Multimodal segmentation of dense depth maps and associated color information

Maciej Stefańczyk[1] and Włodzimierz Kasprzak[1]

Warsaw University of Technology, Institute of Control and Computation Eng.
Nowowiejska 15/19 00-665 Warsaw, Poland
stefanczyk.maciek@gmail.com, w.kasprzak@elka.pw.edu.pl,
http://robotics.ia.pw.edu.pl

Abstract. An integrated segmentation approach for color images and depth maps is proposed. The 3D pointclouds are characterized by normal vectors and then grouped into planar, concave or convex faces. The empty regions in the depth map are filled by segments of the associated color image. In the experimental part two types of depth maps are analysed: generated by the MS-Kinect sensor or by a stereo-pair of cameras.

Keywords: depth map, integrated image segmentation, surface segmentation, 3D point clouds

1 Introduction

There is a long history of 3D data acquisition technology, ranging from stereovision, laser scanners to structured light processing techniques [1], [2]. But only when Microsoft's project Natal (now known as Kinect) became reality, the use of depth maps (or now more often called as 3D pointclouds) has became more and more popular on many fields, including computer entertainment, robotics, etc. [3]. This paper focuses on segmentation methods of dense depth maps produced by such a sensor, and using an RGB image aligned with it. Our aim is to use the segments extracted by the proposed method as the input for a model-based 3D object recognition system, with applications in service robotics [1].

In section 2 some methods of gathering pointclouds are described and a synthetic comparison of them is given. In next section, a method of enhancing the raw depth map by additional information - the normal vector per point - is proposed. Section 4 presents the actual segmentation process, that makes use of many modalities (depth, normal vector, color, etc.). Then section 5 describes various features that can be calculated for obtained segments, the 3D point subsets. The explanation of approach is in-place illustrated by experimental results. The paper ends with a summary.

2 Acquisition of pointcloud data

All methods of gathering 3D scene information can be divided into groups, depending on used hardware – from cheap, highly-available cameras, through those supported by specialized projectors, to the dedicated, highly-sophisticated hardware and laser scanners. Another criterion is the method of information acquisition – a passive registration of light from environment or the usage of active scene illumination. This comparison focuses on practical features of three presented approaches – one that uses an active distance sensor, a second one that uses a stereo-pair of cameras and a third one based on structured light illumination.

The first approach is using traditional distance sensors, such as rangefinders or laser scanners [1]. Mounting them on pan-tilt rotatable units allows the acquisition of a series of single scans, that can be merged to form a full 2D depth map. Precision ranges from single millimeters to centimeters, and depends on the quality of applied sensors and the precision of pan-tilt units. This method is rather slow, as a single scan of the whole scene can take up to few seconds. Very often, there is a need for an additional color camera to get a color image of scanned scene.

Another method is stereo vision, an image processing approach that uses two cameras. The depth of a point in space in front of them is calculated while computing the disparity between its two projections in both images. Most of the stereo-matching algorithms are based on point features detected in the images, so they behave well for outdoor scenes, where many natural characteristic points can be found. In interiors, where many objects with homogeneous surfaces exist (like walls and furniture fronts), there is a need for some active illumination to help the stereo-matching algorithm [2]. The generated depth map is usually automatically aligned with one of the images, so there is no need for separate color image acquisition. Also, this method can be pretty cheap, even classical web cameras can give good results. Drawbacks of this method are relatively high computational load and moderate framerates.

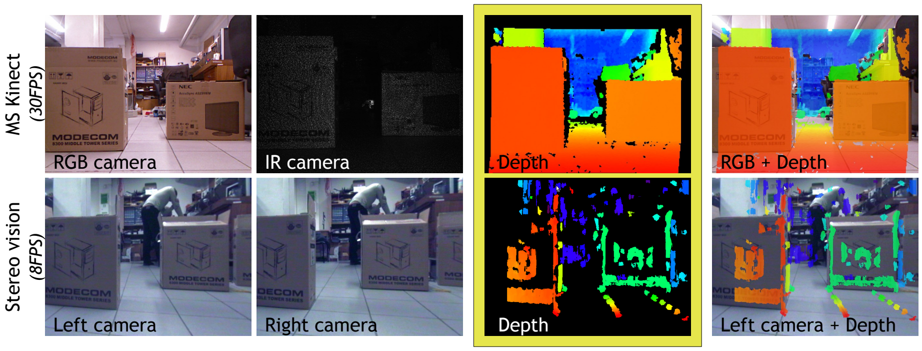

Fig. 1. Comparison of depth maps generated from MS Kinect and stereo vision algorithm from OpenCV library

Another type of approaches that requires strong image analysis is based on structured light illumination. A camera observes the scene, which is illuminated by a well-known light pattern, and the depth map is created by pattern deformation analysis. This method actively uses illumination, so it is well suited for indoor environments, but in outdoor scenes, with high sun exposure, it usually fails. In cases where projected pattern uses visible light, the same camera can be used for depth and color registration, so both maps are perfectly aligned. On the other hand, using visible light can be uncomfortable if working in common space with people, so some infrared projectors and cameras are used instead. In this case, there are at least two cameras needed, and a separate step for aligning depth and color images is necessary. One of popular sensors using structured light is Microsoft Kinect, which was quickly hacked by open-source community [3]. This sensor was chosen for this work, because nearly the whole acquisition processing is done in hardware and the scene coverage is very high (sample images can be seen on fig. 1). We have also implemented a stereo-vision approach, but it is working much slower than Kinect, with an acquisition rate of 8 frames per second only.

Table 1. Comparison of selected methods of generating 3D scene image

Method	**CPU load**	**Acquisition time**	**Resolution**	**I/O**[a]	**HW cost**
2D sensors	moderate	1s-3s	180x256px	●/●	˜1000USD
Stereo vision	high	0.1s-1s	640x480px	◐/●	˜100USD
Proj. texture stereo	high	0.1s-1s	640x480px	●/◐	˜400USD
MS Kinect	low	0.03s	640x480px	●/○	˜100USD
TOF cameras	low	0.01s-0.04s	176x144px	●/○	˜8000USD

[a] I - works well indoors, O - works well outdoors

Finally, the last type of 3D data acquisition approaches is using time-of-flight cameras [4], where the light is emitted from a projector and the arrival time (or light phase) is measured after it bounces back from objects on the scene. Again, to get an additional color image information there is a need for another camera and a depth map-to-image registration step. A summary of described methods and their main features is presented in table 1.

3 Extending the depth data

Let us observe, that to rely only on the raw depth map could be insufficient in many applications. For example, a box located on a desk could be left undetected while using the depth map, because no real depth discontinuity is observed. Therefore, from this depth map some more information of different modality need to be extracted.

3.1 Normal vector map

The most straightforward extension is to obtain the normal vector map – at each depth map element we estimate a normal vector while taking into account the local neighborhood of given point in the depth map. The depths in the neighborhood are assumed to sample the unknown surface. Most often two kinds of algorithms are applied: based on numerical optimizations or on Voronoi diagrams [5]. The algorithms may work either on raw depth map (with distance from camera to object held in each pixel), or on its pointcloud representation (3D coordinates in some fixed frame held in each pixel). There are many challenges associated with normal estimation, such us handling the measurement noise and discontinuities in depth. For example, to avoid generating false normals indicating presence of wall, see fig 2. Using normal vector maps, objects mentioned earlier can be easily segmented – even if there are no discontinuities in depth, there will be visible edges in the normal image.

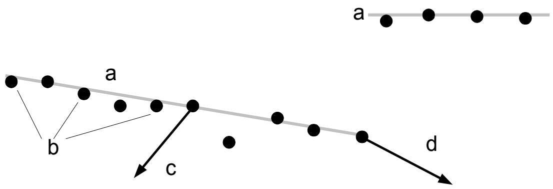

Fig. 2. Some caveats of estimating normals from sampled points; a - parts of real surfaces, b - sampled points, c - incorrect normal due to presence of noise, d - incorrect normal on edge of surface

3.2 Surface curvature

Another modality that extends the depth map is the curvature factor of underlying surface [6]. This kind of information can be directly computed from the raw depth map. It can greatly enhance the detection of fuzzy boundary points, where the normal vectors show approximately continuous distribution, but where different surfaces cross (for example a bottle could be segmented into it's cylinder-like main part and sphere-like top, fig. 4a,b). There are of course visual differences between mentioned maps, that can be easily observed when looking at the image containing flat surfaces and spheres. In the normal vector map, both object types are filled with some gradient (in case of flat surface not parallel to camera plane), but it's hard to tell whether surface is really flat or distinguish a sphere from a cone (fig. 4c). Every flat surface, on the other hand, is filled with uniform color on normal maps (and curvature maps), independent of it's orientation (normal vectors of surface are the same in it's every point). Also sphere-like regions can be easily distinguished from cone-like regions in the depth map, but to tell exactly, where the border between sphere and cylinder is one must look at the curvature map (fig. 4d).

4 The pointclouds segmentation approach

4.1 Similarity of points

Our segmentation approach is based on a region growing technique, but instead of working on a single intensity image, we feed it with all the image maps mentioned earlier (hence the name of this approach – *multimodal segmentation*). These input maps are aligned with each other, so pixels can be immediately sampled from all of them when segments are created. Because of using multiple images as input, similarity function has to be expanded accordingly.

Another modification of the classical region-growing segmentation is that in our approach points can be compared not only to mean values of already segmented ones, but also only to border pixels (pixels are compared only to currently considered ones, to be precise). This allows to detect slowly changing surfaces as single segments, that would be broken into smaller parts by a standard comparison criterion w.r.t. the mean value.

Our point similarity function (1) has four parameters, defining how close each of components has to be in both pixels, T_R, T_D, T_N and T_C, where R stands for normalized RGB, D for depth, N for normal vector and C for curvature, respectively:

$$S(p_x, p_y) = FUN\left(\frac{d_R(p_x, p_y)}{T_R}, \frac{d_D(p_x, p_y)}{T_D}, \frac{d_N(p_x, p_y)}{T_N}, \frac{d_C(p_x, p_y)}{T_C}\right) \quad (1)$$

The final value can be calculated by substituting FUN with either *sum* or *max* function, each giving slightly different results. Distances are calculated by associated d functions. Color comparison is done by calculating Euclidean distance between the normalized RGB values (i.e. R/L, G/L, B/L, where L means luminance) of both points, and T_R is the maximum allowed difference.

The location distance of points is calculated also as a Euclidean distance between their 3D coordinates, and T_D is expressed in meters. Normal vectors could be treated similarly to maps mentioned earlier, but it will be counterintuitive, so we calculate the angle between both vectors (2) using their dot product:

$$d_N(p_x, p_y) = \arccos(normal(p_x) \cdot normal(p_y)) \quad (2)$$

Observe, that both vectors are already normalized, so it's unnecessary to divide the distance by their length. T_N is then given in degrees, which is much more convenient.

The last component in (1) is a curvature, which is expressed as radius of sphere fitted to the point's local neighborhood. Hence, a simple comparison of curvature is needed only, like comparing two numbers.

4.2 Processing pipeline

The pointclouds segmentation processing pipeline is shown on fig. 3 (the blocks represent components, the arrows represent data flows between the components). Along with the real data acquired by the Microsoft sensor, there is also a component for generating test scenes, consisting of some simple shapes, with a possibility of adding synthetic noise. This component is applied for testing whether algorithms works as expected on (almost) perfect data.

The segmentation approach has been implemented in DisCODe data processing framework [7], using the OpenCV library and the data acquisition module of the Kinect device.

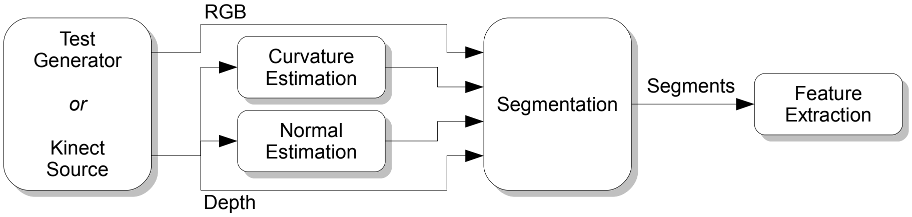

Fig. 3. The proposed segmentation processing pipeline

5 Multidimensional segment descriptors

The usability of raw segments, without having any specific information about them, is rather low. That's why we add another step to the processing pipeline – the feature extraction step. Apart from using widely known image moments [8], there are also popular features that are based on histograms of geometric distances from some fixed point (like center of mass) [9].

We propose another approach to feature extraction, that uses not only point coordinates, but also makes use of other information, their normal vectors specifically. For every point in given segment we calculate the angle between its normal vector and vector from this point to segment center of mass. All angles are accumulated and the mean value along with standard deviation are calculated. Based on those two factors we can classify a segment into planar (when the mean angle is close to 90°, green color in fig. 4d), concave (mean is lower than $90°-\epsilon$, blue color in fig. 4d) or convex surface (mean is larger than $90°+\epsilon$, red color in fig. 4d). Using the standard deviation value one can decide if the segment has proper parameter values – a high deviation value means that the current pointcloud set has not been sufficiently decomposed into segments and that it has to be further split.

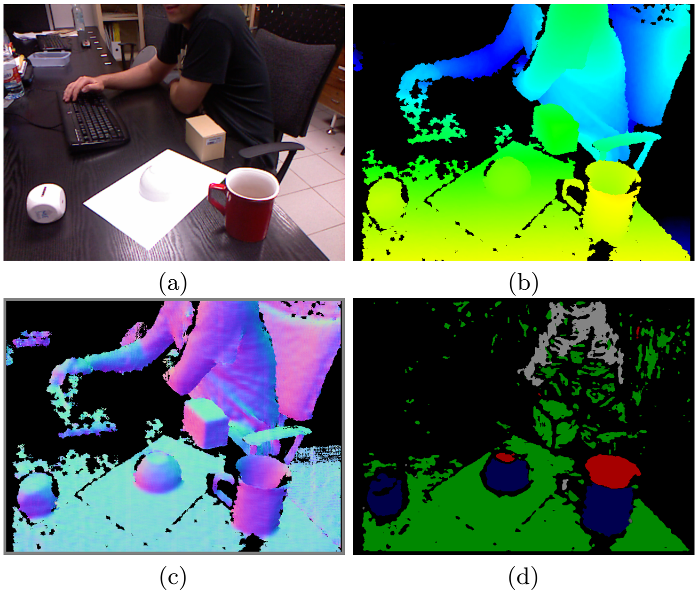

Fig. 4. Illustration of the pointcloud segmentation process: (a) example image, (b) the raw depth map, (c) normal vector map, (d) segmented point clouds

6 Summary

We have proposed an integrated segmentation approach for dense depth maps and associated color images. Differently than in a typical Delaunay triangulation approach the raw 3D pointclouds are first characterized by their normal vectors, estimated on base of the point's local neighborhood. In the segmentation process, the subsets of 3D points are easily grouped into planar, concave and convex faces. In future work, we are going to implement a more detailed surface patch classification step that is based on the superquadrics approach to surface modeling [10]. The empty regions in the depth map are assumed to belong to a distant background and they are filled by segments of the associated color image. This step can also be enhanced in the future by assigning texture patterns, detected in the color image, to the surfaces, detected for 3D pointclouds.

References

1. Surmann, H., Nüchter, A., Hertzberg, J.: An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments. Robotics and Autonomous Systems **45**(3) (2003) 181–198
2. Konolige, K.: Projected texture stereo. In: Robotics and Automation (ICRA), 2010 IEEE International Conference on, IEEE (2010) 148–155
3. Giles, J.: Inside the race to hack the Kinect. The New Scientist **208**(2789) (2010) 22–23

4. Lange, R., Seitz, P.: Solid-state time-of-flight range camera. Quantum Electronics, IEEE Journal of **37**(3) (2001) 390–397
5. Dey, T., Li, G., Sun, J.: Normal estimation for point clouds: A comparison study for a Voronoi based method. In: Point-Based Graphics, 2005. Eurographics/IEEE VGTC Symposium Proceedings, IEEE (2005) 39–46
6. Miao, Y., Feng, J., Peng, Q.: Curvature estimation of point-sampled surfaces and its applications. Computational Science and Its Applications–ICCSA 2005 (2005) 441–451
7. Kornuta, T., Stefańczyk, M.: DisCODe: component-oriented framework for sensory data processing (PL). Measurements, Automation and Robotics **15**(6) (2012) accepted for print.
8. Giordano, P., De Luca, A., Oriolo, G.: 3D structure identification from image moments. In: Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on, IEEE (2008) 93–100
9. Mahmoudi, M., Sapiro, G.: Three-dimensional point cloud recognition via distributions of geometric distances. Graphical Models **71**(1) (2009) 22–31
10. Jaklic, A., Leonardis, A., Solina, F.: Segmentation and Recovery of Superquadrics. Volume 20 of Computational imaging and vision. Kluwer, Dordrecht (2000)