

AUTOMATED SPHERE SEGMENTATION FROM POINT CLOUDS

¹ Richard HONTI*, ² Ján ERDÉLYI, ³ Alojz KOPÁČIK

^{1,2,3} Department of Surveying, Faculty of Civil Engineering
Slovak University of Technology in Bratislava, Radlinského 11, 810 05 Bratislava
e-mail: ¹ richard.honti@stuba.sk, ² jan.erdelyi@stuba.sk, ³ alojz.kopacik@stuba.sk

Received 17 December 2019; accepted 17 February 2020

Abstract: Nowadays huge datasets can be collected in a relatively short time. After capturing these data sets the next step is their processing. Automation of the processing steps can contribute to efficiency increase, to reduction of the time needed for processing, and to reduction of interactions of the user. The paper brings a short review of the most reliable methods for sphere segmentation. An innovative algorithm for automated detection of spheres and for estimating their parameters from 3D point clouds is introduced. The algorithm proposed was tested on complex point clouds. In the last part of the paper, the implementation of the algorithm proposed to a standalone application is described.

Keywords: Automated data processing, Point cloud, Sphere segmentation, Laser scanning, Sphere fitting

1. Introduction

Today, point of clouds is becoming an increasingly common initial digital representation of real-world objects. Since point of cloud data often contains millions of 3D points, attempts have been made to automate the data processing. Point clouds have an important role in creating high-quality 3D models of objects in a variety of areas, e.g. interior (exterior) design, Building Information Modeling (BIM), urban information systems, documentation of objects, 3D cadaster, etc.

* Corresponding Author

Since, in most cases, the objects in the industrial environment consist of basic geometrical shapes, the detection and segmentation of these shapes can be one of the most important steps of data processing. Segmentation of geometric primitives from point cloud has been investigated thoroughly in several works, e.g. plane segmentation from the point cloud is described in [1], [2]; and the cylinder segmentation in [3], [4].

Other frequently occurring geometric shapes in manufactured objects are spheres. Identification and modeling of spheres have many applications in several fields, e.g. reverse engineering [5], [6]; medical imaging [7]; terrestrial laser scanner (alternatively camera) calibration [8], [9]; point cloud registration [10], [11]. Many applications require that the spheres are segmented from the point cloud data accurately and automatically, therefore the paper is devoted to this issue.

This paper shortly describes the reliable methods and approaches proposed for 3D point cloud sphere extraction and segmentation. It introduces then the algorithm proposed for the fully automated sphere segmentation and parameter estimation. The proposed algorithm is aimed at extracting spheres automatically, efficiently and accurately from point cloud data. The paper further presents the testing and the experimental results from the testing of the algorithm proposed and describes the implementation of the algorithm to a standalone application. The application development was carried out in MATLAB[®] software.

2. Sphere segmentation and fitting to point clouds

A sphere, in general, can be described by four parameters (*Fig. 1.*):

- the coordinates of the center of the sphere ($\mathbf{c} [c_x, c_y, c_z]$);
- the radius of the sphere (r).

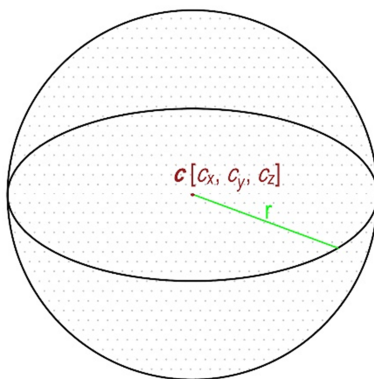


Fig. 1. Parameters of a sphere

Then, the equation of a sphere can be described as follows:

$$(x - c_x)^2 + (y - c_y)^2 + (z - c_z)^2 = r^2, \quad (1)$$

where x, y and z are the coordinates of the points on the surface of the sphere.

Sphere segmentation (extraction) from unorganized point clouds has been recently investigated based mainly on two approaches, the RANdom SAMple Consensus (RANSAC) (e.g. [1]) and the Hough transform (e.g. [12], [13]).

RANSAC is an iterative method for estimating the parameters of a mathematical model from an observed data set, which contains outliers. The algorithm works by identifying the outliers in a data set and estimating the desired model using data that do not contain outliers; therefore, the RANSAC paradigm extracts shapes from the point data and constructs corresponding primitive shapes, based on the notion of minimal sets [14]. This method is fast and efficient, but the limitation is that several threshold values are needed to be set. In the case of complex, unorganized and noisy point clouds with a set of outliers, it can be challenging. The next disadvantage is that the result of the segmentation depends on the random choice of the initial seed point, and in the worst-case RANSAC does not converge to a valid solution even when spheres are present in the point cloud. Another thing is, that RANSAC based approaches are mostly limited to detect only a single sphere at a time.

Another common method for geometric shape segmentation from the point cloud is the Hough Transform (HT) [15]. The HT is a well-known technique originally developed to extract straight lines, since then it has been extended to extract parametric shapes and non-parametric shapes in 2D images [16]. The main challenges of the HT based approaches are the memory requirements and computation time needed. Moreover, it is also a sequential method that cannot detect multiple spheres simultaneously [17].

The next step after prior segmentation of the subsets of points, that belongs to spheres in a point cloud is sphere fitting. In general, the methods for sphere fitting are divided into two categories: algebraic fitting [18] and geometric techniques [19]. Algebraic fitting minimizes the sum of algebraic distance errors and the geometric fitting minimizes the sum of orthogonal distance errors from points to the fitted primitive.

3. Algorithm for automated sphere segmentation and estimation

In this section, a robust algorithm for automated multiple sphere segmentation from the point cloud is proposed. The pseudo-code of the algorithm is shown in *Code 1*. The input data for the algorithm is a point cloud, the number of spheres expected in a point cloud, the threshold value for sphere fitting - distance filtering (t_d).

Code 1

Algorithm for automated multiple sphere segmentation from point cloud

➤ **Input:**
Point Cloud (in *.pts, *.xyz, *.txt);
number of spheres expected in the point cloud (n_{sphere});
a threshold value for distance filtering (t_d);

```

iter := 15 – maximum iteration number; sk := 0 – flag to check bad initial point.
for i = 1 : n_sphere
    while sk == 0
        1. Select a random initial point and its nearest neighbors.
        2. Calculate the initial sphere parameters  $\{c_{init} [c_x; c_y; c_z]; r_{init}\}$  from
           the selected points by fitting a sphere using equations (2)-(5).
        for j = 1 : iter
            3. calculate the distance of the points from the sphere detected.
            4. Extraction: updating the inlier points  $\Phi$  based on distance filtering.
            5. if iter > 4    &&    size( $\Phi$ ) < 200
                sk := 0
                break
            end
            6. if iter > 5    &&    convergence >  $\epsilon$ 
                sk := 0
                break
            end
            7. if iter > 13    &&    std_fitting >  $\vartheta$ 
                sk := 0
                break
            else
                sk := 1
            end
            8. Sphere Re-Fitting: the inlier points  $\Phi$  are used to refit the sphere
                $\{c [c_x; c_y; c_z]; r\}$  by fitting method described by equations (2)-(5).
        end
        9.  $\Phi$  – points of the sphere detected.
            $\{c [c_x; c_y; c_z]; r\}$  – parameters of the sphere detected.
    end
end
➤ Results:
set of parameters for the spheres detected  $\{c [c_x; c_y; c_z]; r; inlier_{number}; std_{fitting}\}$ ;
segmented point clouds ( $\Phi$ ) for each sphere.

```

The first step of the algorithm is a random initial seed point selection, i.e. the algorithm picks one point from the whole point cloud, where the calculations are started. Subsequently, the first sphere is fitted in 50 nearest neighbors using a fitting method, which minimizes the orthogonal distances of the points from the sphere, is an efficient method to compute the mentioned sphere parameters $(c [c_x, c_y, c_z], r)$ from a data set including only points belonging to a sphere.

The solution is based on the general equation of a sphere (1). The expanded and rearranged equation can be described as follows [20]:

$$x^2 + y^2 + z^2 = 2xc_x + 2yc_y + 2zc_z + r^2 - c_x^2 - c_y^2 - c_z^2. \quad (2)$$

The notation for equation (2):

$$\mathbf{f} = \begin{bmatrix} x_i^2 + y_i^2 + z_i^2 \\ x_{i+1}^2 + y_{i+1}^2 + z_{i+1}^2 \\ \vdots \\ x_n^2 + y_n^2 + z_n^2 \end{bmatrix}, \quad (3)$$

$$\mathbf{A} = \begin{bmatrix} 2x_i & 2y_i & 2z_i & 1 \\ 2x_{i+1} & 2y_{i+1} & 2z_{i+1} & 1 \\ \vdots & \vdots & \vdots & 1 \\ 2x_n & 2y_n & 2z_n & 1 \end{bmatrix}, \quad \mathbf{c}_{param} = \begin{bmatrix} c_x \\ c_y \\ c_z \\ r^2 - c_x^2 - c_y^2 - c_z^2 \end{bmatrix}, \quad (4)$$

where the \mathbf{f} vector in (3), the \mathbf{A} matrix and the \mathbf{c}_{param} vector in (4) represents the consolidated terms of the expanded sphere equation (2). The x_i, y_i, z_i represents the 3D coordinates of the data points.

$$\mathbf{f} = \mathbf{A} \cdot \mathbf{c}_{param}. \quad (5)$$

The Eq. (5) represents the over-determined system suitable for spherical fit. The vector \mathbf{c}_{param} contains the sphere parameters. The system is solved by using the augmented matrix of a system of linear equations, described in more detail in [20].

The first sphere fitting step provides initial sphere parameters used by the iterative fitting and extraction process. Extraction (4. in *Code 1*), i.e. updating the inlier points for the detected sphere are performed based on the distance filtering. Criteria for distance filtering are as follows:

$$|\Delta dist_i| < r \cdot t_{d[\%]}, \quad (6)$$

where r is the radius of the sphere, and t_d is the threshold value for the distance filter,

$$\Delta dist_i = \|p_i - c\| - r, \quad (7)$$

where $\Delta dist_i$ is the orthogonal distance of the testing point p_i to the sphere surface.

In the distance filter, the criterion is $r \cdot t_{d[\%]}$. Practically it means that if a sphere with a radius of 10 cm is considered, and t_d equals to 5, only the points, which are closer than 0.5 cm (5% of the detected sphere radius) to the surface of the sphere detected are considered as inliers for sphere fitting. Large values of t_d may cause adding some noisy points to the fitting process or some points, that not belongs to the detected sphere surface.

At each iteration, the points of the point cloud that satisfy the distance condition (6) are determined as inliers. The next step is the sphere fitting (8. in *Code 1*). The method described previously by equations (2) - (5) is used for refitting the sphere. At each iteration, the parameters of the sphere are recalculated using all the inliers from the

extraction step. So, by the process of the algorithm the parameters mentioned are more and more accurate.

These two consecutive steps, extraction and fitting, execute until all the remaining points on the sphere are detected or the break condition is met.

Generally, there are 3 break conditions that the algorithm determines:

1. there is a small number of inliers found for the randomly selected initial point (5. in *Code 1*);
2. sphere parameter convergence has not occurred (6. in *Code 1*);
3. the standard deviation of the fitting process is too large (7. in *Code 1*).

The first condition is true when there are less than 200 inliers for the selected initial point after the 4th iteration. The second condition is based on determining the differences in the sphere parameters in two consecutive iterations after the 5th iteration, this condition is true when the parameters differ more than the convergence parameter ($\epsilon = 10^{-4}$). The third condition ensures that the standard deviation of the fitting process is not too large. The standard deviation of the fitting is calculated based on the distances of the inlier points from the fitted sphere surface.

If one of these conditions is met, a new segmentation process with a new random initial point follows. These breaking conditions are necessary since the result of sphere estimation depends on the initial seed point and the surrounding area; sometimes the detected spheres are not valid and not represent any of the characteristic spheres of the scanned object. With the mentioned conditions these cases are eliminated.

Noise-free point clouds, with a small number of outliers, converge to a 'good sphere' after a few iterations, but noisy point clouds with outliers need more iterations to converge to the 'good' estimation. Therefore, the maximum number of iterations is set to 15, this number was determined empirically based on several point clouds with different complexity, noise, and the number of spheres.

The whole segmentation process is performed automatically until all the spheres of the point cloud are detected. The number of spheres is selected at the beginning of the algorithm.

In summary, the advantages of the proposed method are as follows:

- iterative extraction and fitting allow estimating the sphere parameters accurately;
- with the proposed extraction (inlier updating) step the outliers and the noise are sequentially removed from the estimation process;
- 'badly' estimated spheres are eliminated during the validation steps (5.-7. in *Code 1*);
- after an estimated sphere is considered as valid, the inliers are removed from the point cloud so, in the next segmentation phases the extraction is executed only on the remaining points, without the points belonging to the previously segmented spheres;
- the extraction step is performed on the whole point cloud at once, this step significantly reduces the processing time.

3.1. Testing of the proposed algorithm

For the experimental testing of the above-described algorithm, a point cloud (Fig. 2) that contains approx. 2 million points and 3 sphere objects (reference spheres for laser scanners) was used. The reference spheres were manufactured with a diameter of 200 mm. For the scanning, a Trimble TX5 3D laser scanner was used. With the mentioned scanner and considering the conditions (maximum distance of the scanned objects from the scanner, resolution and quality setting of the scanner, etc.) during the measurement, the accuracy of a single measured point was less than 2.5 mm in any cases.

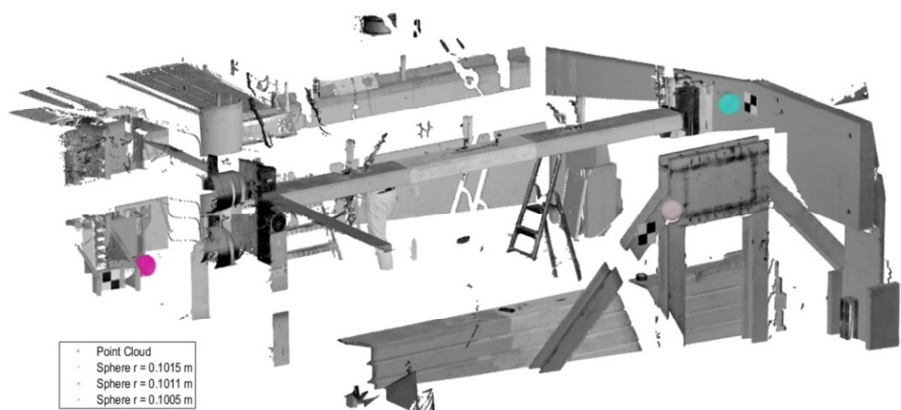


Fig. 2. Point cloud from a static load test

For verification of the results of the segmentation process, the differences (Table I) among the known parameters (radius) and the estimated parameters from the algorithm were calculated.

Table I

Comparison of the sphere parameters

	Reference Values [m]	Values from the Algorithm [m]	Difference [mm]
r_1	0.1015	0.1000	1.5
r_2	0.1011	0.1000	1.1
r_3	0.1005	0.1000	0.5

The differences between the known and the estimated parameters (radiuses) are 1.5 mm, 1.1 mm and 0.5 mm. In these deviations, the measurement error, the instrument systematic errors, the effect of the environmental conditions is also included. The fact, that the scanning process was executed only from one position of the scanner, therefore the whole surfaces of the spheres were not covered with points, has also a negative

impact on sphere parameter estimation. Moreover, the reference spheres are also manufactured with some tolerance.

Fig. 3 shows the segmented points of the individual spheres differentiated by color after the segmentation process. *Fig. 4* shows the photographs from the experiment. The standard deviation of sphere fitting was less than 0.8 mm in any cases. The sphere parameters are shown in the dialog window table in *Fig. 5*.

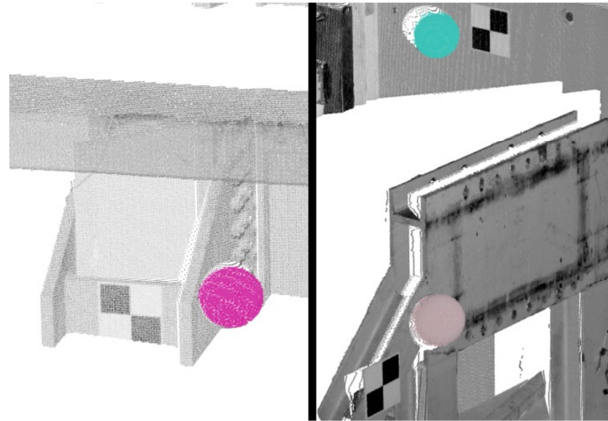


Fig. 3. Details of the sphere segmentation result



Fig. 4. Photographs of the spheres from the experiment

Based on the mentioned comparison and *Fig. 2* and *Fig. 3* the verification was successful.

3.2 Application development

For the automation of the segmentation procedure described a computational ‘*Sphere Segmentation*’ application was developed. The graphical user interface of the application was designed by MATLAB[®] software. The application was created as a stand-alone application; however, MATLAB runtime is needed for its execution.

The dialog window of the application consists of 3 main sections. In the first section ‘*Input File and Parameters*’ serves for the point cloud file loading. The first step is the parent directory selection, where the input point cloud file is saved, and the resulted segmented point clouds will be exported. The next step is loading the point cloud file in the *.txt, *.xyz or *.pts file format, that contains the spatial coordinates of the points scanned.

Next, the input parameters are chosen, so the number of spheres expected (the predefined value is 1) in the point cloud and the threshold value for the distance-based (eq. (6) and eq. (7)) filtering (the predefined value is 5). The segmentation process is launched by pressing the calculate button.

According to the results, the individual point clouds for the estimated spheres are saved into a *.txt file in a new folder ‘*Results*’ in the parent directory, and the sphere parameters are shown in the second section of the dialog window, in the ‘*Sphere Parameters*’ dialog window table (Fig. 5, left-bottom). The dialog window table contains the IDs of the spheres, their estimated parameters (3D coordinates of the sphere center, radius of the sphere), the number of inliers for each sphere, and the standard deviation of the fitting. A figure (Fig. 5, right), which shows the original point cloud and the separate point clouds for the detected spheres, is created for consistent monitoring of the application process, so the user can visually check whether the individual spheres are correctly segmented.

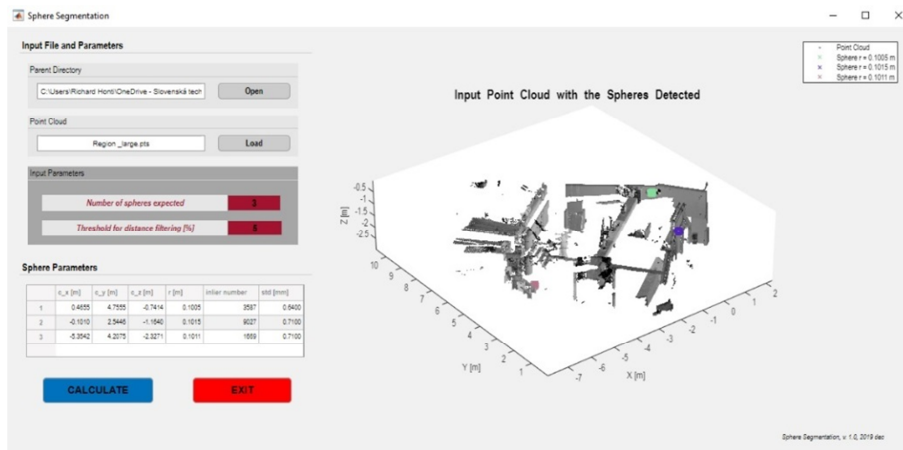


Fig. 5. Dialog window of the ‘Sphere Segmentation’ application

4. Conclusion

Spheres are popular geometric primitives found in many manufactured objects, and they have many applications in several fields. In many of the applications, automation of the sphere detection and segmentation from a point cloud data is required.

A novel robust algorithm for automation of the sphere segmentation is proposed. The algorithm can detect and extract subsets of points belonging to spheres from a complex, noisy, unstructured point clouds with outliers using distance filtering. The validation steps ensure that the segmented spheres are valid. The experimental testing of the proposed algorithm on a large point cloud with approx. 2 million points is also described. For automation, the algorithm was implemented to a stand-alone application. The user can easily execute the sphere segmentation from point clouds with the created application. Only two input parameters are needed to be defined for the execution, the number of spheres expected, and the threshold value for distance-based filtering. The results of the application are the segmented subsets of points saved into the text files for every sphere separately and the parameters of the spheres calculated using geometric fitting shown in the table of the application's dialog window. The robustness and accuracy of the results suggest that the algorithm can be used in several applications.

The future work will be to add an inlier point selection strategy based on curvature computation, which will boost the efficiency of the algorithm and will decrease the computation time. Furthermore, to apply and test the proposed algorithm on point clouds of complex scenes from several areas containing many full or partial spherical components is also planned.

Acknowledgments

This work was supported by the Slovak Research and Development Agency under Contract no. APVV-18-0247.

References

- [1] Schnabel R., Wahl R., Klein R. Efficient RANSAC for point-cloud shape detection, *Computer Graphics Forum*, Vol. 26, No. 2, 2007, pp. 214–226.
- [2] Honti R., Erdélyi J., Kopáčík A. Plane segmentation from point clouds, *Pollack Periodica*, Vol. 13, No. 2, 2018, pp. 159–171.
- [3] Tran T. T., Cao V. T., Laurendeau D. Extraction of cylinders and estimation of their parameters from point clouds, *Computers & Graphics*, Vol. 46, 2015, pp. 345–357.
- [4] Honti R., Erdélyi J., Kopáčík A. Automation of cylinder segmentation from point cloud data, *Pollack Periodica*, Vol. 14, No. 3, 2019, pp. 189–200.
- [5] Várady T., Martin R. R., Cox J. Reverse engineering of geometric models - an introduction, *Computer-Aided Design*, Vol. 29, No. 4, 1997, pp. 255–268.
- [6] Benkő P., Várady T. Segmentation methods for smooth point regions of conventional engineering objects, *Computer-Aided Design*, Vol. 36, No. 6, 2004, pp. 511–523.
- [7] van der Glas M., Vos F. M., Botha C. P., Vossepoel A. M. Determination of position and radius of ball joints, *Proceedings SPIE 4684, Medical Imaging, Image Processing*, San Diego, California, United States, 9 May 2002, pp. 1571–1575.

- [8] Wang L., Cao J., Han C. A calibration algorithm for 3D laser scanner based on spatial sphere, *Journal of Xi'an Jiaotong University*, Vol. 47, No. 4, 2013, pp. 79–85.
- [9] Agrawal M., Davis L. S. Camera calibration using spheres: a semi-definite programming approach, *Proceedings of the Ninth IEEE International Conference on Computer Vision*, Nice, France, 13-16 October 2003, Vol. 2, pp. 782–789.
- [10] Wang Y., Shi H., Zhang Y., Zhang D. Automatic registration of laser point cloud using precisely located sphere targets, *Journal of Applied Remote Sensing*, Vol. 8, No. 1, 2014, Paper No. 083588.
- [11] Huang J., Wang Z., Gao J., Huang Y., Towers D. P. High-precision registration of point clouds based on sphere feature constraints, *Sensors*, Vol. 17, No. 1, 2017, pages 1–14.
- [12] Camurri M., Vezzani R., Cucchiara R. 3D Hough transform for sphere recognition on point clouds, *Machine Vision and Applications*, Vol. 25, No. 7, 2014, pp. 1877–1891.
- [13] Ogundana T., Coggrave C. R., Burguete R., Huntley J. M. Fast Hough transform for automated detection of spheres in three-dimensional point clouds, *Optical Engineering*, Vol. 46, No. 5, 2007, Paper No. 051002.
- [14] Fischler M. A., Bolles R. C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, *Communications of the ACM*, Vol. 24, No. 6, 1981, pp. 381–395.
- [15] Duda R. O., Hart P. E. Use of the Hough transformation to detect line and curves in pictures, *Communications of the ACM*, Vol. 15, No. 1, 1972, pp. 11–15.
- [16] Abuzaina A., Nixon M. S., Carter J. N. Sphere detection in kinect point clouds via the 3D Hough transform, in: Wilson R., Hancock E., Bors A., Smith W. (Eds) *Computer Analysis of Images and Patterns*, ser. Lecture Notes in Computer Science, Vol. 8048, Springer, 2013, pp. 290–297.
- [17] Tran T. T., Cao V. T., Laurendeau D. eSphere: extracting spheres from unorganized point clouds, *The Visual Computer*, Vol. 32, No. 10, 2016, pp. 1205–1222.
- [18] Pratt V. Direct least-squares fitting of algebraic surfaces, *ACM SIGGRAPH Computer Graphics*, Vol. 21, No. 4, 1987, pp. 145–152.
- [19] Forbes A. B. Least-squares best-fit geometric elements, *National Physical Laboratory Report*, Vol. 140, 1989, London, UK, pages 1–30.
- [20] Jekel C. F. Digital image correlation on steel ball (Appendix A), in *Obtaining non-linear orthotropic material*, (Diploma Thesis) Stellenbosch, South Africa, Stellenbosch University, 2016.