that observation vector is $\mathbf{z}_k = [x_c(k), y_c(k)]^T$, where $x_c(k)$ and $y_c(k)$ denote the observations on x - and y -axes, respectively. The system status is predicted and tracked by using Kalman filtering [21]. State equations can be expressed as follows:

$$\begin{aligned}\mathbf{x}_k &= \Phi_k \mathbf{x}_{k-1} + \Gamma_k \mathbf{u}_{k-1} + \mathbf{b}_{k-1}, \\ \mathbf{z}_k &= \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k,\end{aligned}\quad (4)$$

where t_k denotes the iteration time. $\mathbf{x}_k$ denotes system status from t_{k-1} to t_k . $\mathbf{u}_{k-1}$ denotes system control variable. In paper, assume that the system is free from outside influence, so here $\mathbf{u} = \mathbf{0}$. $\mathbf{z}_k$ denotes measurement value from t_{k-1} to t_k . $\mathbf{b}_{k-1}$ and $\mathbf{v}_k$ denote process noise and measurement noise, respectively. They are assumed to be white Gaussian noise (WGN). Φ_k denotes the state transition matrix of system from t_{k-1} to t_k . Since the end-effector's movement is approximated as uniform motion, the state transition matrix is $\Phi_k = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$.

Γ_k denotes transformation matrix of control coefficients. $\mathbf{H}_k$ denotes measurement transition matrix, also called observation matrix; here $\mathbf{H}_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$. In other words, the state vector can be observed directly. Prediction is a process of estimating the state of next moment according to the current state and the error covariance. Thereby, a priori estimate is obtained. Correction is a process of feedback. The new actual observation value and a priori estimate are considered together. Thereby, a posteriori estimate is obtained. When the system is represented by (4), it is possible to estimate the posterior probability density function of the mean and covariance. State prediction equation is

$$\mathbf{x}_k^- = \Phi_k \mathbf{x}_{k-1}^+ + \Gamma_k \mathbf{u}_{k-1}. \quad (5)$$

Covariance prediction equation is

$$\mathbf{P}_k^- = \Phi_k \mathbf{P}_{k-1}^+ \Phi_k^T + \mathbf{Q}_{k-1}. \quad (6)$$

Kalman gain matrix is

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T [\mathbf{H}_k \mathbf{P}_k^- + \mathbf{H}_k^T + \mathbf{R}_k]^{-1}. \quad (7)$$

Covariance estimation is

$$\mathbf{P}_k = [\mathbf{I} - \mathbf{K}_k \mathbf{H}_k]^{-1} \mathbf{P}_k^-. \quad (8)$$

State estimation:

$$\mathbf{x}_k^+ = \mathbf{x}_k^- + \mathbf{K}_k (\mathbf{z}_k - \mathbf{H}_k \mathbf{x}_k^-), \quad (9)$$

where $\mathbf{x}_k^-$ denotes the state prediction at t_k . $\mathbf{x}_{k-1}^+$ denotes the state measurement at t_{k-1} . $\mathbf{x}_k^+$ denotes the state measurement at t_k . $\mathbf{P}_k^-$ denotes prediction covariance matrix at t_k . $\mathbf{P}_k$ denotes measurement covariance matrix at t_k . $\mathbf{P}_{k-1}^+$ denotes measurement covariance matrix at t_{k-1} . $\mathbf{Q}_{k-1}$ denotes covariance matrix of process noise. $\mathbf{R}_k$ denotes covariance matrix of measurement noise. $\mathbf{K}_k$ denotes Kalman gain matrix. After the completion of each prediction and correction, a priori estimate will be predicted by posterior estimate at the next time. And repeat above steps. This algorithm does not need to save the last measurement data. After the data is updated, the new parameters will be estimated according to recurrence formulas. Thus, storage and computation of filtering device are greatly reduced. And system operational efficiency is improved.

5. Object Recognition

5.1. Image Preprocessing. This section describes the preprocessing based on Kinect RGB images. Target recognition is illustrated by the example of cylindrical target object (CTO). End-effector and CTO will appear in the same video. Firstly, image gray processing is carried out. It is the process which the color images are converted into gray images. And this process can reduce the computation greatly. Gray image gradation is 0~255. Grayscale method, gray = 0.114B + 0.587G + 0.299R [22], was used in paper.

Secondly, image median filtering is carried out. It cannot blur the edges while suppressing random noise [23]. It is a kind of nonlinear smoothing method. Gray values of the pixels are sorted in the sliding window. The original gray value of the pixel in the window center is substituted by the median.

Thirdly, mathematical morphology operation is carried out. Dilation and erosion operations are used [24]. It is widely used in edge detection, image segmentation, and image thinning as well as noise filtering and so forth. Assume that $\mathbf{E}(x, y)$ denotes binary image. $\mathbf{B}(s, t)$ denotes structural elements. The following operations are used:

(1) Morphology dilation:

$$\mathbf{X} = \mathbf{E} \oplus \mathbf{B} = \{(s, t) : \mathbf{B}(s, t) \cap \mathbf{E} \neq \emptyset\}. \quad (10)$$

(2) Morphology erosion:

$$\mathbf{Y} = \mathbf{E} \odot \mathbf{B} = \{(s, t) : \mathbf{B}(s, t) \subset \mathbf{E}\}. \quad (11)$$

Then, make a weighted fusion between an input image and its "canny" operator detection. Threshold segmentation of fusion image is carried out. Image segmentation is the basis for determining the feature parameters. The whole contour of the object is obtained after image segmentation. An example

FIGURE 4: The calculation example of feature parameters.

is demonstrated in Figure 4. Shaded area represents the boundary.

At last, autonomous positioning algorithm should make the system possess the capability of automatically extracting geometric features. These features should keep invariant when the image is transformed, such as translation, rotation, twisting, and scaling.

There are two kinds of CTO feature parameters, edge contour feature and shape parameters. The parameter of contour points belongs to edge contour feature. Shape parameters include perimeter, area, longest-axis, azimuth, boundary matrix, and shape coefficient.

① Contour points represent the required number of pixels which can outline the contour. The number of contour points is 22 in Figure 4.

② Perimeter represents the contour length of outer boundary. And it can be calculated by the sum of the distance between two adjacent pixels on outer boundary. In the boundary, assume that the distance between two adjacent edge pixels is 1 in horizontal or vertical direction. The distance between two adjacent edge pixels is $\sqrt{2}$ in oblique direction. So, perimeter is $14 + 8\sqrt{2}$ in Figure 4.

③ Area can be represented with the number of pixels in target region. So, area is 41 in Figure 4.

④ The longest-axis denotes the maximum extension length of target region, that is, the connection line of the maximum distance between two pixel points on outer boundary. So, the longest-axis is 8 in Figure 4.

⑤ Azimuth represents the angle between the longest-axis and x -axis in target region. So, azimuth is 0 in Figure 4.

⑥ Boundary matrix denotes the minimum matrix encompassing the target region. And it is also the intuitive expression of flat level of target region. It is composed of four outer boundary tangents. Two of them are parallel to the longest-axis, and the other two are perpendicular to

the longest-axis. So, boundary matrix is
$$\begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$
 in

Figure 4.

⑦ Shape coefficients denote the ratio of the area to square of the perimeter. So, shape coefficient is 0.0639 in Figure 4.

5.2. Neural Network Recognition. This section describes the recognition based on Kinect RGB images. BP neural network learning algorithm is used in this paper. It is a learning

process of error back propagation algorithm. This network can learn and store a large amount of input-output mappings. And it is also the mathematical equation without describing these mappings in advance. Its learning rule is the gradient descent method. Mean square error is minimized by adjusting the network weights and thresholds continuously. Network topology includes input layer, hidden layer, and output layer.

Feature vectors are extracted as training sample. Neural network is used as a classifier instead of Euclidean distance method to implement target recognition.

The design of input and output layers in neural network is as follows: the number of nodes is 7 in input layer. The elements of the input vector are contour points, perimeter, area, longest-axis, azimuth, boundary matrix, and shape coefficients, respectively. The number of nodes is 3 in output layer. The elements of output vector are cylinder, square, and spherical, respectively. Normalized values of output are 0.1, 0.2, and 1, respectively. The design of hidden layer is as follows: there are two hidden layers which include a logarithmic characteristic function and a "purelin" function. The number of nodes is 20 in the first hidden layer. The number of nodes is 3 in the second hidden layer. Linear excitation function is used in output layer. The number of hidden layers is related to the number of neurons and specific issues. At present, it is difficult to give an accurate function to describe this relation. Experiments show that it is not sure to improve the accuracy of the network when increasing the number of hidden layers and neurons. The initial number of the hidden layers can be selected via $n' = \sqrt{m+n} + a$. Wherein, " m " denotes the number of neurons in input layer. " n " denotes the number of neurons in output layer. " a " denotes an integer from 1 to 10. Here, " n " is set to 15.

Sample Set. Sample set is collected from the shooting scene of Kinect. In Figure 5, there are cylindrical objects, square objects, and spherical objects. There are 30 kinds of cylindrical objects (Figure 5(a)), 10 kinds of square objects (Figure 5(b)), and 10 kinds of spherical objects (Figure 5(c)). For each target object, there are 20 different viewing angles (schematic diagram in Figures 5(d), 5(e), and 5(f)). So, the numbers of cylindrical objects, square objects, and spherical objects are 600, 200, and 200, respectively, in sample set. In Figure 5(g), edge contour is extracted.

Network Training. The weights of neurons are adjusted in the process of training network. The training stops until the mean square error (mse) reaches 10^{-7} . The maximum number of iterations is set to 10000. The momentum constant is set to 0.8. The initial learning rate is 0.01. Increasing ratio of learning rate is 1.05. Reduction ratio of learning rate is 0.7. The dimension of training set is 1000×8 (cylindrical 600, square 200, and spherical 200). The dimensions of validation set and testing set are all 100×7 (cylindrical 60, square 20, and spherical 20). Sample set (including training set, validation set, and testing set) is the normalized data. And the range is $[0, 1]$. Normalization function is $f(x) = 1/(1 + e^{-x})$.

After the identification, we need to calculate the target centroid (TC). The shape of target object is regular, so

(a) 30 kinds of cylindrical objects. Note: from left to right, the first-tea box, the first cup, the second cup, the second-tea box, the third cup, the first-pill bottle, glue bottle, tape measure, marker, herborist bottle, the second-pill bottle, the third-tea box, dishwashing liquid bottle, sealant bottle, the fourth cup, the first pop-top, AA battery, the first-reagent bottle, the second-reagent bottle, the third-reagent bottle, the fourth-reagent bottle, the fifth-reagent bottle, the second pop-top, lubricant oil box, the sixth-reagent bottle, screen cleaner bottle, xylitol cans, sauce bottle, the third pop-top, and glasses cleaner plastic bottle

(b) 10 kinds of square objects. Note: from left to right, square tea box, slide caliper box, rhinitis ning granule box, iPhone 6S plus box, drink box, book needle box, toothpaste box, square box, seed wine packaging, and fingerprint box

(c) 10 kinds of spherical objects. Note: from left to right, basketball, walnut, ping-pong ball, shuttlecock (to remove feathers), kumquat, longan, orange, pear, sweet dumplings, and *Citrus junos*

(d) RGB images and binary segmentation of cylindrical objects from different perspectives

(e) RGB images and binary segmentation of square objects from different perspectives

(f) RGB images and binary segmentation of spherical objects from different perspectives

(g) Edge contour extraction

FIGURE 5: Images preprocessing.

the spatial position of TC is the destination of end-effector positioning. TC is calculated as follows:

$$\begin{aligned}\bar{x} &= \frac{\sum_{i_b}^{i_e} \sum_{j_b}^{j_e} i (p_{ij} - T)}{\sum_{i_b}^{i_e} \sum_{j_b}^{j_e} (p_{ij} - T)}, \\ \bar{y} &= \frac{\sum_{i_b}^{i_e} \sum_{j_b}^{j_e} j (p_{ij} - T)}{\sum_{i_b}^{i_e} \sum_{j_b}^{j_e} (p_{ij} - T)},\end{aligned}\quad (12)$$

where i_b and i_e denote minimum pixel and maximum pixel of target object, respectively, along the row direction. j_b and j_e denote the minimum pixel and maximum pixel of target

object, respectively, along the column direction. “ T ” denotes an adaptive threshold. p_{ij} denotes the grayscale value.

6. 3D Reconstruction

This section describes 3D surface fitting based on Kinect depth images. Vision system captures the end-effector motion. In other words, moving objects can be recognized and tracked. At the same time, the motion state is estimated. Thence, 3D reconstruction is carried out in the positioning process. There is not a full synchronization and alignment between color-frame-stream and depth-frame-stream, so it

FIGURE 6: 3D mapping model from 2D.

is necessary to correct these two data streams. For 3D reconstruction of the positioning process, the system requires faster processing speed and the capacity of a large amount of data processing. So, we use the batch mode of sequence images. According to depth imaging principle [25–27], the extraction model of 3D point cloud is established as shown in Figure 6. $O_w X_w Y_w Z_w$ denotes the world frame. $O_{IR} X_{IR} Y_{IR} Z_{IR}$ denotes IR camera coordinate frame. XOY denotes image physical coordinate frame. uov denotes image pixel coordinate frame. In 3D space, there is point P whose coordinate is $P(x_w, y_w, z_w)$. Camera coordinate frame is established on the optical center of IR camera. $P(x_{IR}, y_{IR}, z_{IR})$ denotes the coordinate in camera coordinate frame. $P(x_u, y_u)$ and $P(x_d, y_d)$ denote ideal coordinate and actual coordinate, respectively, in image physical coordinate frame. (u', v') and (u, v) denote ideal coordinate and actual coordinate, respectively, in image pixel coordinate frame. The formulas are as follows:

$$\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} x_u \\ y_u \\ 1 \end{bmatrix}, \quad (13)$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix}.$$

Ideal coordinate denotes the coordinate without distortion. Actual coordinate denotes the coordinate with radial distortion or tangential distortion. C_{IR} , C_{RGB} , and $C_{projector}$

represent the optical-axis centers of IR camera, RGB camera, and infrared transmitter, respectively.

For Brown distortion model [28], make the world coordinate frame and camera coordinate frame coincide. So, rotation matrix $\mathbf{R}$ is equal to $\mathbf{I}$ ($\mathbf{R} = \mathbf{I}$). And translation matrix $\mathbf{C}$ is equal to 0 ($\mathbf{C} = 0$). Spatial coordinate of the target point can be expressed as follows:

$$\begin{bmatrix} x_{IR} \\ y_{IR} \\ z_{IR} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{C} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}, \quad (14)$$

$$\begin{aligned} \mathbf{X} = \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} &= \begin{bmatrix} \mathbf{R} & \mathbf{C} \\ 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} x_{IR} \\ y_{IR} \\ z_{IR} \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{I} & 0 \\ 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} x_{IR} \\ y_{IR} \\ z_{IR} \\ 1 \end{bmatrix} = \begin{bmatrix} x_{IR} \\ y_{IR} \\ z_{IR} \\ 1 \end{bmatrix}. \end{aligned} \quad (15)$$

From pinhole model of the camera, we can get

$$\frac{x_{IR}}{x_d} = \frac{y_{IR}}{y_d} = \frac{z_{IR}}{f}. \quad (16)$$

From formula (16), the following can be obtained:

$$\frac{x_{IR}}{f x_u} = \frac{y_{IR}}{f y_u} = \frac{z_{IR}}{f}. \quad (17)$$

It can be deduced that

$$z_{IR} \begin{bmatrix} x_u \\ y_u \\ 1 \end{bmatrix} = \begin{bmatrix} x_{IR} \\ y_{IR} \\ z_{IR} \end{bmatrix} = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix}, \quad (18)$$

$$\begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix} = \begin{bmatrix} 1 + k_1 r^2 + k_2 r^4 \\ 2k_3 x_u y_u + k_4 (r^2 + 2x_u^2) \\ 2k_4 x_u y_u + k_3 (r^2 + 2y_u^2) \\ 1 \end{bmatrix}, \quad (19)$$

$$r^2 = x_u^2 + y_u^2, \quad (20)$$

where k_1 and k_2 denote radial distortion coefficients. k_3 and k_4 denote tangential distortion coefficients. z_{IR} denotes the depth value of IR camera. z_{IR} can be extracted from depth image directly. So, (x_u, y_u) can be solved by formulas (13), (19), and (20). Then, we can get (x_w, y_w, z_w) when (x_u, y_u) is substituted into formula (18). But formula (19) is a bivariate

quartic equation set, so (x_u, y_u) is difficult to solve. In the case of only radial distortion, it can be deduced from formula (19) that

$$\begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix} = (1 + k_1 r^2 + k_2 r^4) \begin{bmatrix} x_u \\ y_u \\ 1 \end{bmatrix}. \quad (21)$$

Let $T = r^2$; it can be deduced from formula (21) that

$$\begin{aligned} x_d^2 &= (1 + k_1 T + k_2 T^2)^2 x_u^2, \\ y_d^2 &= (1 + k_1 T + k_2 T^2)^2 y_u^2. \end{aligned} \quad (22)$$

In formula (22), two equations add

$$x_d^2 + y_d^2 = (1 + k_1 T + k_2 T^2)^2 (x_u^2 + y_u^2). \quad (23)$$

Formula (20) is substituted into formula (23):

$$x_d^2 + y_d^2 = (1 + k_1 T + k_2 T^2)^2 T. \quad (24)$$

(x_d, y_d) can be solved by formula (13). So, formula (24) becomes a quadratic equation and “ T ” is obtained. Then, “ T ” is substituted into formula (21). Then, (x_u, y_u) is solved. At last, 3D coordinate of target point is obtained by substituting (x_u, y_u) into formula (18). Since the distortion of Kinect itself cannot be ignored, we must consider the radial distortion and tangential distortion simultaneously. Therefore, formula (15) becomes [29]

$$\begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} = \begin{bmatrix} \frac{zx_d + zx_d(k_1 r^2 + k_2 r^4) + 2zk_3 x_d y_d + k_4 z(r^2 + 2x_d^2)}{(b_1 r^2 + b_2 x_d + b_3 y_d + b_4)r^2 + 1} \\ \frac{y_d z + y_d z(k_1 r^2 + k_2 r^4) + 2zk_4 x_d y_d + k_3 z(r^2 + 2y_d^2)}{(b_1 r^2 + b_2 x_d + b_3 y_d + b_4)r^2 + 1} \\ z_{\text{IR}} \end{bmatrix}, \quad (25)$$

where b_1, b_2, b_3 , and b_4 denote implicit factors. k_1, k_2, k_3 , and k_4 are obtained by calibration. b_1, b_2, b_3 , and b_4 can be solved by converting into least squares problem. Assume that there are a total of N calibration points. (x_{di}, y_{di}) and (x_{ui}, y_{ui}) denote actual value and ideal value, respectively, in image physical coordinate frame. Three vectors $\mathbf{X}_{Ui}$, $\mathbf{Y}_{Vi}$, and $\mathbf{S}$ are given:

$$\begin{aligned} \mathbf{X}_{Ui} &= [x_{ui} r_i^4, x_{ui} x_{di} r_i^2, x_{ui} y_{di} r_i^2, x_{ui} r_i^2]^T, \\ \mathbf{Y}_{Vi} &= [y_{ui} r_i^4, y_{ui} x_{di} r_i^2, y_{ui} y_{di} r_i^2, y_{ui} r_i^2]^T, \\ \mathbf{S} &= [\mathbf{X}_{U1}, \mathbf{Y}_{V1}, \dots, \mathbf{X}_{Ui}, \mathbf{Y}_{Vi}, \mathbf{X}_{UN}, \mathbf{Y}_{VN}]^T. \end{aligned} \quad (26)$$

Let vectors $\mathbf{p}$ and $\mathbf{e}$ be

$$N = 80,$$

$$r_i^2 = x_{di}^2 + y_{di}^2, \quad (27)$$

$$\mathbf{p} = [b_1, b_2, b_3, b_4]^T,$$

$$\mathbf{e} = [A_1 - x_{u1}, B_1 - y_{u1}, \dots, A_N - x_{uN}, B_N - y_{uN}]^T. \quad (28)$$

In formula (28),

$$\begin{aligned} A_i &= x_{di} + x_{di} (k_1 r_i^2 + k_2 r_i^4) + 2k_3 x_{di} y_{di} \\ &\quad + k_4 (r_i^2 + 2x_{di}^2), \\ B_i &= y_{di} + y_{di} (k_1 r_i^2 + k_2 r_i^4) + 2k_4 x_{di} y_{di} \\ &\quad + k_3 (r_i^2 + 2x_{di}^2). \end{aligned} \quad (29)$$

According to formula (25), vectors $\mathbf{p}$, $\mathbf{e}$, and $\mathbf{S}$ meet $\mathbf{e} = \mathbf{S} \cdot \mathbf{p}$. And we can get $\mathbf{p} = (\mathbf{S}^T \mathbf{S})^{-1} \mathbf{S}^T \mathbf{e}$ by the least squares method. At last, 3D coordinate of target point is obtained by substituting (x_u, y_u) into formula (25).

After point cloud of the target is obtained, surface fitting will be made by using triangular facets [30–32]. This process is also called grid generation. In real space $\mathbf{R}^3$, N scattered points $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ are given. Each point corresponds to constraint $\{h_1, h_2, \dots, h_N\}$. Function $g : \mathbf{R}^3 \rightarrow \mathbf{R}$ is constructed. For each scattered point, it satisfies

$$g(\mathbf{x}_i) = h_i \quad (i = 1, 2, \dots, N). \quad (30)$$

The number of solutions to (30) is infinite. Ideal solution is the minimum of energy function (31) when satisfying the constraint of formula (30):

$$E = \int_{\mathbf{R}^3} [h_{xx}^2(\mathbf{x}) + h_{yy}^2(\mathbf{x}) + h_{zz}^2(\mathbf{x}) + h_{xy}^2(\mathbf{x}) + h_{yz}^2(\mathbf{x}) + h_{zx}^2(\mathbf{x})] d\mathbf{x}. \quad (31)$$

Variational technology is used to solve the minimum problem of energy function. General solution forms

$$g(\mathbf{x}) = \partial(\mathbf{x}) + \sum_{i=1}^n \omega_i \cdot \Phi(\|\mathbf{x} - \mathbf{x}_i\|), \quad (32)$$

where $\Phi : \mathbf{R} \rightarrow \mathbf{R}$ denotes radial basis function (RBF). $\mathbf{x}$ denotes any one point on surface. $\mathbf{x}_i$ denotes a scattered point and is also called sample point. ω_i denotes the weight of RBF. $\|\mathbf{x} - \mathbf{x}_i\|$ denotes the Euclidean distance. $\Phi(r) = \|r^3\|$ is usually selected as RBF during the interpolation of scattered points. In order to ensure the continuity and linear of surface, $\partial(\mathbf{x})$ is defined as follows:

$$\partial(\mathbf{x}) = a_0 + a_1 x_w + a_2 y_w + a_3 z_w. \quad (33)$$

In order to minimize the energy function and make (32) have a unique solution, the orthogonalization condition is defined:

$$\sum_{i=1}^n \omega_i = \sum_{i=1}^n \omega_i x_w = \sum_{i=1}^n \omega_i y_w = \sum_{i=1}^n \omega_i z_w = 0. \quad (34)$$

FIGURE 7: Experimental platform.

Additional constraint points are calculated on the basis of normal direction of scattered points. Interpolation constraint points and additional constraint points are substituted into formulas (32), (33), and (34). Thus, unique solution is obtained. Then, ω_i and a_k are substituted into $g(x)$. Thus, surface equation is obtained. At last, Bloomenthal algorithm [33] is used to achieve triangular facets of surface. In this situation, 3D surface obtained is often rougher. And the quality of interpolation surface is also affected by many narrow triangular facets. Therefore, triangle mesh needs some local optimization-processing.

7. Experiment and Analysis

Experimental platform is composed of computer (Acer TMP455, 16 G memory, 500 G SSD), manipulator system, and Kinect vision as shown in Figure 7. The effective measurement range of Kinect is 0.8~2.3 meters. And measurement accuracy decreases with increasing distance. The software includes VC++ 2010, Matlab 2012a, and Kinect for Windows SDK v1.7.

In accordance with 3D reconstruction steps, the experiments can be divided into end-effector motion capture, target recognition, and 3D surface fitting. The experiment of end-effector motion capture includes RGB video converting to frame stream, image fusion, image binarization, and the state estimation. The experiment of target recognition includes feature extraction, gray processing, threshold segmentation, feature vectorization, data normalization, and BP identification. The experiment of 3D reconstruction includes information extraction of target point cloud and surface fitting.

7.1. End-Effector Motion Capture. The video is converted into image frame format ($480 \times 640 \times 3$). There are a total of 73 frames. The target region will be captured when the end-effector is moving in each frame image. Regional boundaries are described by wireframe and its geometric center. Motion capture result of the end-effector is in square region at t_k as shown in Figure 8. Blue “o” denotes the center of target region obtained by three-frame differencing method. Blue frame denotes the moving region boundary obtained by three-frame differencing method. Motion state of the end-effector is estimated by Kalman filter. Thereby, the position and velocity of the end-effector are obtained. At time t_0 , the measurement covariance matrix is $\mathbf{P}_0 = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 100 \end{bmatrix}$. The measurement

FIGURE 8: Motion capture of the end-effector at t_k .

FIGURE 9: Position estimate.

noise covariance matrix is $\mathbf{R}_0 = \begin{bmatrix} 0.2845 & 0.0045 \\ 0.0045 & 0.0455 \end{bmatrix}$. The process noise covariance matrix is $\mathbf{Q}_0 = \begin{bmatrix} 0.01 & 0 & 0 & 0 \\ 0 & 0.01 & 0 & 0 \\ 0 & 0 & 0.01 & 0 \\ 0 & 0 & 0 & 0.01 \end{bmatrix}$. Red “o” denotes the center of target region obtained by Kalman filter. Red frame denotes the moving region boundary obtained by Kalman filter. During the experiment, blue center “o” is tracked by red center “□.” And blue frame is tracked by red frame.

Figure 9 shows the relation between actual position and predicted position. Here, the actual values represent the measurements, also called the observations. The curve is fitted by image pixel coordinates of 73 centers. And it reflects that center position changes in image sequences. The green line indicates the actual position of end-effector. The red line represents the Kalman estimate position of end-effector. From Figure 9, the position tracking is very stable in the initial period of time. But, in the intermediate period of time, there is a certain amount of pixel error. And the maximum error is 8 pixels. The reason is the variable motion of end-effector. In the end period of time, the position of end-effector is corrected, and the tracking becomes stable again. Thus, the position of end-effector can be obtained in real time.

FIGURE 10: Velocity estimate.

Figure 10 shows the relation between actual velocity and estimated velocity. Here, the actual values represent the measurements, also called the observations. And it reflects that center velocity changes in image sequences. The green line indicates the actual velocity of end-effector. The red line represents the Kalman estimate velocity of end-effector. From Figure 10, the velocity tracking is basically stable the whole time. Thus, 3D reconstruction can be carried out in real time in the positioning process.

7.2. Target Recognition Experiment. Edge contour of target object is extracted according to Section 4. Then, calculate the shape parameters (7 kinds) to obtain the sample set. Normalization processing of sample set is carried out. The situation of BP network training is shown in Figure 11. This training takes 5 seconds with 47 iterations. Sample set is divided into automatically training set, validation set, and test set. Blue line, green line, and red line represent their convergence situations, respectively. The network stops training when “mse” reaches $1.24e-8$. Validation set has a convergence when “mse” reaches $8.57e-6$. Test set has a convergence when “mse” reaches $7.84e-5$. Function gradient decreases from 1.49 to $9.41e-6$. Thin dotted line and “○” denote the best status of validation set. Thick dotted line denotes preset “mse” of stopping training.

The result of the identification is shown in Figure 12. The horizontal axis denotes the number of verification and test samples/group. Vertical axis denotes the classification (identification result). Blue “•” denotes network predictive result. Red “○” denotes the actual classification. If the classification is equal to 0.1, the target is cylindrical object. If the classification is equal to 0.2, the target is square object. If the classification is equal to 1, the target is spherical object. From Figure 12, we can see that there is a one-to-one correspondence between network classification and actual result. For validation sample, the recognition rate is 0.998. For test sample, the recognition rate is 0.997. High recognition

FIGURE 11: BP training.

FIGURE 12: Recognition result.

rate shows that the extracted features are comprehensive and critical. And it also shows that the design of BP network is rational. At last, a cylindrical object is selected randomly from test sample. Its coordinate of TC is (507 pixels, 306 pixels) according to formula (15).

7.3. Reconstruction Experiment. First of all, calibration results are imported. There are 20 calibration images. There are 4 calibration points in each image. Focal length is $f_{cx} = 587.944$, $f_{cy} = 589.418$, and $f = 6.1$ mm. Origin is $u_0 = 313.222$

FIGURE 13: Reconstruction result.

and $v_0 = 259.304$. Radial distortion coefficients are $k_1 = -0.307$ and $k_2 = 0.729$. Tangential distortion coefficients are $k_3 = 0.048$ and $k_4 = 0.013$. The coordinates of 80 calibration points are shown in Table 3. On the left is the image pixel coordinate. On the right is the image physical coordinate. Clamping mechanism (also called end-effector) has the maximum opening range 287 mm. According to the actual positioning requirement, the maximum permissible errors are 20 mm, 25 mm, and 20 mm, respectively, along “x-,” “y-,” and “z-” axis.

3D coordinates (including end-effector and cylindrical TC) are calculated according to formulas (13)–(28). Cylindrical TC is 214.2 mm, -3.9 m, and 825 mm. Under ideal conditions, it indicates the positioning successful if TC coordinates coincide with the center coordinate of end-effector. But, in the experiment, it is normal if there is a certain deviation. And the center coordinate of end-effector should gradually converge to TC coordinate.

For the video (including 73 frames), image coordinates and 3D coordinates of end-effector centers are shown in Table 4. (u, v) denotes image coordinate. (x, y, z) denotes

spatial coordinate of end-effector center with respect to base coordinate frame. From the table, we can see that the center of end-effector gradually approaches TC in the time domain. Absolute errors are $214.2 - 212.8 = 1.4$ mm, $-3.3 + 3.9 = 0.6$ mm, and $825 - 823 = 2$ mm along “x-,” “y-,” and “z-” axis, respectively.

Theoretical values are constant along “x-” axis. But experimental data fluctuate within a certain range. And the maximum random fluctuation is $225.7 - 208.2 = 17.5$ mm. Theoretical values are decreasing along “y-” axis. Experimental data are also decreasing. Theoretical values are increasing along “z-” axis. Overall trend of experimental data is also increasing. But, sometimes, these data are unchanged (such as 34th~39th frame and 40th~41st frame) or fluctuant (such as 42nd~46th frame and 50th~52nd frame) in some consecutive frames. The reason of deviation is low pixel accuracy of Kinect. 3D point cloud extraction is shown in Figure 13(a). Surface fitting is implemented by using triangular facets and morphological processing as shown in Figure 13(b). Paper only displays the reconstruction results of 5th, 25th, 45th, 65th, and 73rd frame.

TABLE 3: Calibration point coordinates.

Calibration images (N)	Calibration points (N)							
	1	2	3	4	1	2	3	4
	(pixel)	(pixel)	(pixel)	(pixel)	(mm)	(mm)	(mm)	(mm)
1	(395.44, 277.95)	(419.34, 431.35)	(211.45, 276.86)	(183.37, 435.75)	(0.0101, -0.051)	(0.1094, 0.2014)	(-0.2342, -0.053)	(-0.280, 0.208)
2	(316.38, 89.96)	(325.04, 384.09)	(143.69, 50.58)	(151.97, 425.29)	(-0.059, -0.360)	(-0.048, 0.1243)	(-0.348, -0.427)	(-0.330, 0.190)
3	(386.20, 78.20)	(376.58, 400.37)	(181.60, 59.85)	(176.88, 408.30)	(0.0564, -0.381)	(0.0402, 0.1461)	(-0.283, -0.409)	(-0.295, 0.1651)
4	(427.13, 27.68)	(409.01, 443.65)	(152.58, 16.75)	(155.62, 443.42)	(0.0101, -0.051)	(0.1094, 0.2014)	(-0.2342, -0.053)	(-0.280, 0.208)
5	(279.39, 112.28)	(367.26, 281.19)	(162.72, 226.26)	(179.53, 444.50)	(-0.122, -0.323)	(0.0231, -0.046)	(-0.314, -0.138)	(-0.287, 0.2236)
6	(328.32, 19.74)	(383.67, 444.30)	(68.72, 48.74)	(181.87, 416.62)	(-0.040, -0.478)	(0.0505, 0.2234)	(-0.470, -0.427)	(-0.284, 0.1752)
7	(511.48, 85.75)	(356.46, 400.65)	(258.13, 25.40)	(154.26, 442.60)	(0.2614, -0.370)	(0.0060, 0.1514)	(-0.156, -0.4673)	(-0.330, 0.2193)
8	(466.40, 174.56)	(352.62, 367.62)	(214.13, 166.97)	(185.33, 380.02)	(0.1866, -0.221)	(0.0001, 0.0948)	(-0.229, -0.235)	(-0.278, 0.1177)
9	(441.64, 411.50)	(238.88, 369.16)	(346.80, 237.88)	(155.20, 251.66)	(0.1464, 0.1689)	(-0.189, 0.0978)	(-0.010, -0.118)	(-0.326, -0.094)
10	(248.51, 271.29)	(503.75, 260.67)	(168.48, 408.76)	(407.92, 362.78)	(-0.174, -0.063)	(0.2498, -0.078)	(-0.304, 0.1654)	(0.0891, 0.0864)
11	(303.88, 294.29)	(593.38, 315.16)	(197.27, 387.80)	(557.31, 446.40)	(-0.081, -0.023)	(0.3969, 0.0089)	(-0.257, 0.1286)	(0.3374, 0.2266)
12	(368.17, 223.77)	(422.15, 434.15)	(211.43, 224.87)	(180.48, 439.08)	(0.0247, -0.139)	(0.1139, 0.2047)	(-0.233, -0.140)	(-0.285, 0.2151)
13	(502.23, 402.00)	(123.60, 443.40)	(404.50, 282.87)	(114.67, 295.45)	(0.2447, 0.1503)	(-0.379, 0.2227)	(0.0873, -0.040)	(-0.395, -0.024)
14	(294.12, 222.68)	(416.12, 422.37)	(133.91, 218.88)	(178.59, 444.40)	(-0.097, -0.140)	(0.1037, 0.1845)	(-0.362, -0.151)	(-0.288, 0.2243)
15	(338.10, 223.56)	(336.59, 445.48)	(184.89, 177.12)	(87.08, 389.13)	(-0.024, -0.143)	(-0.027, 0.2260)	(-0.279, -0.216)	(-0.439, 0.1304)
16	(327.46, 185.72)	(418.51, 284.77)	(204.42, 286.52)	(213.21, 449.03)	(-0.042, -0.202)	(0.1079, -0.040)	(-0.245, -0.038)	(-0.231, 0.230)
17	(535.38, 156.60)	(398.36, 407.26)	(173.82, 149.55)	(190.53, 430.82)	(0.3013, -0.254)	(0.0747, 0.1641)	(-0.296, -0.261)	(-0.270, 0.1981)
18	(376.88, 19.36)	(388.37, 445.41)	(235.43, 78.19)	(246.70, 396.15)	(0.0384, -0.478)	(0.0591, 0.2249)	(-0.193, -0.380)	(-0.177, 0.1423)
19	(425.73, 59.90)	(407.40, 425.41)	(266.62, 90.41)	(245.44, 382.71)	(0.1211, -0.410)	(0.0877, 0.1900)	(-0.144, -0.362)	(-0.176, 0.1236)
20	(375.46, 78.39)	(357.55, 397.71)	(203.60, 16.85)	(184.36, 444.33)	(0.0368, -0.382)	(0.0065, 0.1472)	(-0.2474, -0.481)	(-0.278, 0.222)

TABLE 4: Coordinate mapping of EEC.

Frames (F)	Coordinates (pixel/mm)			
	(u, v) (pixel)	x (mm)	y (mm)	z (mm)
1	(0, 0)	0	0	0
2	(508.4553, 149.0902)	211.5	-205.5	818
3	(509.6101, 150.4026)	216.4	-201.9	818
4	(510.5211, 151.8155)	215.8	-199.2	819
5	(511.7326, 154.8328)	213.1	-196.9	821
6	(517.1071, 155.5721)	215.0	-194.6	822
7	(516.2842, 153.7535)	216.4	-188.9	823
8	(512.8059, 162.4985)	215.8	-187.8	820
9	(515.8507, 157.8256)	213.1	-185.3	820
10	(517.8562, 167.0576)	221.2	-182.7	820
11	(518.5200, 170.5581)	221.2	-177.3	819
12	(517.8046, 172.1101)	222.2	-174.5	818
13	(518.7650, 174.1318)	221.9	-173.1	816
14	(520.5103, 176.8343)	221.4	-170.4	820
15	(516.6733, 177.7260)	222.4	-167.8	818
16	(516.5375, 178.9579)	219.1	-165.1	818
17	(517.3276, 183.5082)	219.1	-159.7	820
18	(516.6664, 184.1815)	214.9	-158.3	820
19	(515.8422, 183.4009)	216.3	-157.1	818
20	(518.1490, 188.9494)	210.8	-154.4	818
21	(516.7696, 193.0880)	212.2	-151.8	820
22	(516.4275, 194.7317)	214.2	-146.3	818
23	(514.9372, 196.9299)	208.2	-143.7	820
24	(515.4128, 200.1725)	209.6	-142.3	818
25	(516.3984, 203.2776)	212.9	-139.6	816
26	(514.6664, 203.9622)	211.5	-132.8	816
27	(515.3714, 208.1466)	216.4	-130.2	815
28	(514.3249, 209.6533)	215.8	-127.4	813
29	(513.2491, 214.3007)	213.1	-124.7	809
30	(519.5372, 214.9963)	217.8	-122.1	805
31	(519.6129, 217.2171)	217.0	-116.7	805
32	(511.5682, 220.1829)	210.8	-114.2	803
33	(520.2266, 222.1736)	212.2	-111.5	803
34	(519.8394, 224.4393)	223.1	-110.2	801
35	(518.6933, 226.3423)	221.7	-107.5	801
36	(518.2638, 228.8997)	221.6	-104.9	801
37	(521.0255, 232.5297)	225.7	-99.5	801
38	(518.5646, 234.5279)	221.5	-97.0	801
39	(519.1914, 236.7338)	222.9	-94.3	801
40	(516.9438, 238.1292)	219.3	-92.0	803
41	(519.1306, 240.7258)	223.4	-89.2	803
42	(517.4656, 241.7497)	221.2	-88.2	805
43	(517.3734, 244.1388)	221.2	-84.2	805
44	(517.4305, 247.5676)	222.2	-80.6	809
45	(516.0781, 249.6231)	221.9	-78.3	813

TABLE 4: Continued.

Frames (F)	Coordinates (pixel/mm)			
	(u, v) (pixel)	x (mm)	y (mm)	z (mm)
46	(516.8702, 253.1202)	221.4	-72.6	811
47	(515.9355, 254.5369)	220.0	-71.5	811
48	(515.6318, 257.4846)	220.5	-67.6	813
49	(516.2515, 259.7152)	222.4	-65.1	815
50	(515.8500, 261.7526)	221.3	-62.5	816
51	(515.4668, 265.0079)	221.3	-57.1	816
52	(516.9849, 267.5525)	222.4	-54.3	815
53	(514.6148, 268.7528)	219.1	-52.9	813
54	(514.1781, 270.8958)	219.1	-50.2	813
55	(511.7849, 272.8321)	214.9	-47.6	813
56	(512.3674, 274.2286)	216.3	-44.9	813
57	(513.0664, 278.0799)	217.8	-39.5	813
58	(513.4077, 279.9725)	217.8	-38.1	813
59	(511.8389, 280.8945)	215.0	-36.9	813
60	(512.3725, 276.9616)	216.4	-34.2	813
61	(511.8123, 285.3395)	215.8	-30.2	816
62	(509.4354, 287.7515)	213.1	-27.6	816
63	(512.0752, 291.5637)	217.8	-22.1	818
64	(511.1836, 293.4690)	217.0	-19.5	820
65	(507.0845, 292.9423)	210.8	-20.9	818
66	(508.7903, 294.5573)	212.2	-18.2	818
67	(509.1120, 299.4822)	214.2	-11.4	820
68	(505.7227, 301.1344)	208.2	-8.8	818
69	(506.8830, 303.1554)	209.6	-6.0	818
70	(508.0559, 305.9869)	212.9	-3.2	820
71	(507.6709, 305.8390)	211.5	-3.1	820
72	(509.1354, 305.9515)	210.3	-3.2	823
73	(508.1354, 306.3515)	212.8	-3.3	823

7.4. Comparative Experiment. In [1], accumulated error cannot be corrected. In [4], 3D reconstruction based on monocular camera only rely on mathematical model. And the requirement for light source is harsh. In [6], 3D reconstruction based on binocular vision requires pattern matching and a lot of computing. And the effect of reconstruction is significantly reduced in the case of large baseline distance. From Table 5, we can obtain the following conclusions. Paper's method (1) can meet large computational applications, (2) has low light source demanding, and is mainly subject to ultraviolet rays. (3) Neural network recognition algorithm reduces the reliance on mathematical model and (4) improves the reconstruction efficiency and accuracy.

8. Conclusion

First of all, paper proposes the Kalman filtering method based on three-frame differencing to capture the end-effector motion. Secondly, target objects are identified by using BP

TABLE 5: Method comparison.

Parameters	Reference [1]	Reference [4]	Reference [6]	Paper's method
① Amount of calculation	47548	32361	61387	53982
② Time consuming/s	5.405	10.249	15.652	2.081
③ Subject to light source influence	Small	Larger	Larger	Small
④ Rely on mathematical model	Large	Larger	Larger	Large
⑤ Reconstruction accuracy/mm	0.15	0.40	0.19	0.15

Note: ① denotes the amount of 3D point cloud.

classification idea. And calculate the coordinates of TC. Then, the center of end-effector gradually converges to TC. This convergence indicates that the positioning is successful. Finally, surface fitting of 3D point cloud is achieved by triangular facets and morphology processing. Comparative experiment shows that paper's method is more practicable and efficient.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

This paper is supported by Beijing project "Equipment Development and Demonstration Application in Rail Accident Scene Emergency." And Project no. is Z131100004513006.

References

- [1] S. Izadi, R. A. Newcombe, and D. Kim, "Kinect fusion: real-time dynamic 3D surface reconstruction and interaction," in *Proceedings of the Association for Computing Machinery's Special Interest Group on Computer Graphics and Interactive Technologies (UIST '11)*, p. 23, Santa Barbara, Calif, USA, October 2011.
- [2] X. Ma, Y. Xue, and R. Yang, "Kinect-based real-time 3D reconstruction of human and its application," *Journal of Computer-Aided Design and Computer Graphics*, vol. 26, no. 10, pp. 1720–1726, 2014.
- [3] F.-S. Guo and W. Gao, "Batch reconstruction from UAV images with prior information," *Acta Automatica Sinica*, vol. 39, no. 6, pp. 834–845, 2013.
- [4] X. Z. Li, A. L. Yang, B. L. Qin, and S. M. Jia, "Monocular camera three dimensional reconstruction based on optical flow feedback," *Acta Optica Sinica*, vol. 35, no. 5, Article ID 0515001, pp. 1–9, 2015.
- [5] D.-Y. Lü, Z.-P. Huang, G.-H. Tao, N.-H. Yu, and J.-K. Wu, "Dynamic Bayesian network model based golf swing 3D reconstruction using simple depth imaging device," *Journal of Electronics and Information Technology*, vol. 37, no. 9, pp. 2076–2081, 2015.
- [6] Y.-M. Lin, N.-G. Lü, X.-P. Lou, and M.-L. Dong, "Robot vision system for 3D reconstruction in low texture environment," *Optics and Precision Engineering*, vol. 23, no. 2, pp. 540–549, 2015.
- [7] S. Izadi, D. Kim, O. Hilliges et al., "KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera," in *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*, pp. 559–568, Santa Barbara, Calif, USA, October 2011.
- [8] F. Kahl and R. I. Hartley, "Multiple-view geometry under the L_∞ -Norm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 9, pp. 1603–1617, 2008.
- [9] B. Hassine, B. H. Saber, K. N. Khaled, and M. Hassen, "Kuka robot control based Kinect image analysis," in *Proceedings of the International Conference on Individual and Collective Behaviors in Robotics*, pp. 21–26, Sousse, Tunisia, December 2013.
- [10] R. Akshara, K. P. Prem, A. Mridul, G. Rohit, and B. Laxmidhar, "Grasping region identification in novel objects using microsoft Kinect," in *Neural Information Processing: 19th International Conference, ICONIP 2012, Doha, Qatar, November 12–15, 2012, Proceedings, Part IV*, vol. 7666 of *Lecture Notes in Computer Science*, pp. 172–179, Springer, Berlin, Germany, 2012.
- [11] Cornell-rgb-d-dataset, <http://pr.cs.cornell.edu/sceneunderstanding/data/data.php>.
- [12] Z. Zhang, "Flexible camera calibration by viewing a plane from unknown orientations," in *Proceedings of the 7th IEEE International Conference on Computer Vision (ICCV '99)*, vol. 1, pp. 666–673, Kerkyra, Greece, September 1999.
- [13] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.
- [14] C. D. Herrera, J. Kannala, and J. Heikkilä, "Joint depth and color camera calibration with distortion correction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 10, pp. 2058–2064, 2012.
- [15] D. Holz, S. Holzer, R. B. Rusu, and S. Behnke, "Real-time plane segmentation using RGB-D cameras," in *RoboCup 2011: Robot Soccer World Cup XV*, T. Röfer, N. M. Mayer, J. Savage, and U. Saranlı, Eds., vol. 7416 of *Lecture Notes in Computer Science*, pp. 306–317, 2012.
- [16] C. Raposo, J. P. Barreto, and U. Nunes, "Fast and accurate calibration of a kinect sensor," in *Proceedings of the 2013 International Conference on 3D Vision (3DV '13)*, pp. 342–349, IEEE, Seattle, Wash, USA, July 2013.
- [17] J. Smisek, M. Jancosek, and T. Pajdla, "3D with Kinect," in *Proceedings of the 2011 IEEE International Conference on Computer Vision Workshops (ICCV '11)*, pp. 1154–1160, IEEE, Barcelona, Spain, November 2011.
- [18] Y. Z. Hu and L. Y. Li, "Series actuator end integrated positioning analysis based-on multilayer perceptron neural network," *Transactions of the Chinese Society of Agricultural Engineering*, vol. 32, no. 1, pp. 22–29, 2016.
- [19] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the Association for Computing Machinery*, vol. 24, no. 6, pp. 381–395, 1981.

- [20] A. Uckermann, C. Elbrechter, R. Haschke, and H. Ritter, "3D scene segmentation for autonomous robot grasping," in *Proceedings of the 25th IEEE/RSJ International Conference on Robotics and Intelligent Systems (IROS '12)*, pp. 1734–1740, IEEE, Vilamoura, Portugal, October 2012.
- [21] G. Du, P. Zhang, and D. Li, "Online robot calibration based on hybrid sensors using Kalman Filters," *Robotics and Computer-Integrated Manufacturing*, vol. 31, pp. 91–100, 2015.
- [22] C. G. Refael and E. W. Richard, *Digital Image Processing*, Publishing House of Electronics Industry, 2nd edition, 2002.
- [23] C.-C. Chang, J.-Y. Hsiao, and C.-P. Hsieh, "An adaptive median filter for image denoising," in *Proceedings of the 2008 2nd International Symposium on Intelligent Information Technology Application (IITA '08)*, pp. 346–350, IEEE, Shanghai, China, December 2008.
- [24] Q. Liu and T.-S. Lin, "Image edge detection algorithm based on mathematical morphology," *Journal of South China University of Technology (Natural Science)*, vol. 36, no. 9, pp. 113–121, 2008.
- [25] S. Indrazno, B. Laxmidhar, T. M. McGinnity, and C. Sonya, "A position based visual tracking system for a 7 DOF robot manipulator using a Kinect camera," in *Proceedings of the IEEE World Congress on Computational Intelligence (WCCI '12)*, Brisbane, Australia, June 2012.
- [26] M. Štrbac, M. Marković, and D. B. Popović, "Kinect in neurorehabilitation: computer vision system for real time hand and object detection and distance estimation," in *Proceedings of the 9th International Conference on High Capacity Optical Networks and Enabling Technologies (HONET '12)*, pp. 127–132, IEEE, Belgrade, Serbia, December 2012.
- [27] K. Khoshelham and S. O. Elberink, "Accuracy and resolution of kinect depth data for indoor mapping applications," *Sensors*, vol. 12, no. 2, pp. 1437–1454, 2012.
- [28] D. C. Brown, "Decentering distortion of lenses," *Photogrammetric Engineering*, pp. 444–462, 1966.
- [29] J. Heikkila and O. Silven, "Four-step camera calibration procedure with implicit image correction," in *Proceedings of the 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1106–1112, June 1997.
- [30] M. Zhao, F. Tan, C.-W. Fu, C.-K. Tang, J. Cai, and T. J. Cham, "High-quality Kinect depth filtering for real-time 3D telepresence," in *Proceedings of the 2013 IEEE International Conference on Multimedia and Expo (ICME '13)*, pp. 1–6, IEEE, San Jose, Calif, USA, July 2013.
- [31] S.-C. Yu, X.-Z. Zang, Y.-S. Gao, and J. Zhao, "Explicit-implicit correction method for image distortion," *Journal of Harbin Institute of Technology*, vol. 40, no. 9, pp. 1387–1390, 2008.
- [32] H. C. Daniel, K. Juho, and H. Janne, "Accurate and practical calibration of a depth and color camera pair," in *Computer Analysis of Images and Patterns: 14th International Conference, CAIP 2011, Seville, Spain, August 29–31, 2011, Proceedings, Part II*, vol. 6855 of *Lecture Notes in Computer Science*, pp. 437–445, Springer, Berlin, Germany, 2011.
- [33] J. Bloomenthal, "An implicit surface polygonizer," in *Graphics Gems IV*, pp. 324–349, Academic Press Professional, San Diego, Calif, USA, 1994.

Hindawi

Submit your manuscripts at
<http://www.hindawi.com>