Contents lists available at Science-Gate

International Journal of Advanced and Applied Sciences

Journal homepage: http://www.science-gate.com/IJAAS.html

Point Cloud Plane Visualization by Using Level Image

Pavel Chmelar[1], Lubos Rejfek[1], Ladislav Beran[1], Natalija Chmelarova[1], Martin Dobrovolny[1]

[1]Department of Electrical Engineering, Faculty of Electrical Engineering and Informatics, University of Pardubice, Pardubice, Czech Republic
Pavel.Chmelar@student.upce.cz

ARTICLE INFO

Article history:
Received
Received in revised form
Accepted
Available online

Keywords:
Plane visualization
Level image
Point cloud

ABSTRACT

This paper presents the point cloud plane visualization by using the level image. A created level image indicates the point presence in a space at a specific level. By knowledge its origin position in a space, the physical pixel size and the detected rotation angle we can easily visualize planes in an analyzed point cloud. The connection of image processing methods with the physical measurement points offers besides the visualization also obtaining important properties about a scanned space. The described algorithm includes also a color point cloud visualization. Presented results showing the level images advantages for point clouds processing.

Introduction

The point cloud visualization is a method how to present measurement data more interactively. A monotone visualization by a single color allows to present space features like a shape and points position, but there is missing a real visual form.

The recent research in this filed deals with augmented reality visualization (VAR) (Zeng et al., 2017a). A surgical robot is equipped by projector-camera system (PCS) attached to its arm flange. The PCS is used to obtain the surface point clouds of the patient's head for patient-to-image registration. VAR is accomplished by merging the real-time patient's video and the preoperative 3D operational planning model. Next research (Zhang et al., 2017b) describes reconstruction framework for recovering the structure models of space objects using a visible sensor and multi-view images of the target object. Experimental results demonstrate efficient object recovering. The work deals with Dense Visual SLAM (Simultaneously Localization and Mapping) (Yan et al., 2017c) uses Probabilistic Surfel Map (PSM) to align the virtual and real world together in augmented reality applications. The main PSM idea is to maintain a globally consistent map with both photometric and geometric uncertainties. Another research deals with an efficient representation of color points for 3D mapping and visualization (Ryde, 2015a). Textured Voxel-Bounded Planes are used for visualization with impressive results.

The most recent library for a point cloud data processing and a visualization is the Point Cloud Library (PCL), presented in (Rusu et al., 2011). It is still actively developed and often used by many researches. The library contains state of the art algorithms for: filtering; feature estimation; surface reconstruction; registration; visualization; model fitting and segmentation. Also modern GPUs help with the accelerating visualization process (Han et al., 2017d). The OpenGL API is supported by many platforms (Raymond et al, 2015b), (Rusu et al., 2011) and especially for big data sets visualization it is necessary to use the GPU power in case of the normal response preservation to the user, which analyses these data.

For the 3D range scanning we are using autonomous mobile platform, see Fig. 1.

Fig. 1: The rotary 3D range scanning system

As a measurement device is used an optical rangefinder, which consist from a camera and a laser with ability of vertical swapping the laser beam. The measurement principle is based on the triangulation method. The principle of a distance determination and the 3D vector map construction is introduced in the paper (Chmelar et al., 2013). The whole measuring device is pivotally carried by a tripod. The used laser diode has an output power 200 mW, especially for a good recognition after vertical swapping. The described 3D range scanning system is a part of the bigger project called ARES (Autonomous Research Exploration System) (Beran et al., 2015c, Beran et al., 2015d). A resulting 3D point clouds need future processing to achieve important space properties.

In the last paper (Chmelar et al., 2017e) we introduced the method how to extract color information from measurement frames and include in point clouds created by rotary optical rangefinders, shown in Fig. 2. The visualization method is based on template matching algorithms and it uses two measurement steps for the point color determination, because in actual measurement frame a point is covered by the laser line. From one image it is determined measurement point in a space and from the second it is extracted color information. The color information serves only as the point cloud visualization. It serves as nicer presentation of measurement data. Color information can be mainly important for architects but it finds usability in many research areas.

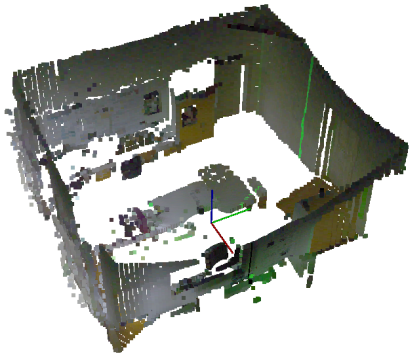

Fig. 2: The colored point cloud of a room

In this paper we would like to present a plane visualization method by the level image application, which is introduced in (Chmelar et al., 2017f). The level image connects the point cloud data with often used image processing methods. This offers to easily observe very important space properties, which can be used for future processing, not only for visualization purposes.

Level image

The level image is a tool for point cloud visualization and processing. Its origin is from point cloud 3D level scanning presented by (Chmelar et al. 2017f). In an analyze cloud it serves as a space points indication in a specific level. One of the main using area for image level is plane visualization. During its construction the origin position $I_{L_{phys}}(x, y)$ and the rotation angle $R(h, v)$, where h and v is a horizontal and a vertical rotation, are stored. By knowledge of the qD parameter, which is the distance space area quantization parameter, we can easily visualize the analyzed space. The knowledge of a physical pixel distance offers many more advantages. Its changing and analyzing by image processing methods offers to get important parameters describing a covered plane like its area, perimeter and total area including holes in point cloud. All methods for getting these parameters are described in (Chmelar et al., 2017g).

Level image construction

The main construction aim is to visualize a plane coverage via a distance quantization parameter qD, which forming point cloud points in a specific level dL acquired by the analyzed point cloud 3D level scanning. The pseudo algorithm is shown in Fig. 3.

```
Function [I_L,(I_Lc)] = PC2ImageLevel(PC, qD, Dim_s)
  Get R for all axis X, Y and Z                (1)
  Get image level size H_IL and W_IL from qD   (2)
  Find real points' position in I_Lx and I_Ly  (3)
  Estimate level image intensity I_L(x,y)      (4)
  Get level image origin I_Lphys(x,y)          (5)
  if(size(PC,2) == 6)
    Create I_Lc from RGB point intensity       (6)
  end
end
```

Fig. 3: The pseudo algorithm of a level image construction

The algorithm needs the parameter scanning dimension Dim_s. It is an input parameter known form 3D level scanning, which precede the image level construction. From the input point cloud it is estimated the range in each axis X, Y and Z according following equation

$$R_A = [min(PC(\forall, A)), max(PC(\forall, A))], \tag{1}$$

where index A is appropriate axis. The selected scanning dimension is the same as from the level image construction and it is presented as the Z axis. Points position from the rest two dimensions are X and Y axis in a level image. In case of Z axis analysis the axis X for a level image is coordinate axis Y and vice versa. For example if scanning axis is axis X it becomes the Z axis for a level image and the axis X in the level image is represented by the Z axis. The Y axis remains the same. The similar situation in axis changing is for Y axis as the scanning axis. In the pseudo code algorithm in Fig. 3 are axis X, Y and Z in meaning of described axis changing and that is why they have index I_L.

From known range R, according equation (1), are evaluated the pixel counts in individual axes according the qD parameter as following

$$\begin{aligned} H_{X_{I_L}} &= \frac{R_{X_{I_L}}(2) - R_{X_{I_L}}(1)}{qD} \\ W_{Y_{I_L}} &= \frac{R_{Y_{I_L}}(2) - R_{Y_{I_L}}(1)}{qD}, \end{aligned} \tag{2}$$

where H is the level image height and W is the width. Both parameters are used for evaluation the points exact position in a level image

$$\begin{aligned} I_{L_x} &= H_{X_{I_L}} - \frac{R_{X_{I_L}}(2) - PC(\forall, X_{I_L})}{qD} \\ I_{L_y} &= W_{Y_{I_L}} - \frac{R_{Y_{I_L}}(2) - PC(\forall, Y_{I_L})}{qD}. \end{aligned} \tag{3}$$

In arrays I_{L_x} and I_{L_y} are stored all coordinates in a level image. Several coordinates can be equal for one pixel in a level image. The individual pixels values are Z_{I_L} coordinates of an input point cloud. The result value for a pixel is mean value of all same coordinates

$$I_L(x, y) = \overline{PC(\sigma_{I_{L_x} = x} I_{L_x} \wedge \sigma_{I_y = y} I_{L_y}, Dim_s)}. \tag{4}$$

Symbol σ means the range of elements in a array. The last step in a level image construction is association with the origin position in a space

$$I_{L_{phys}}(x, y) = \left[R_{X_{I_L}}(1), R_{X_{I_L}}(2)\right]. \tag{5}$$

A level image origin is minimal value of level range R from (1). In Fig. 4 is a point cloud with a marked level and the Fig. 5 gives the created level image.

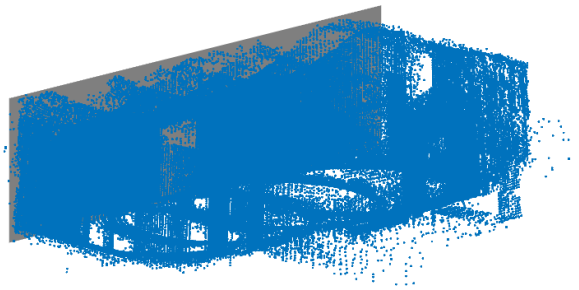

Fig. 4: Input point cloud with marked level

Fig. 5: Resulting level image

From a scanning process there can be also information about a point color. Such point cloud has usually size as $PC(N_p, 6)$, where N_p is the number of points and the last three channels are colors in RGB space. In this case it is

possible to create a second level image with a color by using equation (4), where an image pixel intensity is a color and not space data. The equation is defined as follows

$$I_L(x,y) = \overline{PC(\sigma_{I_{L_x}=x} I_{L_x} \wedge \sigma_{I_y=y} I_{L_y}, PC_{RGB})}. \quad (6)$$

The color is presented as the mean intensity of all same coordinates. In Fig. 6 is color point cloud from Fig. 4.

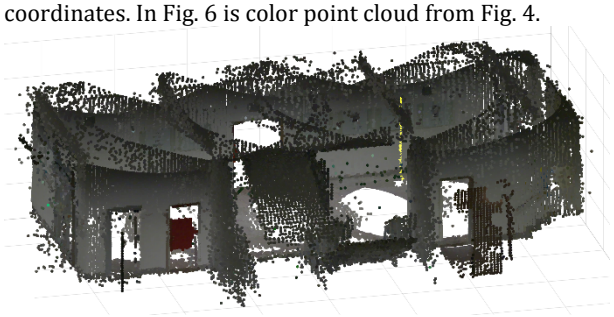

Fig. 6: Input point cloud with color information

Its level image in the same detection level dL is shown in Fig. 7.

Fig. 7: Resulting color level image

Searching image element boundary position

The image level itself visualize the points area coverage in a specific space level. For getting space important parameters the level image has to be further processed by the element analysis.

The output of the image connected component labeling is a binary image with element indexes and its total count. Individual element boundary position is necessary for the plane visualization by a level image.

To get a level image element boundary we subtract from the origin element the same element with applied morphologic erosion by the morphologic element disk with the size 3 pixels. The element boundary is defined as follows

$$I_L e_B = I_L e - I_L e_E, \quad (7)$$

where $I_L e$ is a level image element and $I_L e_E$ is a level image element with applied erosion.

Fig. 8 shows binary form of the level image from Fig. 5.

Fig. 8: Binary from of level image in Fig. 5

Resulting image according equation (7) is in Fig. 9.

Fig. 9: Result of equation (7)

Positions of a boundary are searched by using modified connected component labeling algorithm. The input image is $I_L e_B$. The basis of the modified algorithm is the changed neighbor pixel searching direction, illustrated in Fig. 10.

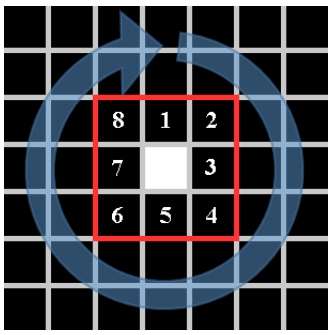

Fig. 10: Neighbor pixel searching direction

Numbers and the arrow shows the searching direction. When first neighbor pixel is located the algorithm record its position and search next neighbor pixel from the last founded pixel. Fig. 11 gives an algorithm pseudo code.

```
Function [e_B, (dir_eB)] = ElementBorderPositions(I_L e_B)
  for x=1:size(I_L e_B,1)
    for y=1:size(I_L e_B,2)
      if(I_L e_B(x,y) == true)
        eIdx = 1
        e_B(eIdx,:) = [x, y]
        checkArray = [x, y]
        continueSearch = true
        while (continueSearch)
          [found, pos, (dir)] = SearchNeighborPixel (Fig. 10)
          if(found)
            eIdx++
            e_B(eIdx,:) = [pos_x, pos_y]
            checkArray = [pos_x, pos_y]
            (dir_eB(eIdx) = dir)
          else
            continueSearch = false
          end
        end
        return
      end
    end
  end
end
```

Fig. 11: Pseudo algorithm of searching border positions

The searching process of an element boundary $e_B(N_{px_B}, 2)$, where N_{px_B} is the element pixels count and two means to store two dimensions, starts also in the left top image corner. When the first nonzero pixel is found, the boundary position is stored in e_B by using the boundary position index $eIdx$. The process continuous in searching other pixels according direction from Fig. 10. The algorithm ends when all boundary pixels are found. In the pseudo code it is also included the value dir_{e_B}, which storing direction from Fig. 10. This parameter is handy for future level image processing. The reason why we are not using simple boundary position from $I_L e_B$ is to have reliable boundary positions in acceding order and searching direction history. Fig. 12 (a) illustrates typical situation which can occur during the boundary searching. Fig. 12 (b) is false boundary detection and Fig. 12 (c) is correct detection because we are searching boundary in diagonal direction too. Black ones are boundary positions.

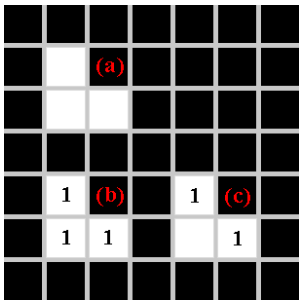

Fig. 12: Border detection cases

Plane visualization by using the level image

Described algorithm for searching elements' boundary position is mainly used for the visualization purpose in point clouds. The level image consists from quantized parts of points position by parameter qD. For each level image its origin $I_{L_{phys}}(x, y)$ is defined and each pixel has a real size and a position in space. The plane detection angle is also known.

For a plane visualization the Matlab function "patch" is used. The input parameters to this function are point coordinates in space and its connection order. In the input image there can be a hole and if a detailed plane visualization is required we cannot use directly the algorithm for searching boundary positions from Fig. 10. Thus whole process is split into partial steps. The Fig. 13 gives the visualization pseudo code.

```
Function PlaneVisualization(I_Le, qD, [∠v,∠h], nS)
  Create R(v,h)
  Determine element's start_E and end_e
  for n = start_e:nS:end_e
    [eP,numP] = findElementParts(I_Le(∀,n:n+nS)))
    for p = 1:numP
      e = eP(eP == p)
      [e_B] = ElementBorderPositions(e)
      Calculate points                          (9)
      points_R = points R(v,h)
      Draw a plane part by function patch(points_R)
    end
  end
end
```

Fig. 13: Pseudo algorithm of plane visualization

The visualization process starts by the rotation matrix construction from plane angle $R(h, v)$ and the level image analysis Fig. 14, which determines the horizontal elements' start $start_E$ and the end end_E.

Fig. 14: Horizontal range determination

Coefficient nS, set by a user, determines the individual column look-through step. For a detail plane visualization it is necessary to set parameter $nS = 1$, because the coefficient qD can have a bigger value and some details may be omitted. The visualization process is divided into

$$\left\lceil \frac{end_E - start_E}{nS} \right\rceil \tag{8}$$

steps. The number of steps nS is rounded to higher integer, because it is necessary to draw a whole plane. According its value are selected individual level image parts and a plane is progressively draw, see Fig. 15.

Fig. 15: Part of visualized plane marked by red

For demonstrative example it is selected a part with an interrupt. Then the part width is checked for 1px width. These parts are omitted. Thus the resulting visualization describes a plane of input level image I_L as best as possible. For finding all input elements' parts from original image in Fig. 9 the connected component algorithm is used. The each founded part p is processed by the described algorithm for searching boundary position e_B. From the level image origin $I_{L_{phys}}(x, y)$, the distance quantum parametr qD and pixel position $e_B(x, y)$ in the level image we can determine exact position in a space defined as

$$\begin{aligned} point_x &= I_{L_{phys}}(x) + e_B(x)qD \\ point_y &= I_{L_{phys}}(y) + e_B(y)qD. \end{aligned} \tag{9}$$

With every pixel position we build vectors in a space for all points. All points positions are rotated by rotation matrix $R(h, v)$

$$points_R = points \cdot R(h, v) \tag{10}$$

Founded vertices are processed by Matlab function "patch", which draw a part of an original plane. This described step is repeated until all plane parts are visualized. The Fig. 16 gives visualized input level image.

Fig. 16: Final plane visualization

In case of color level image, the processing is same. Color information is processed in the similar way as 3D level scanning described in (Chmelar et al., 2017f). The greyscale image intensity is Z axis and we analyzing individual levels. The output color is defined as

$$I_L e_C(R, G, B) = [\overline{I_L e_R}, \overline{I_L e_G}, \overline{I_L e_B}] \tag{11}$$

Mean values of individual color channels in specific intensity level are used. The following part shows visualization results of several point clouds.

Practical results

The visualization algorithm was tested on several point clouds. Fig. 17 and Fig. 18 presenting two input point clouds, the first normal and the second colored.

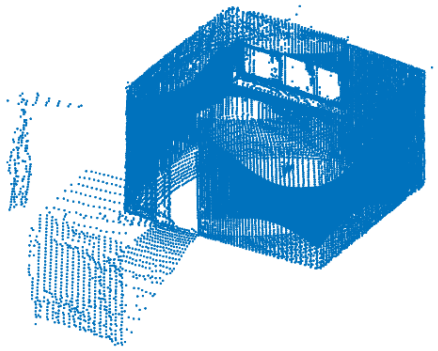

Fig. 17: Point cloud of a room

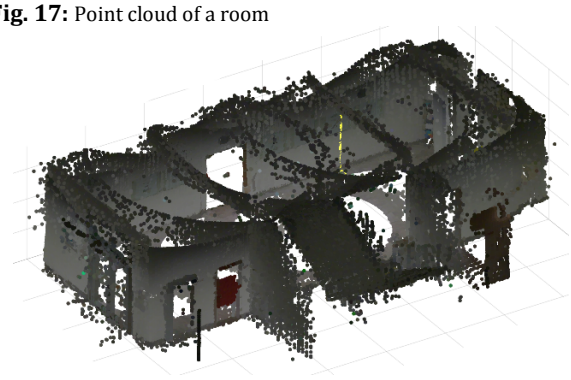

Fig. 18: Colored point cloud of a corridor

From the level image it is possible to get important parameters like elements' area or its perimeter. These values can be used as visualization condition. For example, the quantization parameter qD is set to 5 cm and we want to visualize only planes with area higher than 1 m^2. Following figures Fig. 19 and Fig. 20 are visualizations of point clouds above. Presented point clouds are scanned in all three dimensions X, Y and Z.

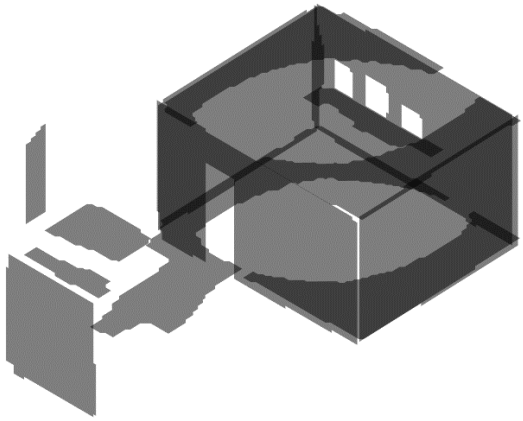

Fig. 19: First point cloud (Fig. 17) visualization

Fig. 20: Colored point cloud (Fig. 18) visualization

Conclusions

In this paper we described a method for point cloud plane visualization by using the level image. Presented results shows its ability to express presence of points in specific point cloud levels. A level image describes input point cloud planes by less points in other words. By knowledge its origin position in a space, the physical pixel size and the detected rotation angle we can easily determine a pixel position in a space.

The connection of image processing methods with the physical distance offers besides the visualization also get important properties about the scanned space. According our requirements, it is possible to present only planes with specific parameters.

Acknowledgment

The research was supported by the Internal Grant Agency of University of Pardubice, the project No. SGS_2017_030.

References

Beran, L., Chmelar, P., Rejfek, L. (2015c) Navigation of robotics platform using monocular visual odometry, 2015 25th International Conference Radioelektronika (RADIOELEKTRONIKA), Pardubice, (2015), pp. 213-216

Beran, L., Chmelar, P., Rejfek, L. (2015d), Navigation of Robotics Platform Using Advanced Image Processing Navigation Methods, VIPIMAGE 2015 - V. ECCOMAS Thematic Conference on Computational Vision and Medical Image Processing (2015)

Han, K., K., and Golparvar-Fard, M. (2017d) Potential of big visual data and building information modeling for construction performance analytics: An exploratory study, Automation in Construction, Volume 73, January 2017, Pages 184-198.

Chmelar, P., Beran, L., Rejfek, L. and Chmelarova, N. (2017e) The Point Cloud Visualisation for Rotary Optical Rangefinders, 2017 27th International Conference Radioelektronika, RADIOELEKTRONIKA 2017.

Chmelar, P., and Dobrovolny, M (2013). The fusion of ultrasonic and optical measurement devices for autonomous mapping, Radioelektronika 23rd International Conference, RADIOELEKTRONIKA 2013

Chmelar, P., Chmelarova, N., Rejfek, L., and Beran, L., (2017g) Advanced Plane Properties by Using Level Image, (unpublished)

Chmelar, P., Rejfek, L., Beran, L., and Dobrovolny, M. (2017f) A Point Cloud Decomposition by the 3D Level Scanning for Planes Detection, International Journal of Advanced and Applied Science, 2017, ISSN: 2313-626X.

Raymond, C., H., Lo. and William, C., Y., Lo. (2015b) OpenGL data visualization cookbook, Packt Publishing, August 2015

Rusu, R., B., and Cousins, S. (2011) 3D is here: Point cloud library (PCL), 2011 IEEE International Conference on Robotics and Automation, Shanghai, 2011, pp. 1-4.

Ryde, J. (2015a) Textured voxel-bounded planes: Efficient representation of color points for 3D mapping and visualization, 2015 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), San Diego, CA, 2015, pp. 166-171.

Yan, Z., Ye, M., and Ren, L. (2017c) Dense Visual SLAM with Probabilistic Surfel Map, In IEEE Transactions on Visualization and Computer Graphics, vol. 23, no. 11, pp. 2389-2398, Nov. 2017.

Zeng, B., Meng, F., Ding, H., Wang, G. (2017a) A surgical robot with augmented reality visualization for stereoelectroencephalography electrode implantation, International Journal of Computer Assisted Radiology and Surgery, 2017 Aug,12, Pages 1355-1368.

Zhang, H., Wei, Q., Jiang, Z. (2017b) 3D Reconstruction of Space Objects from Multi-Views by a Visible Sensor, Sensors (Basel, Switzerland). 2017, 17(7):1689

↑ If possible equalize columns on the last page ↑