# Sphere Detection in Kinect Point Clouds Via The 3D Hough Transform

Anas Abuzaina, Mark S. Nixon, and John N. Carter

School of Electronics and Computer Science, University of Southampton, UK
aa6g08,msn,jnc@ecs.soton.ac.uk

**Abstract.** We introduce a fast, robust and accurate Hough Transform (HT) based algorithm for detecting spherical structures in 3D point clouds. To our knowledge, our algorithm is the first HT based implementation that detects spherical structures in typical in 3D point clouds generated by consumer depth sensors such as the Microsoft Kinect. Our approach has been designed to be computationally efficient; reducing an established limitation of HT based approaches. We provide experimental analysis of the achieved results, showing a robust performance against occlusion, and we show superior performance to the only other HT based algorithm for detecting spheres in point clouds available in literature.

**Keywords:** Evidence gathering; point clouds; 3D object detection; Hough transform; 3D vision; Kinect

## 1 Introduction

Much progress has been made in the development of 3D acquisition methods and technologies, especially recently when new 3D sensing technologies became available to the public such as the Microsoft Kinect. By providing a third dimension of the spatial coordinates of their output data (point clouds), the Kinect and similar depth sensors provide practical and accurate means for overcoming traditional problems in computer vision applications. Such problems are mainly related to background and foreground segmentation, object occlusion, unknown object scaling and changing lighting conditions. In this research we are interested in detecting parametric shapes, specifically spheres, in 3D point clouds generated by the Kinect sensor. We extend the Hough Transform (HT) [1] to three dimensions and show its robust ability to detect spheres in point clouds of a real environment. To our knowledge, our research is the first HT based implementation to detect spherical structures in real point clouds acquired by depth sensors such as the Kinect. To ensure our algorithm's practicability, we have reduced computational complexity and ensured acceptable time performance by using a sparse matrix representation. We provide experimental analysis based on a number of factors, emphasising on performance against occlusion ratio, and we show superior performance to the only other HT based algorithm for detecting spheres in point clouds available in literature.

## 2 Parametric shape detection in 3D

In 3D, parametric shapes such as planes, spheres, cylinders and cones can be considered as geometric primitives that are used to explain parts of the 3D point cloud. According to [2], such geometric primitives enable the system to add surface information for parts of the object that were not covered by the sensor data. Moreover, exploiting the knowledge about the shape enables to eliminate inaccuracies caused by sensor noise as it determines the best shape fit for the sensor data.

There have been two main directions of methods for detecting parametric shapes in 3D data: approaches based on RANSAC [3] and on Hough transform (HT) [1]. RANSAC (RANdom SAmple Consensus) based approaches decompose the point cloud into a concise structure of inherent shapes and a set of remaining points using random sampling, each detected shape serves as a proxy for a set of corresponding points. These approaches have been proven to be efficient for detecting parametric 3D shapes (planes, spheres, cylinders, etc.), such as in [4]. The other direction of methods for detecting parametric shapes is to use evidence gathering techniques such as the Hough transform (HT), in which a voting procedure is carried in the parameter space from which object candidates are obtained as local maxima. Hough transform based methods have been successfully implemented to detect planes [5][6], spheres [7] and cylinders [8] in 3D data. HT based methods in general have faster runtime than RANSAC. Moreover HT approaches can detect multiple instances of the shape while RANSAC methods are limited to a single instance. In this research we will develop an efficient HT based approach to detect spheres. A good comparisons between RANSAC and HT approaches for detecting parametric shapes in 3D data are in [9] and in [10].

## 3 The 3D Hough Transform for sphere detection

The Hough Transform (HT) [1] is a well-known technique originally developed to extract straight lines, since then it has been extended to extract parametric shapes (such as conic sections) and non-parametric shapes in 2D images. The Hough Transform is recognized as a powerful evidence gathering tool in shape analysis that gives good results even in the presence of noise and occlusions. The implementation of Hough transform based methods defines a mapping from the image (or point cloud in case of 3D) to an accumulator space; the evidence is the votes which original image points cast in the accumulator space. In three dimensions, a sphere has a simple polar parametric representation given by the following equations:

$$\begin{aligned} x &= c_x + r\cos\theta\sin\varphi \\ y &= c_y + r\sin\theta\sin\varphi \\ z &= c_z + r\cos\varphi \\ (0 \le \theta \le 2\pi, 0 \le \varphi \le \pi) & \end{aligned} \tag{1}$$

where $x$, $y$ and $z$ are the Cartesian coordinates of the surface points, $c_x$, $c_y$ and $c_z$ are the coordinates of the centre of the sphere, $r$ is the radius expressed in the same units as the centre and surface points and $\theta$ and $\varphi$ are the azimuthal and polar angles respectively. The implementation of the 3D HT to detect spheres is as follows: each point on the sphere surface defines a set of spheres in the accumulator space; these spheres are defined by all possible values of the radius, and they are centered at the coordinates of the point. For a given radius value, each point on the sphere defines only one sphere in the accumulator space. After gathering all evidence of the points on the sphere using the HT mapping equations (2) applied for each point in the sphere's surface, the maximum of the accumulator space corresponds to the parameters of the original sphere. Algorithm 1 describes the implementation of the 3D HT for sphere detection.

Since we are dealing with 3D point clouds, there will be four parameters to vote for ($cx$, $cy$, $cz$ and $r$), hence the accumulator space is 4-dimensional .

$$\begin{aligned} c_x &= x - r\cos\theta\sin\varphi \\ c_y &= y - r\sin\theta\ sin\varphi \\ c_z &= z - r\cos\varphi \end{aligned} \tag{2}$$

---

**for** *for every point (x, y, z)* **do**
  **for** $(r = r_{min}; r \leq r_{max}; r++)$ **do**
    **for** $\theta = 0; \theta \leq 2\pi; \theta++$ **do**
      **for** $\varphi = 0; \varphi \leq \pi; \varphi++$ **do**
        $c_x = x - r\cos\theta\sin\varphi$
        $c_y = y - r\sin\theta\sin\varphi$
        $c_z = z - r\cos\varphi$
        $\text{Accumulator}[r][c_x][c_y][c_z]++$
      **end**
    **end**
  **end**
**end**
Search Accumulator for peak.

---

**Algorithm 1:** 3D Hough transform for sphere detection (3DHT)

## 4 Implementation

To reduce memory requirements, and for fast access of elements, we use a multi-dimensional sparse matrix representation for the accumulator space. Sparse matrices store only non-zero elements, and provide direct mapping between indices and their corresponding values. The non-zero elements are stored in a hash table that grows when it is filled so that the search time is $O(1)$ in average regardless of whether element is present or not, and regardless of the size of the matrix.

We are interested in detecting spheres in real data, which is point clouds acquired from the Kinect and similar depth sensors. Point clouds generated by these sensors are 2.5D, i.e. only points on surfaces facing the sensor are acquired, which are about 50% of the total number of points on the object's surface, hence a complete geometry of the object cannot be acquired. Our algorithm is able to robustly detect "partial" spheres such those acquired by depth sensors, without this characteristic it would not be applicable on real data.

Point clouds generated by the Kinect have on average 300k points, each point cloud have been cropped to a suitable volume before applying Algorithm 1 to discard redundant points, i.e. points that are far away from the interest area of the scene. Figure 1 shows the robust performance of Algorithm 1 applied on Kinect point clouds of different scenes including a football and an orange. The left row shows original textured point clouds, the middle row shows detected spheres superimposed on original clouds and the third row shows the untextured point clouds with detected spheres. Figure 2 demonstrates the algorithm's capability to detect semi-spherical objects.

## 5 Analysis

Our experimental analysis of Kinect clouds shows that the *volume point density*, i.e. the number of points on a 3D surface, is $\sim 25$ points per cube centimetre (taken at distance of 1m). We are interested in coordinate values to the nearest centimetre, hence it is not necessary to include all points in voting process.

Taking every *ith* point instead of every point in the cloud, reduces the processing time significantly. However if the point step ($i$) is too large, false detection will occur. Based on our experimental analysis, a point step value of ($i = 20$) is a suitable value for typical Kinect clouds. Table 1 shows the total processing time of Algorithm 1 for detecting spheres with pre-defined radius, for undefined radii, the processing time is simply multiplied by the number of the radii in the given range $[r_{min}, r_{max}]$.

The computation of Algorithm 1 is divided to three processes:

1. Mapping: This is the transforming of points from the cloud's 3D space to the 3D Hough (parameter) space, applied by equations 2. This process is the most time consuming, it takes roughly 59.4% of the total processing time when Algorithm 1 is applied on a typical Kinect point cloud.
2. Voting: In this process, casting of votes occurs by accessing the elements of the sparse matrix and incrementing their values according to the Hough mapping. This process takes roughly 40.2% of total processing time.
3. Detection: In this process, a search algorithm is performed to find the peak of the accumulator. This process is extremely fast and it is less than 0.4% of total processing time. This is due the implementation of a sparse matrix for the accumulator in which the search time is always $O(1)$ in average.

The work of [7] is the only HT based method available in the literature dedicated for detecting spheres in 3D point clouds. However in their paper the

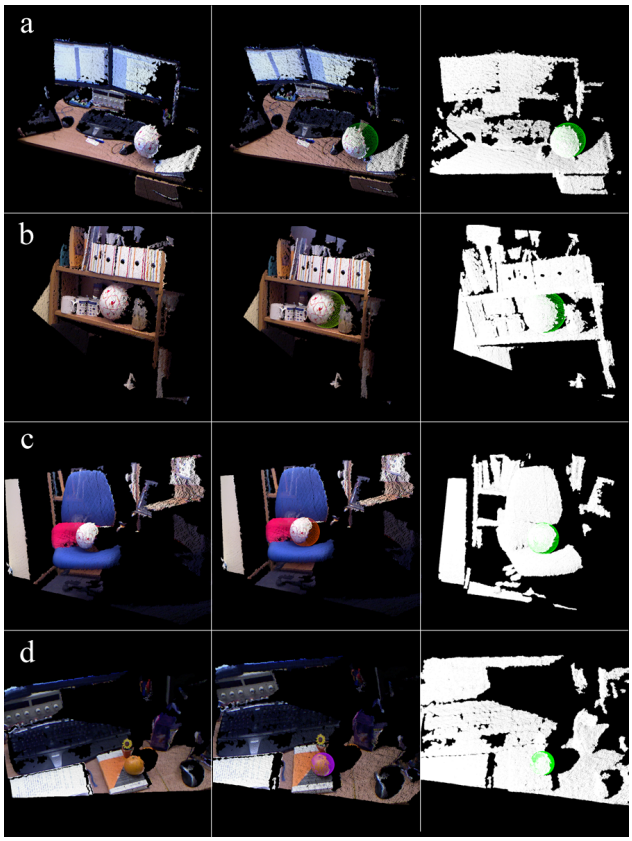


**Fig. 1.** Accurate sphere detection in real point clouds. Point cloud resolution: (a) 146k points, (b) 193k points, (c) 174k points and (d) 120k points.

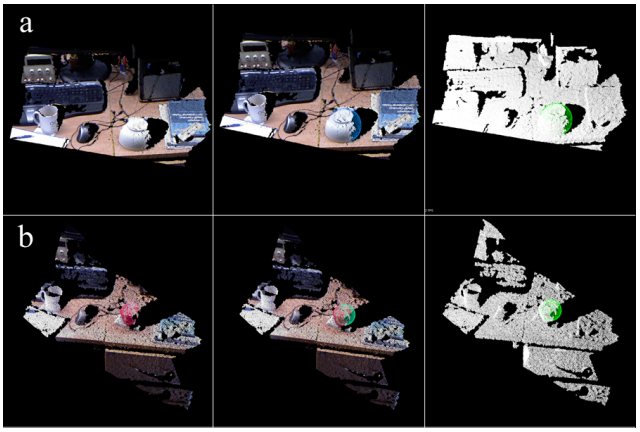


**Fig. 2.** Detection of semi-spherical objects. Point cloud resolution: (a) 136k points, (b) 64k points.

demonstrate their method only on a high resolution point cloud of calibration object of a number of "perfect" spheres acquired by an optical shape measurement system (SMS). Additionally they only provide performance analysis based on the parameter space resolution (bin size).

Even though our developed algorithm is similar to [7] in terms of spherical equations used for voting and sparse matrix use, in our research we are the first to apply HT based method to detect spheres in point clouds in real life scenarios of typical household and office environments generated by consumer depth sensors such as the Kinect. Moreover we demonstrate the capability of detecting semi-spherical objects such as apples an bowls. Additionally, we provide performance analysis against a number of different factors. Finally and most importantly, our algorithm detects spheres correctly in significantly less time.

In [7], their algorithm takes approximately 8 seconds in total to process a point cloud of 63K points. It can be seen from Table 1 that our algorithm is approximately 4 times faster, that is if *every* point has been taken in the voting process. However our algorithm can correctly detect spheres up to 57 times faster if the point step ($i$) is increased to a suitable value. We are interested in providing

| Point cloud name | No. of points | Time(s), i =1 | Time(s), i =10 | Time(s), i =20 |
|---|---|---|---|---|
| Deskball | 146k | 5.30 | 0.76 | 0.44 |
| ShelfBall | 193k | 7.11 | 0.91 | 0.55 |
| Chairball | 174k | 6.31 | 0.89 | 0.52 |
| Bowl | 136k | 4.08 | 0.46 | 0.24 |
| Orange | 120k | 3.42 | 0.38 | 0.20 |
| Apple | 64k | 2.06 | 0.25 | 0.14 |

**Table 1.** Total processing time of Algorithm 1 at point steps of 1,10 and 20, with angle step of 10 ° of both $\theta$ and $\varphi$ and a fixed raduis (running on 2.4GHz Intel Core i7 and 4GB RAM).

a performance measure in terms of accuracy of detection and computational time; there are a number of factors affect the algorithm's performance:

1. The number of points on the surface of the sphere which cast votes. These points should be enough to vote for the correct peak in the parameter space, if only few points are present on the sphere there will be no notable peak and hence the detection will fail. This number can be relative to the total number of points that a "complete" sphere would have, this number can be defined as the *occlusion rate*. We have already shown our algorithm can perfectly detect partial spheres acquired by the Kinect. Figure 3 shows the Euclidean distance error for the coordinates of the spheres at increasing occlusion ratios. The ratios are quantified by dividing the number of points on the occluded sphere by the number of points on the original non-occluded sphere. The root mean square (RMS) is given by:

$$RMS\ error = \sqrt{(x_G - x)^2 + (y_G - y)^2 + (z_G - z)^2} \tag{3}$$

where $(x_G, y_G, z_G)$ are the ground truth coordinates of the centre.The occlusion performance analysis was carried out on four point clouds of a synthetic sphere with no other objects in the cloud. These spheres had different point densities and the error values are the average from the four clouds.

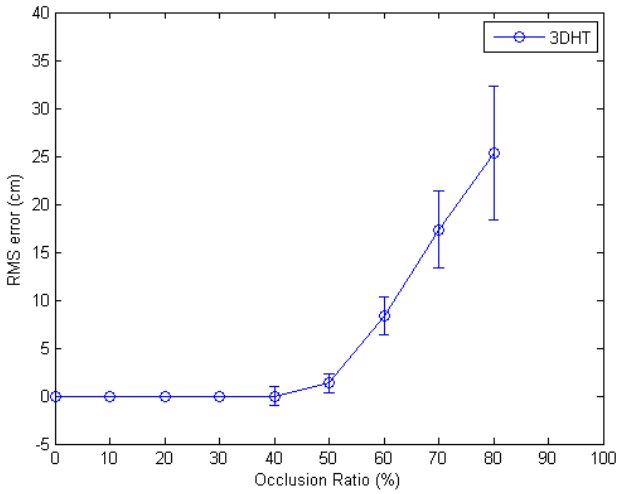


**Fig. 3.** RMS error metric of Algorithm 1 (3DHT) at increasing occlusion ratio.

2. The step size of the azimuthal angle $\theta$ and polar angle $\varphi$. So far, in all generated figures, we have used an angle step of 10 degrees for both angles when applying Algorithm 1, which gave perfect accuracy and satisfying results in terms of processing time. Having an angle step of $10\,^\circ$ means that for each point there will be $(36 \times 18)$ iterations in the casting process. Obviously, reducing the number of iterations leads to faster detection performance; the choice of an angle step is a trade-off between processing speed and accuracy.
3. The resolution of the accumulator space, or in other words the bins size that votes cast into. The larger the bin size is, the smaller the dimensionality of the accumulator, as multiple votes will vote in the same bin. Bin size is also a trade-off between memory storage and accuracy, and it does not have an effect processing time due to the implementation of a sparse matrix for the accumulator, in which the search time is always $O(1)$. Rounding to the nearest bin size significantly reduces memory storage requirements for the accumulator, however it comes on the expense of reduced accuracy in detecting the location in 3D space. In all our experiments on kinect point clouds we have used a bin size of $1cm^3$, and hence all detected centres are of $1cm$ precision.

## 6 Conclusions

We introduced a fast and accurate Hough Transform (HT) based algorithm for detecting spherical structures in 3D point clouds. We showed its robust ability to detect spheres in Kinect point clouds of typical environments. Our approach has been designed to be computationally efficient, reducing an established limitation of HT based approaches. We provided experimental analysis based on a number of factors, emphasising on performance against occlusion ratio and we showed superior performance to the only other HT based algorithm for detecting spheres in point clouds available in literature.

## References


1. Hough, P.: Method and Means for Recognizing Complex Patterns. U.S. Patent 3.069.654.(1962)
2. Rusu, R. B.: Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments. PhD thesis, Computer Science department, Technische Universitat Munchen, Germany. (2009)
3. Fischler, M. A. and Bolles, R. C.: Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Communications of the ACM 24(6), 381–395. (1981)
4. Schnabel, R., Wahl, R., and Klein, R.: Efficient RANSAC for Point-Cloud Shape Detection. Computer Graphics Forum 26(2), 214–226. Blackwell Publishing Ltd, (2007)
5. Borrmann, D., Elseberg, J., Lingemann, K., Nüchter, A.: The 3D Hough Transform for plane detection in point clouds: A review and a new accumulator design. 3D Research 2(2), 1–13 (2011)
6. Ogundana, O. O., Coggrave, C. R., Burguete, R. L., Huntley, J. M.: Automated detection of planes in 3-D point clouds using fast Hough transforms. Optical Engineering 50(5), 053609–053609 (2011)
7. Ogundana, O. O., Coggrave, C. R., Burguete, R. L., Huntley, J. M. Fast Hough transform for automated detection of spheres in three-dimensional point clouds. Optical Engineering 46(5), 051002–051002 (2007)
8. Rabbani, T., Van Den Heuvel, F.: Efficient Hough transform for automatic detection of cylinders in point clouds. ISPRS WG III/3, III/4, 3, 60–65 (2005)
9. Tarsha-Kurdi, F., Landes, T., Grussenmeyer, P.: Hough-transform and extended RANSAC algorithms for automatic detection of 3D building roof planes from lidar data. International Archives of Photogrammetry. Remote Sensing and Spatial Information Systems 36, 407–412 (2007)
10. Kotthäuser, T., Mertsching, B.: Triangulation-Based plane extraction for 3D point clouds. Intelligent Robotics and Applications, 217-228. Springer Berlin Heidelberg, (2012)