Sharp Corner/Edge Recognition in Domestic Environments Using RGB-D Camera Systems

Junlong Zhou, Jianming Yan, Tongquan Wei, Kaijie Wu, Xiaodao Chen, and Shiyan Hu

***Abstract*—This brief proposes a sharp corner/edge detection and evaluation scheme for RGB-D camera-based domestic applications. The proposed approach first preprocesses the original depth map of objects in domestic environments captured by an RGB-D camera system and then uses the recovered depth map to recognize sharp corners/edges on objects and calculate their sharpness. Furthermore, the approaching speed of a mobile manipulator equipped with the system is derived, and the threat of detected sharp corners/edges is evaluated. To the best of our knowledge, this is the first work that explores sharp corner/edge detection and evaluation in the literature. Simulation results demonstrated that the proposed algorithm can effectively identify dangerous sharp corners/edges in domestic environments.**

***Index Terms*—Domestic environments, RGB-D camera systems, sharp corner/edge recognition.**

I. Introduction

RGB-D camera is an emerging trend of technologies that can capture color images along with depth data at each pixel. Compared with color camera, it is capable of providing higher accuracy and robustness using the additional depth information. It has been shown that the performance of an RGB-D camera is close to that of the laser for short-range environments [1]. This observation enables a broad range of RGB-D camera-based indoor applications. In this brief, we propose a sharp corner/edge detection and evaluation scheme for RGB-D camera-based domestic applications. Consider a dangerous scenario that may happen in domestic environments. A mobile manipulator is directly approaching to an object (e.g., table) with sharp corners/edges at speed v, as illustrated in Fig. 1. An intelligent RGB-D camera system integrated with algorithms that can recognize the sharp corners/edges and calculate the speed would be helpful to dismiss the poential danger. The proposed scheme running by the system equipped for a mobile manipulator can recognize the sharp corners/edges in a short range that may pose threats to the mobile manipulator such as a robot or a toddler.

Manuscript received December 20, 2014; revised March 22, 2015; accepted May 19, 2015. Date of publication July 17, 2015; date of current version September 25, 2015. This work was supported in part by the NSF of Shanghai under Grant 12ZR1409200 and in part by the STC of Shanghai under Grant 14DZ2260800. This brief was recommended by Associate Editor G. J. Dolecek.

J. Zhou, J. Yan, and T. Wei are with the Shanghai Key Laboratory of MIP, CST Department, East China Normal University, Shanghai 200241, China (e-mail: tqwei@cs.ecnu.edu.cn).

K. Wu is with Chongqing University, Chongqing 400030, China (e-mail: kaijie@cqu.edu.cn).

X. Chen is with China University of Geosciences, Wuhan 430074, China.

S. Hu is with Michigan Technological University, Houghton, MI 49931-1295 USA.

Color versions of one or more of the figures in this brief are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSII.2015.2458051

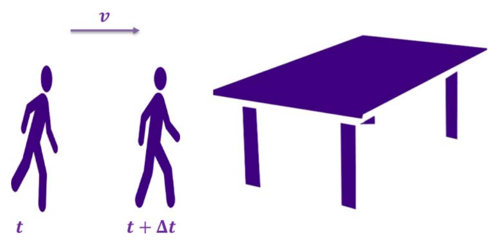


Fig. 1. Dangerous scenario in the domestic environments.

The first step of identifying sharp corners/edges on an object is to detect edges on the object. Several linear and nonlinear Gaussian filtering algorithms for edge detection were discussed in [2], and techniques on gray scale and color edge detection were reviewed in [3] and [4]. Recently, a novel morphological gradient and generalized type-2 fuzzy-logic-based edge detection approach was presented in [5]. However, all the aforementioned works did not use depth data for edge detection.

The depth data of RGB-D images are mainly exploited in image processing for object recognition and categorization. Herbst *et al.* [6] presented a robust multiscene analysis-based algorithm for object discovery using the color+depth data from RGB-D cameras. The authors described an unsupervised feature learning technique for RGB-D-based object recognition [7]. Blum *et al.* [8] introduced a learned feature descriptor to solve the feature extraction problem for RGB-D object recognition. Although RGB-D data have been successfully employed for object recognition in image processing, their potential effects on sharp corner/edge detection on objects in domestic environments are not investigated yet.

II. Recognition and Evaluation Scheme

The depth maps generated by the RGB-D camera system are preprocessed by an existing image filtering technique. We refer the reader to [9] for details of the technique.

A. *Recognize Sharp Corners/Edges and Calculate Sharpness*

Let $P_i = (P_{i,x}, P_{i,y}, P_{i,d})$ be the ith pixel of a preprocessed depth map, where $P_{i,x}$, $P_{i,y}$, and $P_{i,d}$ are x and y coordinates and depth of pixel P_i, respectively. The depth $P_{i,d}$ indicates the distance between the RGB-D camera and the pixel, which is a function of $P_{i,x}$ and $P_{i,y}$ [10]. Let $\Delta D_e(P_i, P_j)$ and $\Delta D_d(P_i, P_j)$ be the Euclidean distance and depth difference between pixels P_i and P_j, respectively, i.e.,

$$\Delta D_e(P_i, P_j) = \sqrt{(P_{i,x} - P_{j,x})^2 + (P_{i,y} - P_{j,y})^2} \quad (1)$$

$$\Delta D_d(P_i, P_j) = \|P_{i,d} - P_{j,d}\|. \quad (2)$$

The pixels inside circle $C(P_i, R)$ are defined as the neighbors of P_i, where P_i is the centroid and R is the radius. If P_i has the smallest depth as compared to its neighboring pixels in $C(P_i, R)$, it is deemed to be a sharp corner/edge. Since pixels on edges of preprocessed depth images are candidates of sharp corners/edges, a detection algorithm is proposed to identify sharp corners/edges from pixels on edges of the preprocessed depth map, as shown in Algorithm 1. Similar to the RGB-D image filtering technique [9], this brief also extracts the edges of preprocessed depth maps using the Canny edge detector [11].

Algorithm 1: Recognize the sharp corners/edges

```
Input: Set $\Psi$ of pixels on edges of a preprocessed depth image, radius $R$,
       dangerous distance threshold $D_1$, minimum euclidean distance threshold $\epsilon$
Output: Set $\Phi$ of sharp corners/edges
1  initialize the set $\Phi$ of sharp corners/edges to $\varnothing$;
2  select the pixels in set $\Psi$ to create a small-root heap using $\Omega$ =
   Modified-BUILD-MIN-HEAP($\Psi$); // filter out pixels beyond dangerous distance
3  for $i = 1$ to $sizeof(\Omega)$ do
4      $flag = 0$;
5      if $\Phi$ is not empty then
6          for $j = 1$ to $sizeof(\Phi)$ do
7              derive the euclidean distance $\Delta D_e(P_i, P_j)$ between pixel $P_i$ and
               $P_j$ using (1); // $P_j \in \Phi$ is a detected sharp corner/edge
8              if $\Delta D_e(P_i, P_j) \le \epsilon$ then
9                  $flag = 2$; break; // $P_i$ is deemed the same as $P_j$, lines 5-9
                   filter out duplicated sharp corners/edges
10     if $flag \neq 2$ then
11         initialize a temporary set $\mho$ to $\varnothing$;
12         store $P_i$'s neighboring pixels in $\mho$, where these pixels are in $C(P_i, R)$;
13         for $k = 1$ to $sizeof(\mho)$ do
14             if $(P_i \to P_{i,d}) > (Q_k \to Q_{k,d})$ then
15                 break; // $Q_k \in \mho$, $P_i$ is not a sharp corner/edge
16         if $k == sizeof(\mho)$ then
17             $flag = 1$; // $P_i$ is a sharp corner/edge
18     if $flag == 1$ then
19         $P_i$ is added to set $\Phi$ from the heap $\Omega$;
20     update the small-root heap using $\Omega$ = MIN-HEAPIFY($\Omega$);
```

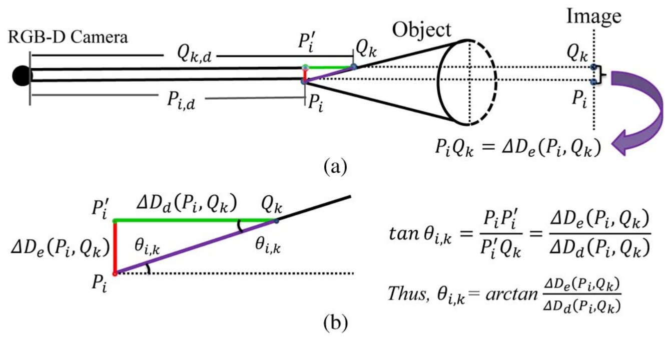
Algorithm 2: Calculate sharpness of detected sharp corners/edges

```
Input: Set $\Phi$ of sharp corners/edges detected by Algorithm 1
Output: Sharpness $S_i$ of detected sharp corners/edges $P_i \in \Phi$
1  for $i = 1$ to $sizeof(\Phi)$ do
2      initialize a temporary set $\mho$ to $\varnothing$;
3      store $P_i$'s neighboring pixels in $\mho$, where these pixels are inside $C(P_i, R)$;
4      for $k = 1$ to $sizeof(\mho)$ do
5          calculate euclidean distance $\Delta D_e(P_i, Q_k)$ and depth difference
           $\Delta D_d(P_i, Q_k)$ using (1) and (2), respectively; // $Q_k \in \mho$
6          compute angle $\theta_{i,k}$ of line $P_iQ_k$ by $\theta_{i,k} = \arctan \frac{\Delta D_e(P_i,Q_k)}{\Delta D_d(P_i,Q_k)}$;
7      derive the sharpness $S_i$ of $P_i$ using (3);
```

The proposed scheme detects sharp corners/edges on an object using a small-root heapsort-based algorithm. The small-root heap is built using a modified heap build procedure, i.e., Modified-BUILD-MIN-HEAP, which operates as follows. The heap is first constructed by sorting all pixels of a given depth map in ascending order of the pixel depth. The heap maintains the pixel of the smallest depth at the root. The heap is then adjusted by comparing the depth of pixels starting from the root against a predefined dangerous distance threshold and removing the pixels whose depth are larger than the threshold. If the depth of a pixel being compared is beyond the dangerous distance, the comparison process terminates, and all the remaining pixels are removed from the heap. This is because the pixel of the small-root heap that is being processed has the smallest depth distance as compared to the remaining unchecked pixels.

The proposed small-root heapsort-based algorithm operates as follows. Line 1 initializes set Φ to $\varnothing$. Line 2 selects the pixels whose depth are less than a predefined dangerous distance threshold D_1 from set Ψ to create a small-root heap using Ω = Modified-BUILD-MIN-HEAP(Ψ). Lines 3–20 describe the process to recognize sharp corners/edges. Line 4 introduces a $flag$ to indicate if a pixel is a sharp corner/edge pixel or not. In each round of iteration, $flag$ is initialized to 0. If the Euclidean distance between two pixels is shorter than the minimum Euclidean distance threshold ϵ, the two pixels are considered duplicate pixels and can be represented by a single pixel. Lines 5–9 filter out the duplicate pixels of P_j. For each detected sharp corner/edge P_j in set Φ, the Euclidean distance $\Delta D_e(P_i, P_j)$ between P_i and sharp corner/edge P_j is derived (line 7). If the Euclidean distance is not greater than ϵ, P_i is a duplicate pixel of P_j, and $flag$ is set to 2 (lines 8–9).

Lines 10–17 examine whether P_i is a sharp corner/edge pixel by comparing the depth value of the pixel with that of neighboring pixels in circle $C(P_i, R)$. A temporary set $\mho$ is used to store the pixels in circle and is initialized to $\varnothing$ (lines 11–12). If the depth $P_{i,d}$ of P_i is larger than that of the neighboring pixel $Q_k \in \mho$, P_i is not a sharp corner/edge pixel (lines 14–15). If the depth $P_{i,d}$ of P_i is smaller than those of all neighboring pixels $Q_k \in \mho$, P_i is a sharp corner/edge, and the $flag$ is set to 1 (lines 16–17). The P_i is then added to set Φ from heap Ω, which is, in turn, updated using procedure MIN-HEAPIFY(Ω) to maintain the small-root heap property (lines 18–20).



RGB-D Camera
$Q_{k,d}$
P_i'
Q_k
Object
Image
Q_k
$P_{i,d}$
P_i
P_i
$P_iQ_k = \Delta D_e(P_i, Q_k)$
(a)
P_i'
$\Delta D_d(P_i, Q_k)$
Q_k
$\Delta D_e(P_i, Q_k)$
$\theta_{i,k}$
$\theta_{i,k}$
P_i
$\tan\theta_{i,k} = \frac{P_iP_i'}{P_i'Q_k} = \frac{\Delta D_e(P_i,Q_k)}{\Delta D_d(P_i,Q_k)}$
Thus, $\theta_{i,k} = \arctan\frac{\Delta D_e(P_i,Q_k)}{\Delta D_d(P_i,Q_k)}$
(b)

Fig. 2. Illustration of the slope of line P_iQ_k and angle $\theta_{i,k}$.

The sharpness of corner/edge pixel P_i is defined as the average of angles $\theta_{i,k}$ for line P_iQ_k, where Q_k is neighboring pixels of P_i in circle $C(P_i, R)$. Thus. the sharpest corner/angle is the one with minimum average angles. Fig. 2 illustrates how to derive the sharpness of corner/edge pixel P_i. Assume Q_k is a neighboring pixel of P_i captured by an RGB-D camera from an object. As shown in Fig. 2(a), the depths of P_i and Q_k are $P_{i,d}$ and $Q_{k,d}$, respectively. The depth difference and Euclidean distance of the two pixels are $\Delta D_d(P_i, Q_k)$ and $\Delta D_e(P_i, Q_k)$, respectively. Fig. 2(b) shows a snapshot of Fig. 2(a). It is clear that the slope of line P_iQ_k and the corresponding angle $\theta_{i,k}$ are derived as $\tan(\Delta D_e(P_i, Q_k)/\Delta D_d(P_i, Q_k))$ and $\arctan(\Delta D_e(P_i, Q_k)/\Delta D_d(P_i, Q_k))$, respectively. Therefore, the sharpness of corner/edge P_i, which is denoted by S_i, is

$$S_i = \frac{\sum_{k=1}^{\text{size of}(\mho)} \theta_{i,k}}{\text{size of}(\mho)}. \tag{3}$$

The sharpness of detected corners/edges is calculated in Algorithm 2. Inputs are sharp corners/edges detected in Algorithm 1, and outputs are their sharpness. Lines 1–7 iteratively compute the sharpness of each pixel in set Φ. In each round of iteration, a temporary set $\mho$ is used to store neighboring pixels of P_i (lines 2–3). For each pixel Q_k in $\mho$, the Euclidean distance $\Delta D_e(P_i, Q_k)$ and depth difference $\Delta D_d(P_i, Q_k)$ between the concerned pixel P_i and its neighboring pixel Q_k in $C(P_i, R)$ are derived using (1) and (2), respectively (line 5). Then, the slope of line P_iQ_k is derived, and the corresponding angle is obtained using $\theta_{i,k} = \arctan(\Delta D_e(P_i, Q_k)/\Delta D_d(P_i, Q_k))$ (line 6), as illustrated in Fig. 2. It is clear that the P_i associated with a pixel $Q_k \in \mho$ results in an angle away from the horizontal. When all associated angles of pixels in $\mho$ are computed, the sharpness S_i of corner/edge P_i is derived (line 7). The iteration stops when the sharpness of all detected sharp corners/edges have been derived.

B. Compute Approaching Speed and Model Threat

The detected sharp corners/edges on preprocessed depth maps can be used to compute the approaching speed of a mobile manipulator. Assume P_i and P_j are two pixels of a single point on an object in two consecutive preprocessed depth maps, then the approaching speed v is calculated as

$$v = \frac{P_{i,d} - P_{j,d}}{\Delta t} \tag{4}$$

where $P_{i,d}$ and $P_{j,d}$ are the respective depths of pixel P_i and P_j, which are captured by the RGB-D camera at time instance t and $t + \Delta t$, respectively. Therefore, the key to obtaining the approaching speed is to identify sharp corner/edge pixels of a single point on an object in two consecutive depth maps.

Gray-level histogram and affine moment invariants (AMIs) are employed to identify sharp corner/edge pixels of a single point on an object in two consecutive depth images. The property that static images of an object captured at different time instances have similar gray-level histograms [12] can be used to identify in two depth images the sharp corner/edge pixels of a single point on the object, which can be further utilized to derive the speed of the moving RGB-D camera system. However, due to the mobility of the camera system, images taken at different time instances may be translated, enlarged, scaled, or rotated, which leads to a different gray-level histogram for the same object and hence undermines the rationale of using gray-level histogram to derive the speed. Thus, the AMIs [13] that can characterize an image under the general affine transformation is then exploited to solve the aforementioned problem.

Here, we proposed a comprehensive recognition scheme to identify in two depth images the same sharp corner/edge pixels of an object. This is achieved by combining the gray-level histogram and the AMI-based technique. The proposed scheme first filters out the dissimilar sharp corners/edges of the object in two depth maps by detecting their gray-level histogram similarity via cosine similarity [14]. It then uses the more precise yet complex AMI-based technique to further validate if the corners/edge pixels of similar gray-level histogram are pixels of a single point on the object.

Before detecting the histogram similarity between two sharp corners/edges in terms of gray-level distribution, the gray value of pixels in the image needs to be acquired first, which is achieved by scaling the RGB value of pixels to the range of [0, 255]. Given two sharp corner/edge pixel P_i and P_j with neighboring pixels inside the respective circle $C(P_i, R)$ and $C(P_j, R)$, two eigenvectors $\mathbf{a}$ and $\mathbf{b}$ are then introduced to demonstrate the distribution of gray values of pixels in the circles over the range [0, 255]. Finally, cosine similarity [14] is adopted to examine the similarity between $\mathbf{a}$ and $\mathbf{b}$. It detects the similarity of two vectors by measuring the cosine of the angle in between.

Let $\cos(\mathbf{a}, \mathbf{b})$ denote the cosine similarity between eigenvectors $\mathbf{a}$ and $\mathbf{b}$, then it is given by

$$\cos(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|} = \frac{\sum_{\zeta=1}^{255} a_\zeta \times b_\zeta}{\sqrt{\sum_{\zeta=1}^{255} (a_\zeta)^2} \times \sqrt{\sum_{\zeta=1}^{255} (b_\zeta)^2}}. \tag{5}$$

If the value of $\cos(\mathbf{a}, \mathbf{b})$ is not less than a predefined threshold, the sharp corners/edges P_i and P_j are assumed to be similar in terms of histogram, then the feature similarity is checked using the AMI-based technique. Otherwise, the detection of similarity for P_i and P_j stops, and the two pixels are deemed to be sharp corners/edges of different points on the object.

The AMI-based technique first tries to derive the AMIs of the image consisting of a given sharp corner/edge and corresponding neighboring pixels. Then based on three basic AMIs $(I_1 - I_3)$ [13] for feature extraction, an integrated AMI is built to characterize the image consisting of a given sharp corner/edge and corresponding neighboring pixels under the general affine transformation, as shown in the following:

$$I = C_1 I_1 + C_2 I_2 + C_3 I_3 \tag{6}$$

where C_1, C_2, C_3 are weights and hold for $C_1 + C_2 + C_3 = 1$.

By jointly considering the gray-level distribution and AMI feature, the similarity between two sharp corners/edges P_i and P_j is then defined as

$$\text{Sim}(P_i, P_j) = \frac{\cos(\mathbf{a}, \mathbf{b})}{\|I(P_i) - I(P_j)\|} \tag{7}$$

where $\|I(P_i) - I(P_j)\|$ is the integrated affine invariant moment difference between P_i and P_j.

The calculation of the approaching speed is therefore described as follows. Assume two preprocessed depth maps M_1 and M_2 captured at different time instances. P_i is one detected sharp corner/edge pixel in M_1, and P_J is the sharp corner/edge pixel that is most similar to P_i among those in M_2, where $\text{Sim}(P_i, P_J) = \max\{\text{Sim}(P_i, P_j) | \forall j \in [1, N]\}$, and N is the number of detected sharp corners/edges in M_2. If $\text{Sim}(P_i, P_J)$ is no less than a predefined threshold, P_i and P_J are considered the same sharp corner/edge pixels of a single point, and the approaching speed can thus be derived using (4).

A danger coefficient that indicates the risk level of detected sharp corners/edges is modeled and can be used to activate appropriate countermeasures accordingly. It not only depends on the depth and sharpness of the corner/edge but also relies on the approaching speed of the mobile manipulator, i.e.,

$$W_i = \frac{v}{P_{i,d} S_i}. \tag{8}$$

III. Experimental Results and Discussions

Extensive experiments have been carried out to validate the proposed detection and evaluation scheme. Our method is tested using color images and synchronized depth maps from data sets [15]. Due to the inherent mixed, lost, and noisy

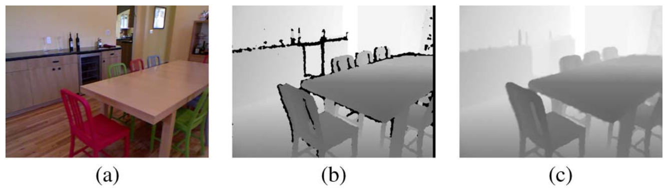


Fig. 3. Example of depth map recovery via the technique in [9]. (a) Input color image. (b) Input depth map. (c) Recovered depth map.

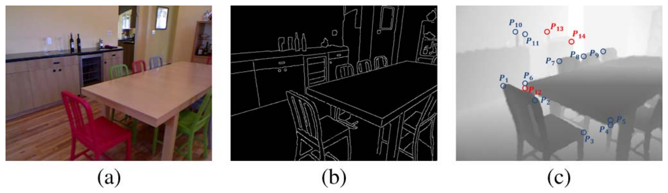
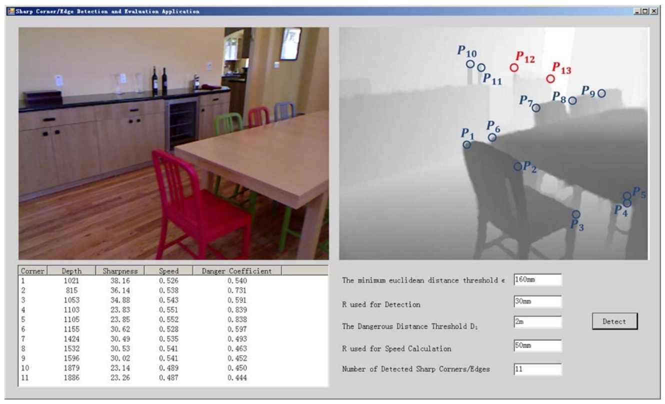


Fig. 4. Result for RGB-D image *Living Rooms* [15] at t_0. (a) Input color image at t_0. (b) Edge map at t_0. (c) Labeled depth map at t_0.



Corner	Depth	Sharpness	Speed	Danger Coefficient
1	1021	38.16	0.526	0.540
2	815	36.14	0.538	0.731
3	1053	34.88	0.543	0.591
4	1103	23.83	0.551	0.839
5	1105	23.85	0.552	0.838
6	1155	30.62	0.528	0.597
7	1424	30.49	0.535	0.493
8	1532	30.53	0.541	0.463
9	1596	30.02	0.541	0.452
10	1879	23.14	0.489	0.450
11	1886	23.26	0.487	0.444

The minimum euclidean distance threshold ϵ: 160mm
R used for Detection: 30mm
The Dangerous Distance Threshold D_1: 2m
R used for Speed Calculation: 50mm
Number of Detected Sharp Corners/Edges: 11

Fig. 5. Parameters for detected sharp corners/edges at $t_0 + \Delta t$.

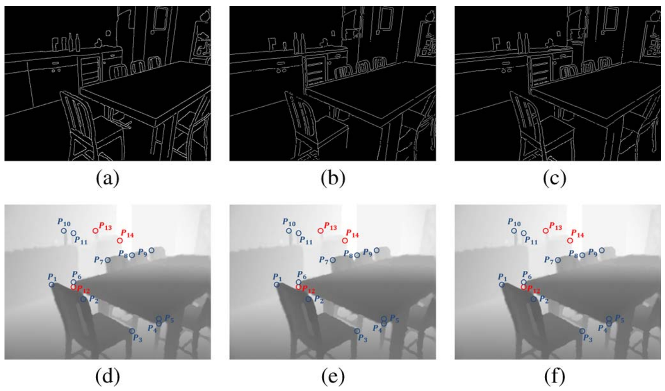


Fig. 6. Sharp corner/edge identification for *Living Rooms* [15] at t_0 using 3 edge detection techniques. (a) Canny. (b) MVD. (c) RCMG-MM. (d) Detection results for (a).

pixel problems of RGB-D cameras, the depth image filtering technique [9] is first utilized to preoptimize the input depth map. An example of depth map recovery is shown in Fig. 3. The resolution of the input color image and depth map is 640×480. During the preoptimization, thresholds T_1 and T_2 used for CDT map generation are set to 18 and 54, and the local analysis window size used for median filtering is 5×5 [9].

The proposed scheme then recognizes sharp corners/edges and calculates their sharpness using Algorithms 1 and 2, respectively. In the two algorithms, the radius R, threshold D_1, and threshold ϵ are set to 30 mm, 2 m, and 160 mm, respectively. In the calculation of the approaching speed, radius R is set to 50 mm. The threshold used in similarity checking of $\cos(\mathbf{a}, \mathbf{b})$ is set to 0.5, and the threshold used in similarity checking of $\mathrm{Sim}(P_i, P_J)$ is set to 0.8. The weighting constants C_1–C_3 are all assumed to be 1/3, and the time interval Δt is set to 0.2 s.

It is noteworthy that an RGB-D camera has a practical ranging limit of 0.5–5 m distance [10], which implies that the manipulator equipped with an RGB-D camera has to move at low speed to respond in time to external stimulus (e.g., sharp corners/edges). In other words, the proposed algorithm running on a slow-moving manipulator aims to identify sharp corners/edges on objects in indoor environments. In this way, experimental parameters for this scenario are specified and given as above. In addition, a corner/edge is deemed to be dangerous from the perspective of the manipulator when the corner/edge is pointing to the manipulator and the angle is less than a certain threshold value (e.g., 120°).

The RGB-D image *Living Rooms* [15] is selected as the input to the proposed algorithm. The results for the image at t_0 are shown in Fig. 4. Specifically, the input color image and edge map are given in Fig. 4(a) and (b), and the detected sharp corners/edges in recovered depth map are given in Fig. 4(c). According to the definition of sharp corner/edge, there are 14 candidate sharp corners/edges in *Living Rooms*. Fig. 4(c) shows that 11 sharp corners/edges $(P_1–P_{11})$ are recognized, whereas 3 sharp corners/edges $(P_{12}–P_{14})$ are left out, leading to a recognition ratio of 78.6%. As described in Section II-B, the approaching speed can be derived by identifying the same sharp corners/edges in two consecutive images, and the potential threats of detected sharp corners/edges to the mobile manipulator can be evaluated by considering the approaching speed, depth, and sharpness. Fig. 5 shows the parameters for detected corners/edges in *Living Rooms* at $t_0 + \Delta t$.

Fig. 6 shows the identification of sharp corners/edges using the proposed method for *Living Rooms* image [15] under different edge detection techniques, including Canny [11], MVD [4], and RCMG-MM detectors [4]. Fig. 6(a)–(c) plots the edge information extracted using the three techniques, and Fig. 6(d)–(f) presents the respective sharp corner/edge recognition results. For *Living Rooms* image under test, results indicate that the proposed scheme is also effective when different edge detectors are employed.

When we compute the approaching speed, the similarity between two succeeding images is evaluated based on the gray-level histogram difference and the proposed integrated AMI difference. The gray-level histogram difference is assessed using cosine similarity [14] for simplicity. Other typical histogram-based methods (e.g., histogram intersection technique [17]) also apply to similarity checking. We compare the proposed scheme with a benchmarking method that combines the histogram intersection method and the integrated AMI technique. One hundred RGB-D images from [15] are used for this comparison, and Fig. 7 demonstrates that the two methods have a close recognition ratio. The recognition ratios of the two methods are both in the approximate range of [0.751, 0.849], the recognition ratio differences are all below 0.1, and the average is 0.042.

The proposed scheme is compared with the pruned corner detection (PCD) [16] in terms of sharp corner/edge recognition

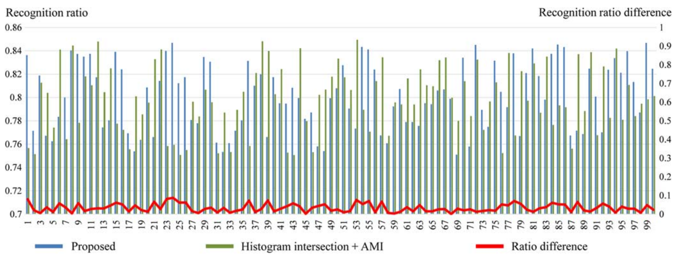


Fig. 7. Compare the recognition ratio of the proposed scheme with the benchmarking method (Histogram intersection [17] + AMI).

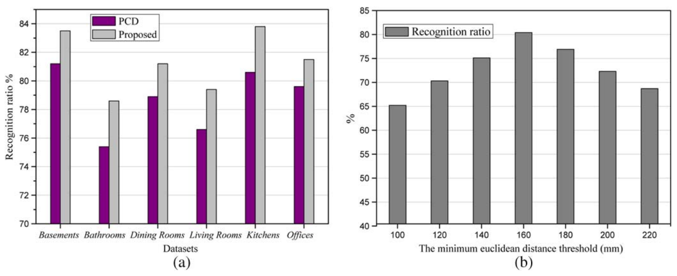


Fig. 8. Recognition ratio tested by six data sets (under varying ϵ). (a) Recognition ratio tested by six data sets. (b) Recognition ratio under varying ϵ.

ratio. The comparison study is conducted using six RGB-D image data sets [15] of indoor environments, including *Basements*, *Bathrooms*, *Dining Rooms*, *Living Rooms*, *Kitchens*, and *Offices*. Fig. 8(a) gives the average recognition ratio for the six data sets under the proposed scheme and PCD [16]. The figure shows that the average recognition ratio of the proposed scheme exceeds that of method PCD [16] by up to 3.5%. This is because method PCD [16] only processes a small portion of sharp corner/edge candidates rather than all candidate corners/edges on the entire image. In addition, it does not take advantage of depth data to improve the recognition ratio.

The impact of the minimum Euclidean distance threshold ϵ on ratio of identifying sharp corners/edges is also investigated in the experiments. One hundred twenty different images from [15] are selected for the investigation. Fig. 8(b) shows that the average recognition ratio starts with 65.2% and grows up to 80.4% when threshold ϵ increases from 100 to 160 mm, then starts with 80.4% and drops to 68.7% when ϵ increases from 160 to 220 mm. This is because in the proposed algorithm, when ϵ is low, a sharp corner/edge and its close-by sharp corners/edges could be considered the same sharp corner/edge, which leads to a degradation of the recognition ratio. Similarly, when ϵ is high, some sharp corners/edges could be omitted, and the recognition ratio is thus decreased. Therefore, the proper setting of ϵ is critical, and $\epsilon = 160$ mm that generates the highest recognition ratio is selected in the experiments to validate the proposed scheme.

To check real-time feasibility of the proposed scheme, the running time of the proposed scheme needs to be measured. The Linux built-in function *gettimeofday()* that can get the current time at microsecond level is called when the program of the proposed scheme starts and ends. The running time of the program is thus derived as the difference between the end time and the start time. The results averaged over 1000 runs show that it takes about 1.75 s to detect and evaluate sharp corners/edges on RGB-D images captured at two succeeding time instances. Since the dangerous distance threshold is 2 m, the maximum ranging limit of Kinect is 5 m [10], and the measured approaching speed is about 0.5 m/s (see Fig. 5), the response time left for the manipulator ranges from 2.25 s to 8.25 s which is enough for the slow-moving manipulator to avoid collision. Thus, the proposed scheme is practicable to deploy on a slow-moving manipulator that aims to identify sharp corners/edges on objects in domestic environments in real time.

IV. Conclusion

We have proposed a sharp corner/edge detection and evaluation scheme for RGB-D camera-based domestic applications. The proposed algorithm recognizes sharp corners/edges, derives the corresponding sharpness, and assesses the potential threat. Simulation results demonstrated that the proposed scheme can identify up to 84.9% of visible dangerous corner/edges in a scenario.

References

[1] J. Han, L. Shao, D. Xu, and J. Shotton, “Enhanced computer vision with Microsoft Kinect sensor: A review,” *IEEE Trans. Cybern.*, vol. 43, no. 5, pp. 1318–1334, Oct. 2013.
[2] M. Basu, “Gaussian-based edge-detection methods—A survey,” *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 32, no. 3, pp. 252–260, Aug. 2002.
[3] V. Torrem and T. A. Poggio, “On edge detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, no. 2, pp. 147–163, Feb. 1986.
[4] A. Mittal, S. Sofat, and E. Hancock, “Detection of edges in color images: A review and evaluative comparison of state-of-the-art techniques,” in *Autonomous and Intelligent Systems*. Berlin, Germany: Springer-Verlag, 2012, pp. 250–259.
[5] P. Melin, C. I. Gonzalez, J. R. Castro, O. Mendoza, and O. Castillo, “Edge-detection method for image processing based on generalized type-2 fuzzy logic,” *IEEE Trans. Fuzzy Syst.*, vol. 22, no. 6, pp. 1515–1525, Dec. 2014.
[6] E. Herbst, X. Ren, and D. Fox, “RGB-D object discovery via multi-scene analysis,” in *Proc. IEEE Int. Conf. Intell. Robot. Syst.*, 2011, pp. 4850–4856.
[7] L. Bo, X. Ren, and D. Fox, “Unsupervised feature learning for RGB-D based object recognition,” in *Proc. Int. Symp. Exp. Robot.*, 2013, pp. 387–402.
[8] M. Blum, J. T. Springenberg, J. Wulfing, and M. Riedmiller, “A learned feature descriptor for object recognition in RGB-D data,” in *Proc. IEEE Int. Conf. Robot. Automat.*, 2012, pp. 1298–1303.
[9] S. Kim, M. Kim, and Y.-S. Ho, “Depth image filter for mixed and noisy pixel removal in RGB-D camera systems,” *IEEE Trans. Consum. Electron.*, vol. 59, no. 3, pp. 681–689, Aug. 2013.
[10] Microsoft Kinect. [Online]. Available: https://www.microsoft.com/en-us/kinectforwindows/meetkinect/features.aspx
[11] J. Canny, “A computational approach to edge detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.
[12] S. Lefevre, J. Holler, and N. Vincent, “A review of real-time segmentation of uncompressed video sequences for content-based search and retrieval,” *Real-Time Imag.*, vol. 9, no. 1, pp. 73–98, Feb. 2003.
[13] S. Flusser and T. Suk, “Pattern recognition by affine moment invariants,” *Pattern Recognit.*, vol. 26, no. 1, pp. 167–174, Jan. 1993.
[14] Cosine. [Online]. Available: http://en.wikipedia.org/wiki/Cosine_similarity
[15] Dataset. [Online]. Available: http://cs.nyu.edu/~silberman/datasets/
[16] N. Ramakrishnan, M. Wu, S.-K. Lam, and T. Srikanthan, “Enhanced low-complexity pruning for corner detection,” *J. Real-Time Image Process.*, pp. 1–17, Jan. 2014.
[17] U. Gargi and R. Katsuri, “An evaluation of color histogram-based methods in video indexing,” in *Proc. Int. Workshop Image Databases Multimedia Search*, 1996, pp. 75–82.