Target Segmentation and Pose Estimation for Rapid Grabbing Using Depth Information

Ru-ning Wang¹, Chang-long Wang¹ and Jun Zheng¹,a
¹Department of Mechanical Engineering, Tsinghua University, Beijing, China
a Corresponding author: zhengj@mail.tsinghua.edu.cn

Abstract. In order to realize rapid grabbing of stacked objects, we propose a quick and accurate method to realize target segmentation and pose estimation using only depth information. We use filtering to remove the noises in detection. Then we detect the contours and use the averages and gradients of depth to find the contour of the top plane. We calculate the centroid and normal vector of the top plane to get the pose estimation. Thus we can realize the rapid grabbing of the top object, then achieve rapid grabbing of stacked objects one by one.

1. Introduction
The grabbing of three-dimensional stacked objects has been widely used, such as object sorting, industrial assembly and medical treatment. Traditional manual operation has lower efficiency, stability and security, so it has been replacing by automatic grabbing as 3D object recognition technologies develop quickly.

Traditional methods of 3D object recognition mainly combine the 2D images got from different viewpoints to compute 3D data [1]. This kind of methods require high computational complexity and high hardware requirements. Hai-yong Wei in our lab designed an algorithm, which can achieve affine invariance and high speed in stereo vision system. But this method cannot handle the circumstances very well when the objects are stacked or in cluttered scenes, which are the main problems in many stereo systems.

Recently, the RGB-D cameras become more and more widespread, like Kinect sensor from Microsoft. This kind of cameras provide synchronized color and depth images [2]; it can provide more accurate geometrical information directly compared to 2D images [3]. Besides, it has the advantages of low cost and high speed, since it reduces the computational complexity and hardware requirements significantly. Most methods using RGB-D cameras need to calibrate the visual and depth images first and to generate point clouds, which can be used for object segmentation.

In this paper, we aim at rapid grabbing of stacked objects. In actual process, we often grab one object each time from the top, so we only need to realize the recognition of the top object. Based on this process, we propose a real-time object segmentation and pose estimation method using only depth information, instead of using point clouds. Therefore, we can reduce the computational load and improve detection speed further.

2. The Kinect depth sensor mechanism
The Kinect sensor from Microsoft incorporates an infrared projector, an infrared camera and a color camera. Fig.1 is a picture of Kinect sensor. The depth sensor consists of an infrared projector and an infrared camera. The infrared light emitted by the infrared sensor is evenly projected into the...
measuring space through the grating located in front of the lens of the projector. The speckle image is formed through the reflection of the object surface. By comparing the reference speckle image, we can obtain the actual distance between the target and the sensor. Therefore, the depth image can be created since pixel values of depth image are represented by the actual distance information of according pixels.

Figure 1. Kinect sensor.

3. Depth data preprocessing
There are two kinds of noises in the original depth map obtained by Kinect. One is the gross error of some discrete points, which values are got incorrectly; and the other one is random noise introduced by the measurement error. The latter can be considered to accord with the Gaussian distribution. Aiming at these two kinds of noises, we use the methods of clustering and Kalman filtering, which can effectively eliminate the outliers in the depth data, reduce the jump of the data, and obtain an optimal estimation of the depth data.

3.1 Clustering filtering
In order to get the optimal estimate of the current depth data, we need to continuously measure the depth value. Firstly, we use clustering method to deal with the first \( n \) frames of the measure results to get rid of outliers. Considering that the probability and magnitude of outliers are uncertain, it is impossible to determine the number of clusters and the central point of each data cluster in advance, so Sequential Leader algorithm is suitable for separating the correct measurement and outliers.

For the depth value \( d \) from continuous \( n \)-frame data of a certain point, the depth of the first frame is considered as a clustering center point; and then we calculate each frame after one by one. If the distance between current depth value and the previous clustering center is less than the set threshold, the data is classified in the former cluster and the center of this clustering is recalculated; otherwise, the current depth value is used as a new clustering center to regenerate a new cluster. After traversing all the frames, the cluster with the largest number of data points is the cluster that we need. And the center point of this cluster is the reference value of depth measurement; other depth data are processed as outliers.

For a cluster with \( k \) data points, which center is \( c \), a threshold value is used to determine whether a new data \( d \) is in such a cluster, as shown in Eq.1:

\[
\|d - c\| \leq \text{threshold}
\]

(1)

Then we can update the new cluster center \( c' \) from Eq.2:

\[
c' = \frac{kc + d}{k+1}
\]

(2)

3.2 Kalman filtering
After clustering filtering, the outliers of the depth data have been eliminated, and then all the points in the cluster can be filtered by a Kalman filter to reduce the data jump and get the optimal estimation of the depth data.

Kalman filter is a kind of predict-correct filter. The prediction process uses the state variables of the previous time to predict variables of the current time [4]. It mainly includes the following two parts:

a. Update state variables.
\[ x_{i} = A_{i-1} \]  
\[ b. \text{Update error covariance matrix.} \]
\[ P_{i} = A_{i-1} P_{i-1} A_{i-1}^T + Q \]  

The correction process obtains the optimal estimate of the current state by the predicted value and the current measurement value. The following three steps are included:

a. Calculate Kalman gain.
\[ K_{i} = P_{i}^{-1} H [H P_{i}^{-1} H^T + R]^{-1} \]  
b. Correct state variables.
\[ x_{i} = x_{i}^{'} + K_{i} (y_{i} - H x_{i}^{'}) \]  
c. Correct error covariance matrix.
\[ P_{i} = P_{i}^{-1} - K_{i} H P_{i}^{-1} \]

Considering that the measurement of each point in depth data is independent, Kalman filtering only deals with scalar values. The observation scene is static, so both the state transition matrix \( A \) and the observation matrix \( H \) can be set to a unit matrix (1 in this case). The process noise \( Q \) is zero, but usually a minimum is used to represent the covariance matrix of the process noise (variance in this case). The covariance matrix \( R \) of observation noise (the variance of observed noise in this case) is the parameter to be set. In this case, the variance of all the data points in the cluster can be directly used as the variance of the observed noise.

Fig.2 is a continuous statistics of the depth of one image point (after clustering filtering process). It can be seen that the use of Kalman filtering can effectively reduce the jump of the depth value and improve the stability of the data.

4. Pose estimation of stacked objects

4.1 Segmentation of target area

After preprocessing, image normalization is performed to obtain the depth image; each pixel value uses 8 bits. Then, bilateral filtering is used to smooth the depth map. Bilateral filtering is a kind of edge-preserving filtering, which can make the pixel values of the same plane smoother without losing the edge gradient information. The kernel function of bilateral filtering is the result of the combination of spatial kernel and pixel range kernel. In the flat region of the image, the pixel value changes very little, and the corresponding pixel range domain weight is close to 1. At this time, the weight of the spatial domain plays a major role, which is similar to Gaussian blur. While in the edge region of the image, the pixel value varies significantly, and the pixel value is close to 1. In this region, the weight of the scope domain becomes larger, thus maintaining the edge information. Eq.8 and Eq.9 give the operation of bilateral filtering [5].

\[ \omega(i, j, k, l) = \exp(-\frac{(i-k)^2 + (j-l)^2}{2\sigma_d^2} - \frac{||I(i,j) - I(k,l)||^2}{2\sigma_r^2}) \]  

\[
\begin{align*}
\end{align*}
\]  

Figure 2. Depth values before/after Kalman filtering.
In order to realize the sorting of stacked objects, we need to start from the top of the stacked objects, then do the sorting one by one, and finally realize the sorting of all objects. The segmentation of the top object can be achieved by identifying the top plane. In depth maps, pixel values are represented by the depth values of pixels, so there are obvious differences in pixel values between planes with different depth values. The contour information of the depth map can be extracted, and then the average depth of all pixels within each contour can be obtained. The contour with the lowest average depth can be assumed as the top plane of the top object first.

Considering there might be the situation that the plane with the lowest average depth is covered a small part by another object, that is, this contour is not a complete outline of a plane. So for the point on the contour, we take its 7 neighboring points and investigate whether it is outside the contour area. If it is an exterior point, we subtract the depth value of the contour point from the exterior point. If all the depth gradients of one contour point are more than 0, then it proves that this contour point is the top of surrounding area. If every contour point satisfies the above conditions, the whole contour area can be considered the top of the surrounding area. Otherwise, it proves that part of this plane is covered by another object.

In actual operation, we cannot avoid the existence of noise in contour detection. Even if the contour area is the top area, it cannot be completely guaranteed the contour point depth is less than the depth values of all exterior neighboring points. Therefore, the threshold coefficient $\mu$ is set to measure the proportion of the number of exterior neighboring points that meet the requirements to all neighboring points. The size of $\mu$ is relate to the parameters of edge detection. In our experiments, the effect is better when $\mu = 0.8$ is obtained.

![Figure 3. Segmentation of target area. (a) Color image. (b) Incomplete plane with lowest depth average. (c) Complete top plane. (d) Second complete top plane after the first object has been taken away.](image)

4.2 Centroid and pose estimation of target object

Since the target region has been identified, the pixel coordinates $(u, v)$ and depth values $Z$ of all pixels within the target region can be obtained. The coordinate values of all pixels in the camera coordinate system are calculated according to the internal parameter matrix $K$ of the depth camera. The formula is based on Eq.10:

$$
\begin{align}
Z = K \begin{pmatrix}
X \\
Y \\
1
\end{pmatrix} = \begin{pmatrix}
f_x & 0 & c_x \\
0 & f_y & c_y \\
0 & 0 & 1
\end{pmatrix} \begin{pmatrix}
X \\
Y \\
Z
\end{pmatrix}
\end{align}
$$

(10)
\( K \) is the internal parameter matrix of the depth sensor.

The pose estimation of the target is mainly to estimate the centroid and normal vector of the top plane. The centroid coordinates can be estimated directly from the average of all point coordinates. The PCA method can be used to estimate the normal vector [6]. This method has the advantage of simple structure and small amount of computation.

Subtracting the coordinates of all points within the target region from the centroid coordinates, we can obtain an \( N \times 3 \) matrix \( A \) with zero column mean. Assuming the estimated normal vector is \( x \), the vector can be obtained by the equation \( Ax = 0 \). By using the algorithm of least squares estimation, we can obtain the optimal formula as follows:

\[
\arg \min_x \|Ax\| \quad \& \quad \|x\| = 1
\]  

(11)

\( x \) represents the direction in which the change of the target data set is minimum, which is the eigenvector corresponding to the minimum eigenvalue of the covariance matrix \( A^T A \). \( A^T A \) is a \( 3 \times 3 \) square matrix, and \( x \) can be obtained by solving all its eigenvalues and corresponding unit eigenvectors.

The PCA method is fast and simple, but the estimated normal vector is vulnerable to the influence of the outliers. Although all the three-dimensional coordinate points are assumed to be on the top plane of the object, some points still deviate from the plane. Based on this assumption, RANSAC algorithm can be used to filter out the points with large errors and extract all the data points that are closest to the local plane [7]. The optimal estimate of the normal vector can be obtained by PCA method.

The RANSAC removal process of outliers is as follows:

a. We randomly take samples of \( m \) points from the data set to calculate normal vectors.

b. We calculate whether all the points are on the plane determined by the normal vector. This process is determined by a tolerance range \( \varepsilon \); then, we count the number of points.

c. We reselect \( m \) points randomly and repeat the first two operations until the set number of iterations is reached.

d. We find the situation with largest number of data points, which is the final fitting result.

Fig.4 and Fig.5 are the data sets before and after RANSAC algorithm processing. It can be seen that after processing, the points with large errors can be eliminated and the optimal data set can be obtained. After RANSAC, we use the new point set to calculate normal vector again to get the optimal estimation.

Figure 4. Point set before RANSAC.
5. Grabbing experiments

In order to test the accuracy of our algorithms, we use the high-accuracy three-dimensional platform to move a specific calibration block for 50 mm each time, as is shown in Fig. 6. We calculate the distance between the centroids we obtain. Then we compare the difference between the centroid coordinates in world coordinate system (actual centroid coordinates) and that in camera system (calculated coordinates). According to the experiments, the relative error is within 4%, as is shown in Table 1. Meanwhile, since we move the calibration block in x, y, z, directions, the normal vectors should be the same. We calculate the normal vectors and obtain the error as is shown in Table 2. The small deviation of normal vectors proves that our algorithms are robust.

We calibrate the relationship between the depth camera and the robot arm, and according to the centroid and the normal vector of the plane, we obtain the position and pose of the top plane in the robot coordinate system. We use a suction cup powered by vacuum pump as our grabbing device. The grabbing experiment is shown in Fig. 7. In 20 groups of grabbing experiments, 10 objects in each group are stacked randomly. The experiment proves that the robot arm can grab each object accurately, thus achieving the effect of automatic rapid grabbing of all the objects.

Compared with the grabbing process, the total time of depth information collection, region segmentation and normal vector estimation is very fast, and the average time is about 1.2 seconds.

Figure 5. Point set after RANSAC and the normal vector.

Figure 6. Three-dimensional platform and calibration block.
### Table 1. Error of centroid position.

| Actual centroid coordinates | Calculated centroid coordinates | Relative Errors |
|----------------------------|--------------------------------|-----------------|
| X | Y | Z | X’ | Y’ | Z’ | — |
| 0 | 0 | 0 | 237.93 | 231.86 | 759.41 |
| 0 | 50 | 0 | 238.03 | 194.85 | 793.63 | 0.0082 |
| 0 | 150 | 0 | 237.47 | 115.76 | 861.76 | 0.0318 |
| 0 | 200 | 0 | 237.85 | 75.85 | 896.72 | 0.0392 |
| 0 | 200 | 50 | 239.80 | 44.20 | 860.36 | 0.0337 |
| 0 | 200 | -50 | 234.96 | 117.25 | 930.62 | -0.0005 |
| 0 | 200 | 100 | 242.66 | 10.46 | 825.78 | 0.0339 |
| 50 | 0 | 0 | 187.14 | 228.55 | 758.72 |
| 100 | 0 | 0 | 136.84 | 227.43 | 757.56 | 0.0121 |
| 150 | 0 | 0 | 86.49 | 225.21 | 755.65 | 0.0109 |
| 200 | 0 | 0 | 35.78 | 221.98 | 754.89 | 0.0122 |

### Table 2. Error of normal vector.

| Normal vectors | a | b | c | Errors |
|----------------|---|---|---|--------|
| -0.036576 | 0.267379 | 0.962897 | — |
| -0.033011 | 0.266522 | 0.963263 | 0.003685 |
| -0.005241 | 0.274603 | 0.961543 | 0.032185 |
| -0.021374 | 0.260896 | 0.965130 | 0.016677 |
| -0.041496 | 0.275613 | 0.960373 | 0.009919 |
| -0.024099 | 0.269876 | 0.962594 | 0.012727 |
| -0.082283 | 0.283014 | 0.955580 | 0.048858 |
| -0.015521 | 0.288981 | 0.957209 | 0.030697 |
| -0.028111 | 0.259741 | 0.965269 | 0.011646 |
| -0.026179 | 0.268231 | 0.962999 | 0.010432 |
| -0.068730 | 0.266277 | 0.961443 | 0.032205 |

Figure 7. Grabbing experiment.

### 6. Conclusions

In this paper, we proposed a method which can realize rapid grabbing using only depth information. We segmented the top plane of all the objects and calculated the centroid and normal vector of it to obtain the pose estimation. The accuracy of our detection algorithms can meet industrial requirements.
Our system has the advantages of simple structure and small computational load. Therefore, the method proposed in this paper is suitable for the occasion of high efficiency requirements.

Acknowledgement
This research was fully funded by the National Science and Technology Major Project (2015ZX04005006).

References
[1] D. Buchholz, *Bin-Picking—5 Decades of Research* (Springer International Publishing, 2016)
[2] J. Han, et al, IEEE Trans. Cybern. 43,5(2013):1318-1334
[3] Y. Guo, et al, IEEE Trans. Pattern Anal. Mach. Intell. 36,11(2014):2270-2287
[4] R.E. Kalman, J. Basic Eng. 82,1(1960):35-45
[5] C. Tomasi, R. Manduchi, Proc. IEEE Int. Conf. Comput. Vision (1998):839-846
[6] J.V. Miller, C.V. Stewart, Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit. (1996):300-306
[7] M.A. Fischler, R.C. Bolles, Commun. ACM 24, 6(1981):381-395