Theoretical and Applied Informatics
ISSN 1896–5334
Vol. 25 (2013), no. 2
pp. 105–117
DOI: 10.2478/thai-2013-0009

Using RANSAC for 3D point cloud segmentation

Leszek Luchowski[a], Przemysław Kowalski[b]

[a,b] Institute of Theoretical and Applied Informatics
Polish Accademy of Science
ul. Bałtycka 5, Gliwice, Poland
[a] *leszek.luchowski@iitis.gliwice.pl*
[b] *przemek@iitis.gliwice.pl*

Received 17 March 2013, Revised 16 August 2013, Accepted 26 September 2013

Abstract: The article presents a method for 3D point cloud segmentation. The point cloud comes from a FARO LS scanner – the device creates a dense point cloud, where 3D points are organized in the 2D table. The input data set consists of millions of 3D points – it makes widely known RANSAC algorithms unusable. We add some modifications to use RANSAC for such big data sets.

Keywords: 3D segmentation, point cloud, RANSAC

1. Introduction

There are many algorithms proposed for 3D segmentation. The proposed algorithms differ due to the input data. Most of the solutions are dedicated to 3D mesh segmentation (eg. [14, 16, 19]). Our 3D data is represented as a (grid) point cloud, and the solutions for 3D mesh segmentation cannot be used for our purposes.

There are two main types of segmentation algorithms for 3D data [11, 20]: edge-based [7, 10, 13, 15] and region-based [8, 9, 17]. The third group consists of hybrid methods. Our RANSAC algorithm is an example of region-based methods.

We use a 3D point cloud given by FARO LS scanner (see fig. 1-3). The input data is a huge grid point – the table consists of almost 9000x4000 3D points (about thirty six millions points).

Our input data represents human made interiors (we may assume the segments are simply shapes, especially planes [18]). The found shapes may be used to:
– building of semantic map,
– finding of badly represented areas used for adaptive scanning,
– finding of correspondence between scans (using similar shapes).

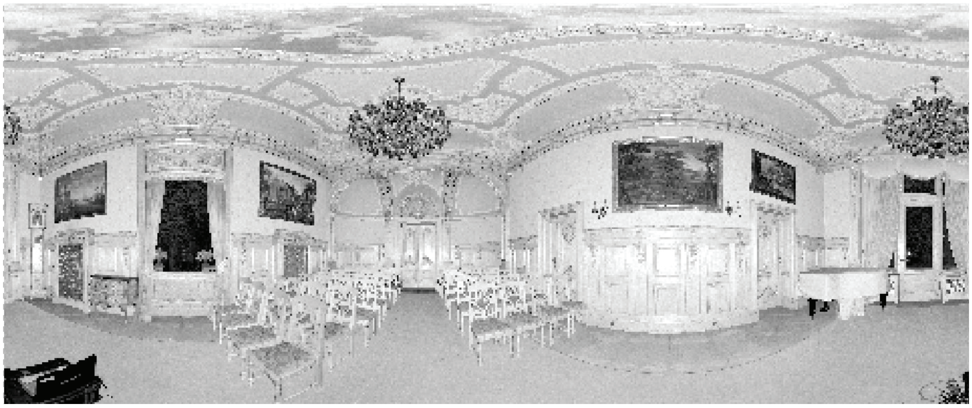

Fig. 1. Scan made by FARO LS laser 3D scanner – the dining room in Caro Villa.
Light intensity represents reflection of infrared light.
(The view was generated by JRC 3D Reconstructor®.)

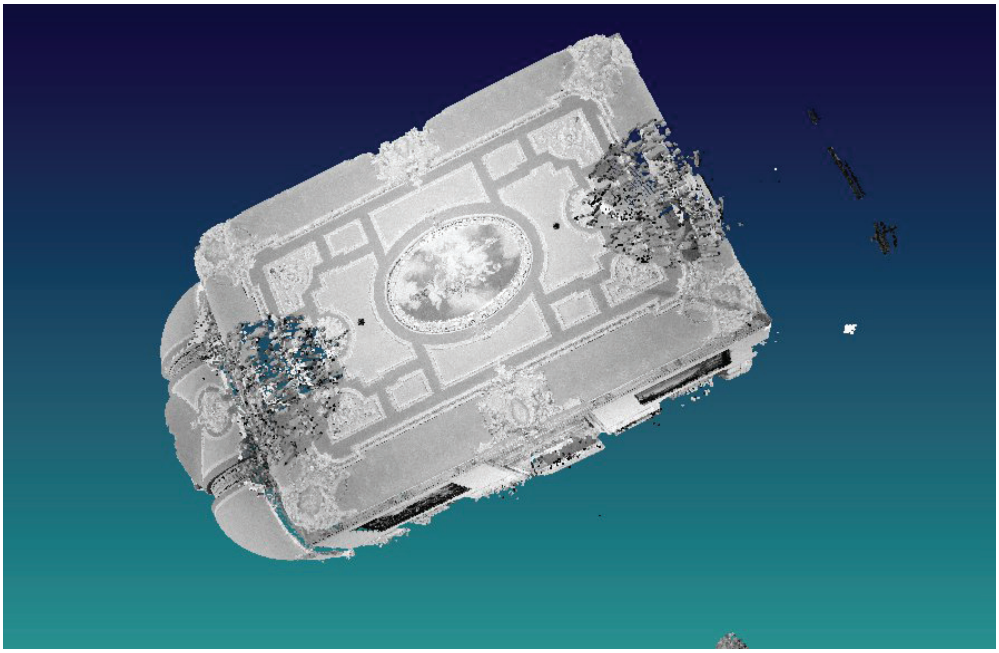

Fig. 2. 3D visualisation of the same scan – outside of the dining room in Caro Villa.
(The view was generated by JRC 3D Reconstructor®.)

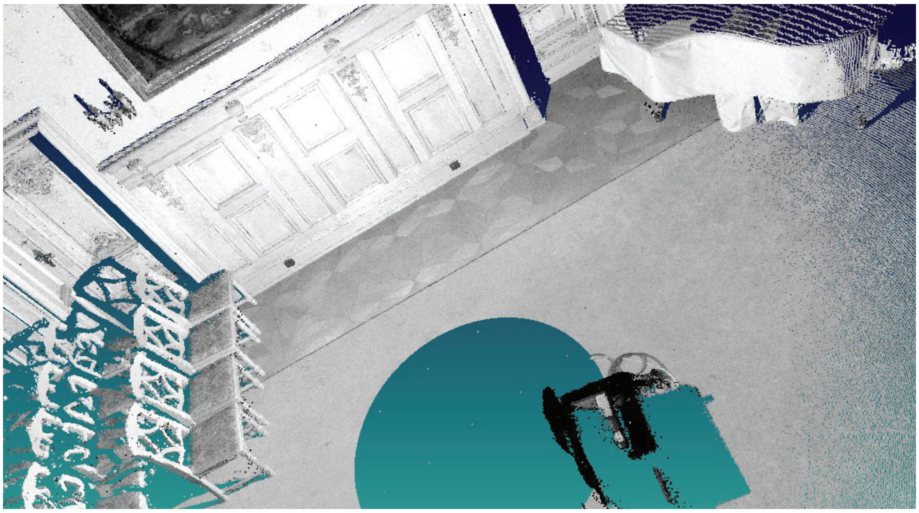

Fig. 3. 3D visualisation of the same scan – inside of the dining room in Caro Villa. (The view was generated by JRC 3D Reconstructor®.)

There are two known methods that base on RANSAC algorithm for segmentation of 3D shapes: the first [8, 17] is used to find parts of buildings (roofs and facades – only planes) in LIDaR 3D data; the second [9] finds planes, spheres, cylinders, cones and tori (objects with three to seven parameters; for natural human perception environment as a set of primitives [11]). In our method we are looking for quadrics of all types. The method is also adapted to characteristics of the input data – the volume of data (about thirty five millions of points) is too big for RANSAC, so RANSAC algorithm is used only for parts ("windows") of the incoming data.

2. Related Work

In the previous work on 3D data segmentation using RANSAC [21] an input data comes from Minolta Vi-9i laser scanner. The input data consists of a smaller number of points – it makes possible using RANSAC algorithm with the whole input data set. The new version of the algorithm uses 3D point clouds generated by the FARO LS laser scanner. The data sets are much bigger (millions of points in comparison to thousands of points) and points are stored in the 2D table. The size of analysed point clouds makes previous version of the algorithm unusable.

The previous work uses only quadrics, while the new work prefers planes, still using quadrics.

2.1. Comparing RANSAC to other algorithms serving a similar purpose

The purpose of the present work is to identify, in a given set of 3D points, its subsets, which can be reasonably well approximated by quadric surfaces. This task belongs to a class of problems well known to data processing: the identification of structured subsets in sets of observation vectors.

In general terms, this class of problems can be described in the following terms:

- given is a set S={p[i]}, i=1...k of data vectors (points in R^n);
- some of the vectors may belong to one or more geometric structures defined by an equation F[j](p)=0, where F[j] is a function with adjustable parameters; those points comply with the equation with a certain degree of error, either because of measurement imprecision or because the physical reality they represent is not perfect;
- given is a procedure G (a fitting algorithm) which can identify the parameters of F from a small set of points, provided that they all belong to F;
- also given is an error measure quantifying the degree of non-compliance of a point with a structure;
- the task is to identify all structures which are represented by a sufficient number of points, determine their parameters and the corresponding point subsets (the *support sets* or *inlier sets* of the structures; points not belonging to a given structure – which may or may not belong to another one – are referred to as *outliers*).

Among the algorithms that exist for this type of problems, we considered the Hough transform [2, 8, 3, 4] and RANSAC [5].

The Hough transform and its generalizations detect structures in point clouds by allowing each point to „vote" for all potential structures that pass through it. At the heart of the method is an accumulator table, which is an array with as many dimensions as there are parameters describing the structures.

$$\sum_{i,j=1}^{3} x_i Q_{ij} x_j + \sum_{i=1}^{3} P_i x_i + R = 0$$

Where: Q, P, and R are parameters. Q is 3x3 matrix, P is 3-dimensional row vector, R is a scalar constant.

In our case, where the structures are quadrics described by 10 parameters (reducible to 9 by eliminating the scale factor), the size of the array poses several problems:

- the array will exceed the RAM capacity of most computers. Quantifying each parameter into just 10 bins and using four-byte long integer format leads to a 40GB array;
- the time required to fill the array by having each data point „vote” for all quadrics it can potentially belong to is also considerable, due both to the large number of such quadrics and the complexity of determining them. According to Shapiro and Stockman [6], the computational complexity of the Hough transform is $O(M^{m-2})$, where m is the number of parameters and M is the number of discrete values each of them can assume.

The RANSAC algorithm does not use such exponentially-growing data structures, and its computational cost depends on the number of samples that need to be considered to ensure a sufficient probability of finding one that only consists of inliers.

This probability Γ is given by the formula [1]:

$$\Gamma = 1 - (1 - (1 - \varepsilon)^p)^s$$

where:
ε – the fraction of outliers in the population,
s – the number of samples taken,
p – the number of features in each sample.

For a given required Γ, the required number of samples is

$$s = \ln(1 - \Gamma) / \ln(1 - (1 - \varepsilon)^p)$$

In our case, where a Γ of 0.95 was acceptable and ε was typically near 0.7, a few hundred thousand iterations yielded satisfactory results. As each iteration involved checking a few tens of thousands of points, the overall computing time was orders of magnitude less than in the case of the Hough approach.

3. Using RANSAC on large data sets.

The volume of data (tens of thousands of points) obtained from a typical 3D scanner makes it impractical to process with the original RANSAC algorithm as a single set. Storing the whole scan in a core-memory array proved impossible (on a contemporary PC-class computer), and running the algorithm on a set stored in a disk file would be prohibitively time-consuming, because RANSAC, by definition, accesses data in random order.

A modified version of the algorithm was developed, relying on the assumption that points belonging to a feature will tend to occur close to each other, and therefore little is risked by splitting the full data set into square windows and applying RANSAC to one such window at a time.

By „square windows" we mean subsets cut out from the 3D set by squares in the XY plane.

The modified algorithm proceeds in two stages. In the first stage, each window is submitted to the RANSAC algorithm and quadrics are identified among its points. In the second stage, the quadrics are compared to each other and merged (along with their supporting point sets) if they are similar enough.

3.1. Stage 1 – Search for quadrics in windows.

```
Define numbered set QS={QS[i]} of quadrics, initially
empty.
Define numbered set T={T[i]} of point files, also initially
empty.
N = 0;
for each square window W
      mark all points in W as unused
      for each unused point P in W
            if P lies on any quadric QS[i] in QS
                  mark P as used and add it to T[i]
            endif
      endfor
      itercount = 0;
      do
            apply RANSAC to the unused points in W
            if a quadric is found
                  N++;
                  add the quadric to QS as QS[N]
                  store its points in a file and add the
                  file to T as T[N]
                  mark the points as used
            endif
            itercount++;
      until ( (itercount>itermax) OR (fewer than 10 unused
      points remain) )
endfor
```

In fact, sometimes RANSAC algorithm gives quadrics equation with Q and P parameters close to 0. In such situation we cannot use the quadric, so the RANSAC algorithm is used in the loop (break if the Q or P parameters significantly different from zero, or if the number of iterations reach the maximum).

3.2. Stage 2 – integration of quadrics

```
N = number of quadrics in Qs
mark all quadrics in QS as unused
for i = 1 to N
    if QS[i] unused
        for j = i+1 to N
            if distance(QS[j], QS[i]) < upsilon
                mark QS[j] as used
                fuse T[j] into T[i] (i.e. Add points
                of T[j] toT[i] and make T[j] empty)
            endif
        endfor
    endif
endfor
```

Where `distance(QS[j], QS[i])` is defined as a weighted difference between parameters of the quadrics. The biggest weight is associated with R parameter, as associated with position of the quadric (parallel quadrics differ by R).

4. Results and conclusions

The algorithm was tested using scans of the dining room in the Caro Villa (Museum in Gliwice). A typical scan used for our experiments consists of 38016000 points (grid size: 9600x3960). To reduce number of the points we use only mean position for a given neighbourhood (in tests the neighbourhood was set to 4x4, that reduces the number of points to 2 376 000).

Results of the algorithm depend on the data and parameters. The main parameter is an acceptable distance between analysed point and the found quadric. The parameter determines size of the found surfaces. In presented example the parameter was set to 0.0025 (the parameter depends on the input data). Size of the used window was set to 90 x 90.

In the first stage we found 299 surfaces (consisting between 210 and 20512 points). Some of the found surfaces are presented on fig. 4.

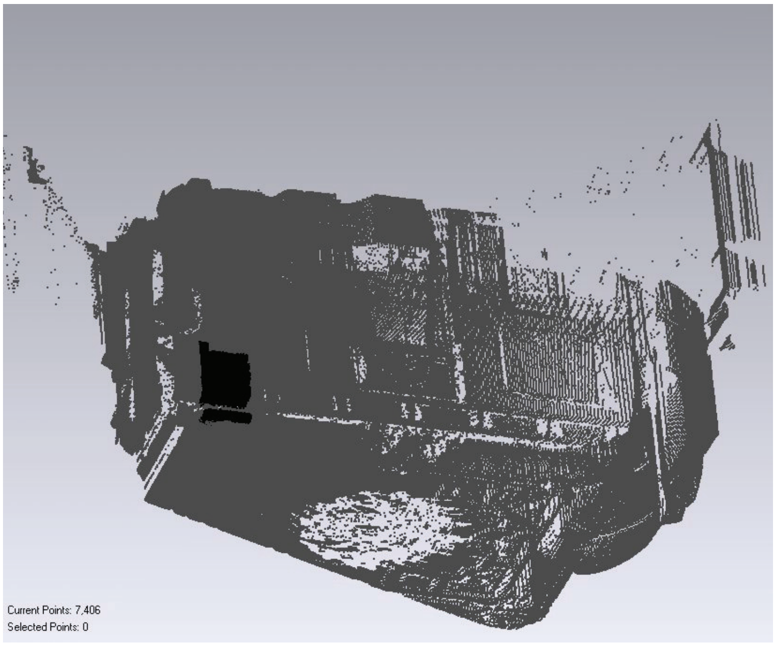

Fig. 4. The found quadric (black) in part of the scan (grey).
(The view was generated by Geomagic Studio 10®.)

There are some problems with the output: number of found quadrics (found in small windows), artificial surfaces.

4.1. Quadrics integration

We have too many surfaces in the first stage of the algorithm and found surfaces are small (see fig. 4). The surfaces may be integrated, but the comparison criteria are not-trivial choices. The objects in human made environment are not only built using primitives, but also contains many parallel surfaces – in result, the position of a surface (given by R parameter) is the most important for comparison.

We presented one wall of the analysed dining room in Caro Villa on fig. 5. The quadrics of the wall were integrated using weighted square error (parameters Q and P are multiplied by 0.9, while R parameter is multiplied by 1.9). The threshold was set to 0.1.

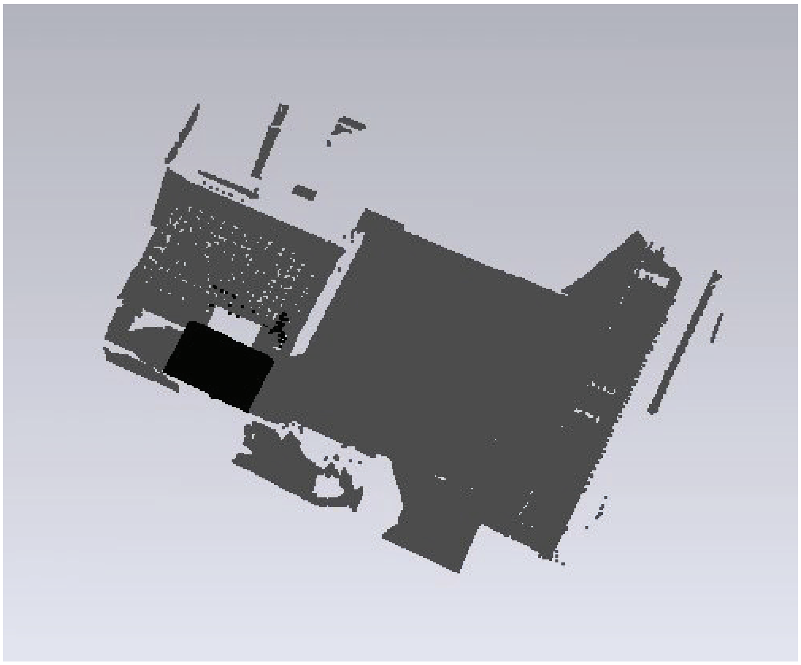

Fig. 5. A set of quadrics integrated into one wall. One of them is black.
(The view was generated by Geomagic Studio 10®.)

The "wall" presented on the fig. 5 consists of twelve quadrics, with the biggest error 0.047. We found 85 'integrated surfaces' (but most of them are single quadrics found in the first stage with not found equivalents).

4.2. Artificial surfaces

Some of the surfaces are artificial planes made by points on the scanning plane. The artificial planes can be easily removed – the planes consists of central point of the scan (0, 0, 0) that means R=0.

5. Usage and future works

Segmentation of 3D scans is used for many purposes. We are using 3D scans segmentation for building semantic maps, adaptive scanning and finding correspondence between scans.

5.1. Semantic map

The semantic map represents the environment describing parts using labels – e.g. a group of integrated quadrics (see fig. 5) should be described as ‘wall’. Other planes in the scan should be labelled as: wall, floor, roof. To set a proper label for a part of environment we need: type of the quadric (e.g. ‘plane’), position (altitude) and orientation (‘vertical’, ‘horizontal’, etc.).

5.2. Adaptive scanning

Considering quality of scans of cultural heritage objects, we observe that parts of the same object differ in quality [12]. Using a 3D scanner we are facing additional dilemma: using higher density of scanning gives us more points and causes problem with analysis of the scan. We may analyse distances between points and quadrics – if the distance is small, the surface is well suited and the part of the environment is described with proper density of points; if the distance is higher we need additional scanning process with higher density for a part of the environment.

5.3. Correspondence between scans

Finding correspondence between scans is an important problem in scan registration and integration tasks. The number of found quadrics is smaller than number of points, and we may reduce complexity of the algorithm using only quadrics of the same type in process of finding correspondence.

5.4. Future works

We are planning modify criteria for quadrics integration and add new criteria for the best quadric parameters for a set of points.

Acknowledgments

Thanks to NCN (project number: N N516 440738), and to Museum in Gliwice – the data presented on illustrations (see fig. 1-5) represent a dining room in Caro Villa.

References

1. M. Pollefeys, *Visual 3D modeling from images – tutorial notes*, University of North Carolina, Chapel Hill, USA
2. P. V. C. Hough, *Machine analysis of bubble chamber pictures*, Tech. Report, CERN, 1959
3. R. O. Duda, P. E. Hart, *Use of the Hough Transformation to Detect Lines and Curves in Pictures*, Comm. ACM, vol. 15, pp. 1115, 1972
4. D. H. Ballard, *Generalization the Hough Transform to Detect Arbitrary Shapes*, Pattern Recognition, vol. 13, no. 2, pp. 111-122, 1981
5. M. A. Fischler, R. C. Bolles, *Random samples consensus: a paradigm for model fitting with applications to image analysis and automated cartography*, Comm. ACM, vol. 24, no. 6, pp. 381-395, 1981
6. L. Shapiro, G. Stockman, *Computer Vision*, Prentice Hall, 2001
7. G. H. Liu, Y. S. Wong, Y. F. Zhang, H. T. Loh, *Error-based segmentation of cloud data for direct rapid prototyping*, Computer-Aided Design, vol. 35, pp. 633-645, 2002
8. F. Tarsha-Kurdi, T. Landes, P. Grussenmeyer, *Hough-transform and extended ransac algorithm for automatic detection of 3d building roof planes from lidar data*, Proceedings of the ISPRS Workshop on Laser Scanning, pp. 407-412, 2007
9. R. Schnabel, R. Wahl, R. Klein, *Efficient RANSAC for point-cloud shape detection*, Computer Graphics Forum, vol. 26, no. 2, pp. 214-226, 2007
10. A. Courtial, E. Vezzetti, *New 3D segmentation approach for reverse engineering selective sampling acquisition*, International Journal Advances Manufactory Technology, vol. 35, pp. 900-907, 2008
11. Y. Liu, Y. Xiong, *Automatic segmentation of unorganized noisy point clouds based on the Gaussian map*, Computer-Aided Design, vol. 40, pp. 576-594, 2008
12. K. Skabek, P. Kowalski, *Building the Models of Cultural Heritage Objects Using Multiple 3D Scanners*, Theoretical and Applied Informatics, vol. 21, pp. 115-129, 2009
13. J. A. P. Kjellander, M. Rahayem, *Planar segmentation of data from a laser profile scanner mounted on an industrial robot*, International Journal Advances Manufactory Technology, vol. 45, pp. 181-190, 2009
14. A. Golovinskiy, T. Funkhouser, *Consistent segmentation of 3D models*, Computer & Graphics, vol. 33, pp. 262-269, 2009
15. E. Li, B. Lévy, X. Zhang, W. Che, W. Dong, J-.C. Paul, *Meshless quadrangulation by global parametrization*, Computer & Graphics, vol. 35, pp. 992-1000, 2011

16. F. Bergamasco, A. Albarelli, A. Torsello, *A graph-based technique for semi-supervised segmentation of 3D surfaces*, Pattern Recognition Letters, vol. 33, pp. 2057-2064, 2012
17. J. Martinez, A. Soria-Medina, P. Arias, A. F. Buffara-Antnes, *Automatic processing of Terrestrial Laser Scanning data of building façades*, Automation in Construction, vol. 22, pp. 298-305, 2012
18. L. Wang, J. Cao, Ch. Han, *Multidimensional particle swarm optimization-based unsupervised planar segmentation algorithm of unorganized point clouds*, Pattern Recognition, vol. 45, pp. 4034-4043, 2012
19. M. Meng, J. Xia, J. Luo, Y. He, *Unsupervided co-segmentation for 3D shapes using iterative multi-label optimization*, Computer-Aided Design, vol. 45, pp. 312-320, 2013
20. O. Wirjady, *Survey of 3D image segmentation method*, Berichte des Frauenhofer ITWM, no. 123, 2007
21. L. Luchowski, *Adapting the RANSAC algorithm to detect* 2^{nd}*-degree manifolds in 2D and 3D*, Theoretical and Applied Informatics, vol. 24, no. 2., pp.151-158, 2012

Wykorzystanie algorytmu RANSAC dla segmentacji chmur punktów 3D

Streszczenie

Artykuł prezentuje metodę segmentacji chmury punktów 3D. Segmentacja znajduje w chmurze (kracie) punktów kwadryki. Źródłem danych są chmury punktów uzyskane przy pomocy skanera FARO LS. Skany wykonane przy wykorzystaniu tego skanera charakteryzują się zapisem punktów w tablicy (stąd określenie ‘krata’ punktów), przy czym jej rozmiary są znaczne – w eksperymentach wykorzystano kratę liczącą 9600x3960, co daje 38 016 000 punktów, podkreślając znaczenie czynnika złożoności pamięciowej algorytmów. Przedstawione rozwiązanie uwzględnia ten problem wywołując czasochłonny algorytm RANSAC jedynie dla wycinków analizowanej sceny, a następnie wykorzystuje uzyskane rezultaty do dalszej analizy. W artykule zaprezentowano szczegółowo algorytm RANSAC i zasady analizy wycinków skanu.

Dane wejściowe dla algorytmu reprezentują scenę utworzoną przez człowieka (wnętrze pomieszczenia), co oznacza pojawianie się wielu płaszczyzn i innych prostych obiektów geometrycznych (np. wycinków walca). Prezentowane rozwiązanie pozwala na odnalezienie w scenie kwadryk, rozwiązanie takie pozwala objąć wiele kształtów tworzonych przez człowieka.

W przeprowadzonych eksperymentach analizowano skan jadalni Willi Caro – dziewiętnastowiecznej willi, będącej jedną z siedzib Muzeum w Gliwicach. Wybór takiego przedmiotu eksperymentów jest powiązany z jednym z docelowych zasto-

sowań – skanowaniem obiektów dziedzictwa kulturowego celem dokonania ich inwentaryzacji architektonicznej. Wyznaczenie kwadryk opisujących fragmenty skanu pozwala dobrać dokładność skanowania (zwiększenie dokładności dla wybranych fragmentów – detali artystycznych) w zależności od złożoności powierzchni. Ilustracje 1-3 prezentują analizowany skan, ilustracja nr 4 przedstawia punkty przypisane do kwadryk (wszystkich znalezionych przez oprogramowanie), a nr 5 zintegrowane kwadryki dla jednej ze ścian jadalni.

W wyniku analizy znaleziono 299 kwadryk (o rozmiarach od 210 do 20512), które po integracji utworzyły 85 zintegrowanych powierzchni (wiele z nich to jednak pojedyncze kwadryki z pierwszego etapu przedstawiania, dla których nie znaleziono odpowiedników).