ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume X-5/W2-2025
Unleashing the power of Geospatial & Frontier Technologies for a Sustainable Future – GIFTS Summit 2025,
1–3 September 2025, Symbiosis International University, Pune, India

Automatic Building Boundary Extraction from Point Cloud Data

Jagadeesan Nalini[a], Subbarayan Saravanan[b] , Bommineni Narender[a] , Swaminathan Muralikrishnan[a]

[a]National Remote Sensing Centre, ISRO, Hyderabad-500037, India - (nalini_j, naren_br, muralikrishnan_s) @nrsc.gov.in
[b]Department of Civil Engg, National Institute of Technology,Tiruchirapalli, India – (ssaravanan@nitt.edu)

KEY WORDS: Point Cloud, LiDAR, Satellite, Building Footprint, Boundary, Corner Points

ABSTRACT

This paper presents a methodology and solution for extracting building footprints from three-dimensional point cloud data. The extraction and normalisation of building boundaries represent a fundamental approach for three-dimensional modelling of buildings for urban mapping. The automatic building boundary extraction utilising dense point cloud data has emerged as a prominent and labour intensive subject in lidar and photogrammetric point clouds. The methodology comprises three principal stages: Roof Face Plane Detection, Boundary Detection, Outline Smoothing, and Footprint Generation. The building point clouds are segregated from surface point clouds by filtering the bare earth model and removing above-ground objects other than buildings. Rooftop structures are generated by roof face plane detection. An extracted rooftop point cloud is subsequently traced to produce a series of corner points. The regularisation phase is conceived to extract corners from the irregular boundary and yield a polygon that accurately represents a rectilinear building footprint. To assess the performance, centroids of manually obtained building footprints are compared with centroids of automatically generated footprints. Euclidean distance between centroids of up to 2m is achieved for 90% buildings derived from the airborne lidar point cloud. Most buildings have an Euclidean distance variation between 4m and 8m for high-resolution stereo satellite data. The footprints generated from airborne point cloud data exhibit a smooth texture and closely resemble the manually derived building footprints in comparison to those obtained from satellite point clouds. The results indicate that automatic building footprints predominantly rely on the density of the point clouds.

1. Introduction

1.1 Building Boundary Extraction from point cloud

Building boundary from remote sensing data is a prerequisite for many GIS (Geographic Information System) applications, such as urban planning and 3D city modelling, disaster management (Alrajhi et al.,2016). Building is the main element to form a city. Building boundary is the fundamental of urban mapping and 3D building modelling(Hu, J et al.,2003, Ribarsky et al.,2002).

The complete solution to automatic building extraction from satellite or airborne LIDAR has three parts.

The first part involves generating dense point cloud data from satellite stereo images. The RPC (Rational Polynomial Coefficient) model, the GCP (Ground Control Point), and the Triangulation techniques are used to generate epipolar stereo images. Finally, the Digital Surface Model (DSM) uses semi-global matching techniques to provide dense point cloud data.Very dense point cloud data is strictly recommended to get high accuracy. Dense point clouds from airborne LiDAR are also used with a point density of 3 points/m2.

The second part involves a segmentation technique where each building is identified as a subset of the input points set. This segmentation technique separates the non-ground points (mainly buildings and trees) from the ground points. Further, filtering or the vegetation index removes the trees from the non-ground points (Alobeid et al). The final point cloud should contain only buildings.

The third part generates the individual building outline from its point set (Outline Extraction). Boundary edges that delineate the perimeter are initially recognized. Each edge constitutes a linear segment that links two sequential boundary coordinates. Subsequently, they are delineated to create an irregular architectural perimeter. Finally, the irregular boundary is regularised to form a regular outline.

This paper concentrates on part 3, building outline extraction from a point cloud which contains buildings points only.

1.2 Existing Algorithms

Segmentation and Preprocessing: The initial step often involves creating Digital Surface Models (DSM) and Digital Terrain Models (DTM) to differentiate between ground and non-ground points, which helps in isolating building structures from vegetation and other elements (Erdem & Anbaroglu, 2023).

Clustering Algorithms: Techniques like DBSCAN are used to segment LiDAR points into clusters, each representing an individual building. This segmentation is crucial for further processing and outline extraction (Li et al., 2022).

For Outline Extraction the works carried out as explained below

Generative Adversarial Networks (GANs): A parameter-free method using GANs has been proposed to extract building outlines from gridded binary images, overcoming the challenge of parameter selection and improving the extraction of concave shapes (Kong et al., 2022).

Recursive Convex Hull Algorithm: This method involves extracting the outermost points of each building cluster and applying a recursive convex hull algorithm to generate initial building outlines, which are then regularized to match true building boundaries (Li et al., 2022).

Regularization and Boundary Refinement are discussed in the following papers

* Corresponding author: J.Nalini, jpnalini77@gmail.com

Signal-Based Regularization: After initial outline extraction, signal-based regularization is applied to transform the outline into a signal, which is then denoised to identify corners and generate a regularized building footprint (Li et al., 2022).
Feature-Based Techniques: Regularization can also involve feature-based techniques that adjust corners and line segments based on assumptions about building geometry, such as parallel or perpendicular sides (Awrangjeb & Lu, 2014).
Robustness and Accuracy of the generated building footprint are explained in detail by the papers mentioned below
Robust Algorithms: Techniques like MCMD and MSAC are employed to enhance robustness against outliers and irregular data distributions, ensuring reliable footprint extraction even in challenging conditions (Nurunnabi et al., 2022).
Multimodal Approaches: Combining LiDAR data with orthoimages using deep learning frameworks can improve accuracy by leveraging complementary information from different data sources (Vostikolaei & Jabari, 2024).
Challenges and Considerations
Point Density Requirements: The accuracy of building footprint extraction is influenced by the density of the LiDAR point cloud.A minimum density of 4 pts/m² is generally sufficient for vendor-classified data, while re-classified data may require higher densities (Proulx-Bourque et al., 2021).
Outlier and Noise Management: Techniques like KDE and DBSCAN are used to manage outliers and noise, which are common in point cloud data, to ensure clean and accurate footprint extraction (Rottmann et al., 2022).
The new method to extract the building boundary with the help of Alpha Shapes is explained by Shen Wei (2008), "Building Boundary Extraction based on LIDAR point clouds data". He used the alpha shapes to detect the boundary points of individual building. For smoothing the outline the improved pipe algorithm is used.
The concept of Alpha Shapes which is helpful to detect the boundary of the building is also explained in "M. Awrangjeb (2016) - Using point cloud data to identify, trace, and regularize the outlines of buildings.
F. Nex, F. Remondino (2012), explained algorithm to extract building boundary from Digital Surface Model (DSM) in their paper.
In this paper the use of 3D alpha shapes to detect the point cloud of individual building from point cloud of city is explained. The Convex Hull algorithm is used to detect the boundary points of individual building. To smoothen the outline of building roof top, the regularization step is developed to extracts corners from the irregular boundary and obtain a polygon representing a rectilinear building footprint. Along with proposed algorithm improved pipe algorithm is also used to improve the accuracy of result obtained. To evaluate the performance; manually obtained building footprint is compared with automatically generated footprint. Results show that building footprints mainly depends on point cloud density.

1.3 Study area and Data

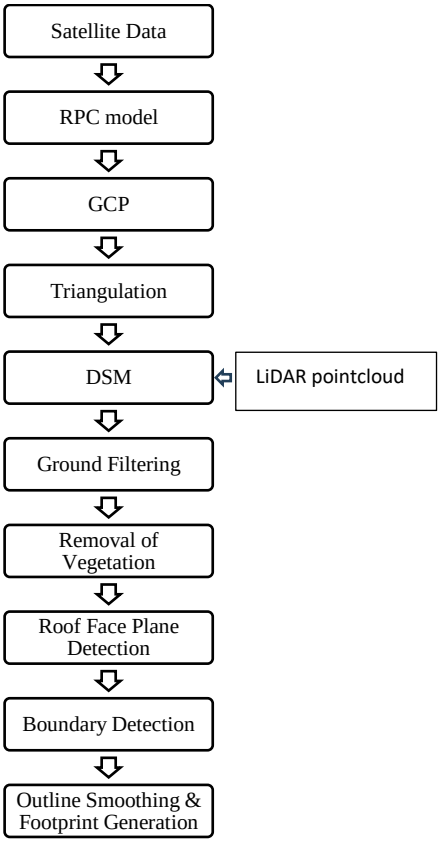
Hyderabad is the fifth largest metropolis of India and capital of the state of Telangana. It lies on the Deccan plateau, 541m above sea level over an area of 625sq km.
The study area covers 8 square kilometers of Hyderabad city. It lies in 17° 21′ 57.6″ N Latitude, 78° 28′ 33.6″ E Longitude.

Airborne LIDAR pointcloud acquired during Jan 2025 at point density of 3-4 pts/m2 and Geoeye satellite data of 50 cm GSD from 2016 pertaining to the study area is used here.

2. Methodology

The overall methodology to obtain final footprint from airborne LiDAR and satellite images can be explained in following workflow model. This paper mainly focuses on Roof Plane Detection, Boundary Point Detection, Outline Smoothing and final Footprint Generation. The last three processes are explained in detail along with the results obtained.

Workflow:



RPC: Rational Polynomial Camera Model
GCP: Ground Control Point
DSM: Digital Surface Model.

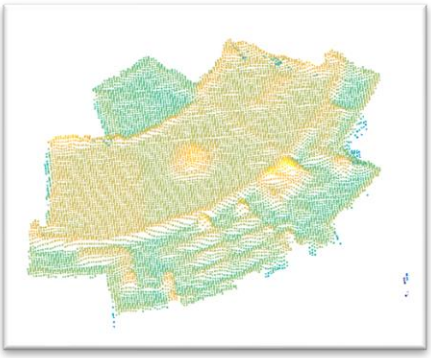


Figure 2.1. Point cloud data of single building (LIDAR)

The point cloud containing only buildings is obtained after removal of vegetation from DSM model. The 3D point cloud of single building is shown in Figure 2.1. This point cloud of the single building is obtained from LiDAR pointcloud. From the satellite images point cloud of the city is obtained is shown in Figure 2.2. Point cloud obtained from aerial images is denser than point cloud obtained from satellite images.

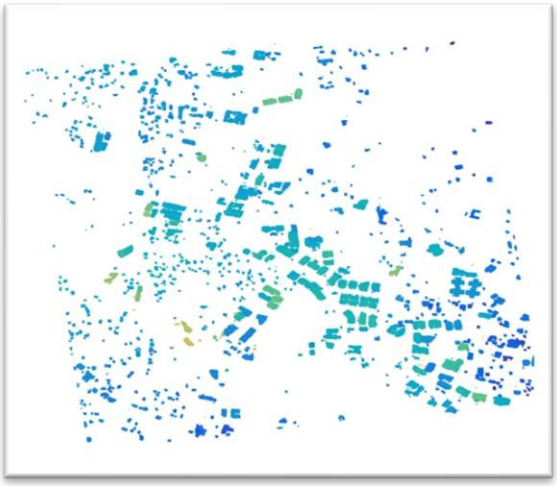


Figure 2.2. Point cloud data of city (satellite image)

3. Roof Plane Detection

3.1 Alpha Shapes

The use of alpha shapes in large point cloud data is explained in research paper 'Building Boundary Extraction based on LIDAR Point Clouds Data' by Shen Wei (2008) . In computational geometry, an alpha-shapes or α-shape, is a family of piecewise linear simple curves in the Euclidean plane associated with the shape of a finite set of points. They were first defined by Edelsbrunner, Kirkpatrick & Seidel (1983).

To detect the cluster of each building in large point cloud data, alpha shapes can be used. It can be used to perform the boundary reconstruction from the irregular point clouds in 2D or 3D space. As depicted in Figure 3.1 the set S of points (2D point cloud) has an α-shape in polygon. This polygon is determined by the S and α. α-shape directly illustrates the S shape, at the same time, the precision of the boundary is control by parameter α.

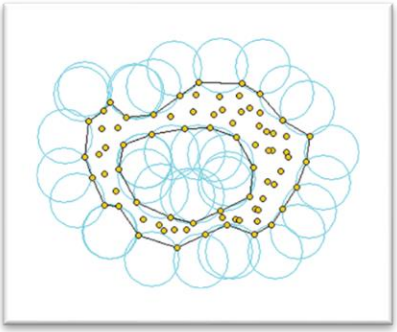


Figure 3.1. Alpha shapes algorithm

Figure 3.1 shows an example of the Alpha Shapes Algorithm. Let S be the point set in 2D space as shown in Figure 3.1. A circle with a radius α is rolling around the point-set S. If α value is bigger than a threshold value then the circle won't fall into the area of S.the boundary of this point-set S is formed along with the rolling track of circle in point cloud. α-shape must be the convex hull, when the α value is approaching infinity (α→infinity). On the other hand, every point might be the boundary when the α value is very small (α→0). The α-shape can extract the inner and outer outlines of the polygon at the same time, when the point-set S containing evenly distributed points and α approaching an optimal value, as shown in Figure 3.1. The earliest Alpha Shape concept was been applied to 2D space and now it has been developed into 3D space lately.

In 3D point cloud, instead of circle a sphere of radius α rolls around the point cloud. In 2D point cloud alpha shape gives a line segment while rolling in point cloud, since at every time it detects two boundary points. In 3D point cloud three boundary points are detected in each roll of sphere in the point cloud. 3D alpha shapes gives boundary points of each building in 3D space. 3D boundary points of buildings can be consider as cluster of each building. To detect the the cluster of individual building from point cloud of city is the main use of 3D alpha shapes.

Figure 3.2 shows the 3D alpha shapes of the individual buildings obtained from the building point cloud of city shown in Figure 2.2. Figure shows small part of city (some buildings) taken from the city model.

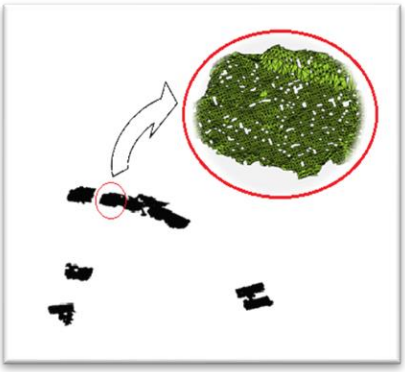


Figure 3.2. Alpha shapes of buildings

4. Boundary Points

The results obtained from roof plane detection still contain 3D point cloud but having only the roof top of buildings. To detect the boundary points of the roof plane the 2D point cloud is needed. The 'mean' of the Z co-ordinates (height axis) of the building is calculated to set the roof top in 2D plane at particular height in Z axis. Mode as well as median can be used to make the roof of building in 2D. To find out the boundary points of the 2D plane Convex Hull is used.

4.1 Convex Hull

Given a set of points in the plane, the convex hull of the set is the smallest convex polygon that contains all the points of it. Convex hull is good for detecting the boundary regions of the point cloud. But it is not so good in describing shapes. Convex hull is an element of an alpha shape family. When alpha is equal to infinity it calculates the convex hull of point set. There is major difference in convex hull and alpha shapes when more than one clusters are present in the point cloud. Figure 4.2

shows the difference of convex hull and alpha shapes when more than one clusters are present.

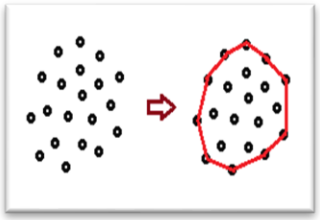


Figure 4.1. Convex Hull

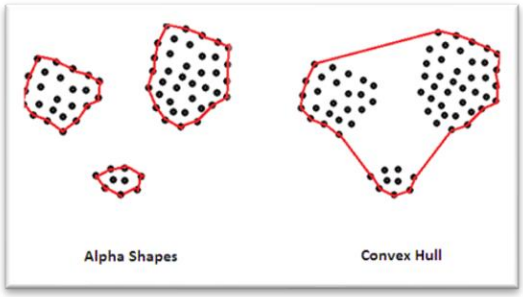


Figure 4.2. Difference between convex hull and alpha shapes

Additional parameter is taken into consideration while calculating boundary points is the shrink factor. Shrink factor is a scalar between 0 and 1. Setting shrink factor to 0 gives the convex hull, and setting it to 1 gives a compact boundary that envelops the points. When so many corners are present in the building roof top. Convex hull does not give compact boundary points containing all corners. With the help of shrink factor boundary points containing all corners can be extracted.

Figure 4.3 shows the boundary points obtained after applying convex hull algorithm to the 2D plane of the roof top of single building (Figure 2.1) with shrink factor equals to 0.7. Figure 4.3 shows the boundary points of roof top as well as roof plane points at constant Z axis obtained by taking mean of the 3D boundary points.

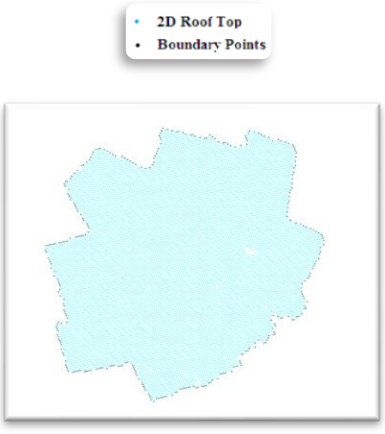


Figure 4.3. 2D Plane of roof top with boundary points

5. Outline Smoothing & Footprint Generation

When the boundary points are joined together irregular shape of the building roof is obtained as shown in Figure 5.1a. The main challenging task is to make it smooth and straight outline of building as shown in Figure 5.1b. To make the outline of building smooth an algorithm was developed which can detect the corner points of the building. The developed algorithm is explained in detail below.

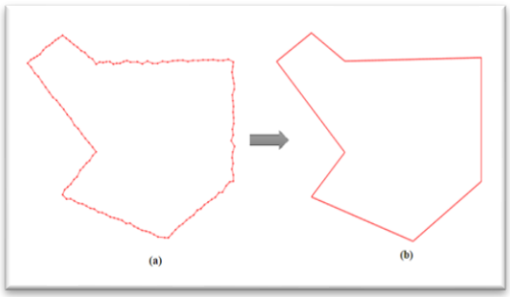
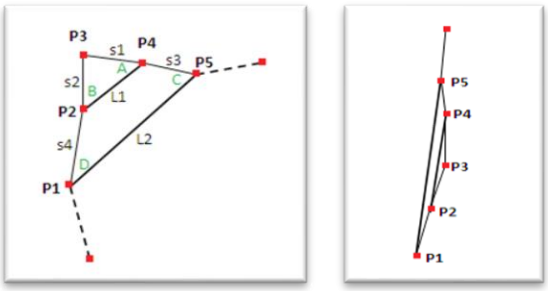


Figure 5.1. Irregular boundary to smooth boundary

5.1 Proposed Algorithm:

The proposed algorithm retrieves the corner points of polygon by judging the slopes of the line segments obtained by joining the boundary points surrounded by the current point.
Let us consider P1, P2, P3, P4, P5 are five consecutive points of boundary as shown in Figure 5.2. Let point P3 (middle point) be the current point. Figure 5.2a shows boundary points which form corner of the building and Figure 5.2b shows boundary points which form straight line. To check whether the current point is corner point or not, line segments s1, s2, s3, s4, L1, L2 and their slopes were considered.



(a) Corner of building (b) Straight wall of building

Figure 5.2. Boundary Points

The slope of the line segment s1 is easily calculated by-

$$M11=|(Y_{P4} - Y_{P3})|/|(X_{P4} - X_{P3})|$$

Modulus of the slope value is taken to keep the angle made by line segment with X axis in between 0 to 90 degree.
After calculating the slopes, the angle made by line segment with X axis is calculated by taking the inverse tangent of the slope. When the current point is corner point, there is specific angle (30 to 60 degree) made by line L1 with s1 and s2 as shown in Figure 5.2. Consider angle A between L1 and s1 and angle B between L1 and s2. If the angle A or B is in between 30 to 60 degree, then consider P3 as corner point. The use of second point in forward and backward direction of current point is comes in the case shown in Figure 5.3. In this case the angle A or B might be less than 30 degree. But the angle C or D is in between 30 to 60 degree. So by considering the second point we won't miss the corner point in this particular case.

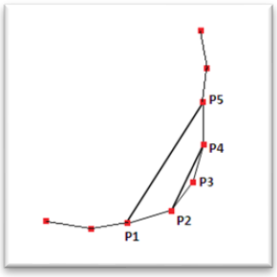


Figure 5.3. Curve corner boundary points

5.1.1 The general step by step overview of algorithm:

1. Calculate slopes of line segments
2. Calculate angle A, B, C, and D as discussed above
3. If 30<A<60 || 30<B<60 ||30<C<60 || 30<D<60
   i. Consider the current point as corner point
   ii. Save the position of the corner point in new array
4. Repeat the step 1 to 3 until we covered all boundary points

The position of the corner point is saved in an array. If any peaks or valley are present in boundary point, above algorithm detects it as corner points (Figure 5.4). In case of curve corner case (Figure 5.3) all three points P2, P3, P4 can be detected as corner points. The problem is solved as follows.

Let the 2nd, 6th, 7th, 8th and 20th points of the boundary points are detected as corner by above algorithm. 6th, 7thand 8th points are consecutive points detected as corners. All three points can't be the corners of the building in both cases shown in Figure 5.4.The angle (V) made by line segment joining the points 2 and 6 with X axis is calculated. Also angle (W) made by line segment joining the points 8 and 20 with X axis is calculated. The difference between these angles has to be greater than 30 degree to consider it as corner point. In the curve corner case out of 6th, 7th, 8th points only 7th point is considered as corner point and 6th and 8th points has to be deleted. If difference between angles V and W is less than 30 degree (peaks and valley case) all three 6th, 7th, 8th points has to be deleted.

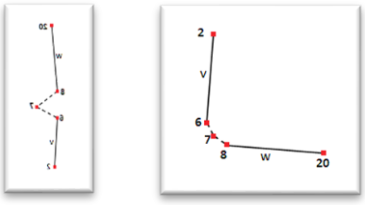


Figure 5.4. Corner points detected

5.1.2 The overview of algorithm:

1. Detect the consecutive points in position array of corner points detected earlier.
2. Consider it as set of points.
3. Calculate the angle (V) made by line joining previous point and starting point of set, also angle (W) made by line joining next point and end point of set.
4. Difference (diff) = abs(V-W)
5. If diff > 30 degree consider middle point of set as corner point and delete other points of set.
6. If diff < 30 degree, delete all the points in the set. (peaks and valley case)
7. Repeat the step 3 to 6 until we covered all corner points detected previously.

Along with the above proposed algorithm improved pipe algorithm is used. This algorithm was proposed by Shen Wei (2008).

5.2 Improved Pipe Algorithm

Improved pipe algorithm judges the direction changes of pipe and detects the corner points of polygon. A pipe with diameter *d* has been used to interpret the changes of boundary as shown in Figure 5.5.

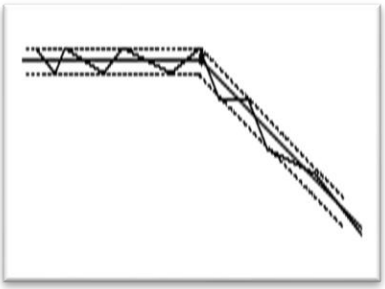
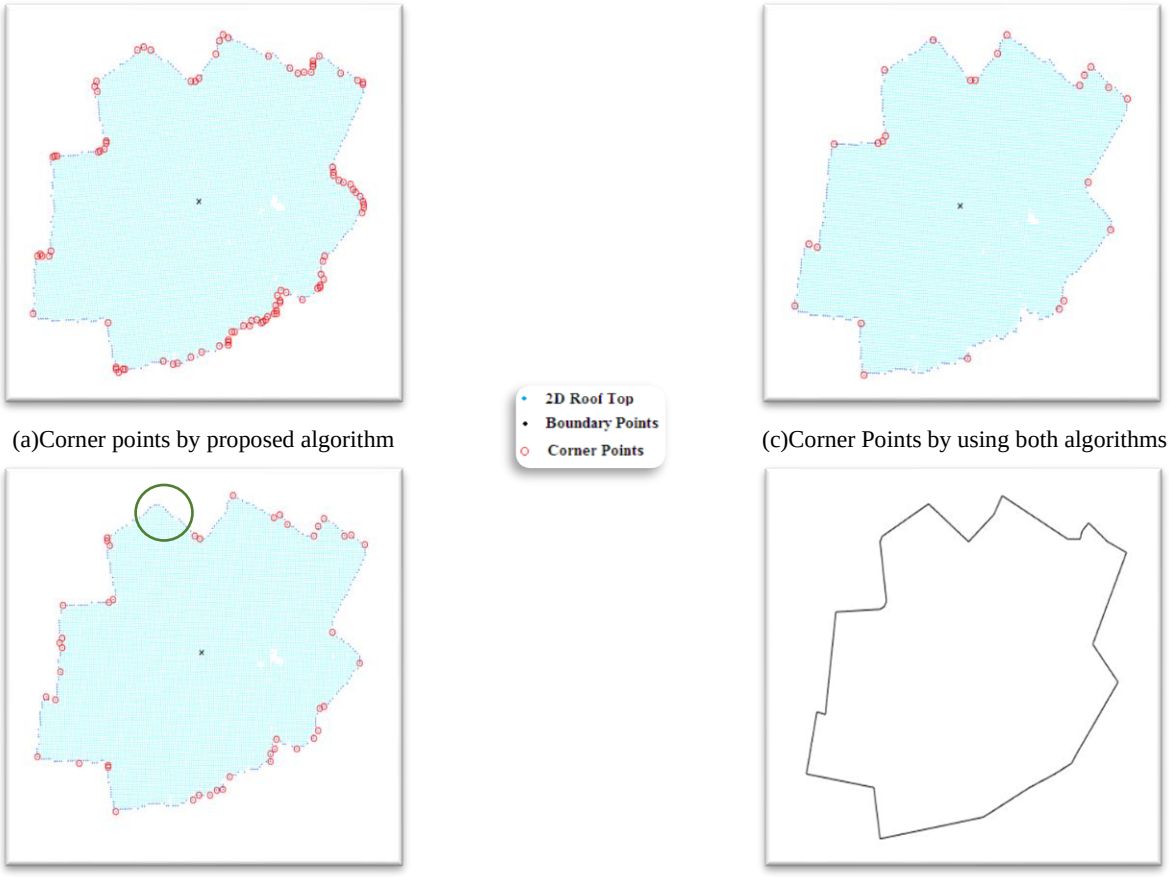


Figure 5.5. Pipe Algorithm

If difference between the points on the boundary line is less than d, it means there are no essential changes on the boundary where the intermediate points can be removed. However, if the difference between the points on the boundary is more than d, it indicates that the point is corner point and should be retained. Building boundary points can be processed in flexible way where only related points are considered for calculation and this is the main advantage of pipe algorithm. The algorithm not only considers the azimuth changes but considers the overall changes of the building shape. So, the azimuth changes of line will not affect the overall boundary, however it helps to pick up the corner points (Shen Wei., 2008).



(a)Corner points by proposed algorithm

(c)Corner Points by using both algorithms

(b) Corner points by improved pipe algorithm

(d) Final footprint of rooftop

Figure 5.6. Building Roof Top

Figure 5.6 shows the corner points obtained by different algorithm. The corner points obtained by applying proposed algorithm are shown in Figure 5.6a. By using only proposed algorithm, corner points are not precise as it gives irregular outline of building. The corner points obtained by applying only improved pipe algorithm are shown in Figure 5.6b. Improved pipe algorithm reduced the roughness of outline but one important corner point of the building (green circle in Figure 5.6b) is not detected by this algorithm. Figure 5.6c shows the corner points obtained by applying both algorithms one after another (first proposed algorithm and then improved pipe algorithm).The result obtained by both algorithms is more precise than single algorithm. Final footprint of building rooftop by joining the corner points is shown in Figure 5.6d. These algorithms are used to obtain both smoothing and rectilinear output of building.

Figure 5.7 shows 2D plane of the roof top of the buildings at constant height (constant Z value). It shows the boundary points obtained by convex hull algorithm and shrink factor equal to 0.7. The boundary points are irregular in shape. The Figure also shows the corner points of the individual building in red circles. The corner points were obtained by using both proposed and improved pipe algorithm one after another.

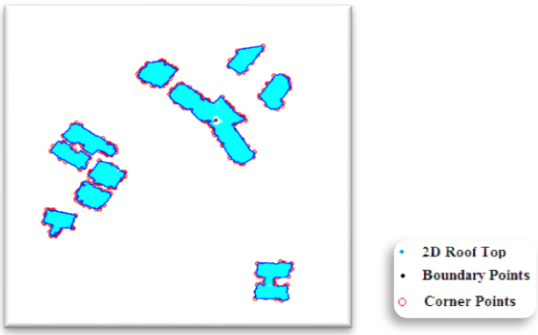


Figure 5.7. 2D Planes of Individual Building

The corner points detected were not smooth. This is mainly due to sparse point cloud of buildings. The dense point cloud is recommended to obtain very smooth outline of the building. The final footprints of the buildings obtained are shown in Figure 5.8.

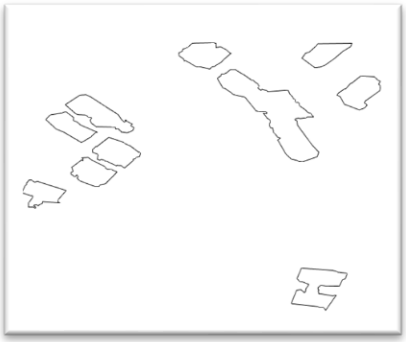


Figure 5.8. Final footprint of the buildings of one block of city model

The footprints of the individual buildings of city are shown in Figure 5.9.

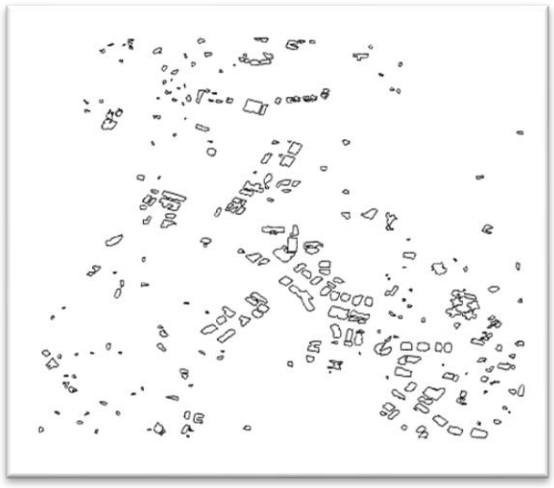


Figure 5.9. Final footprint of the buildings

6. Accuarcy Assessment

Quantitative Assessment of generated 3D buildings using automated methods are verified using a manual footprint.

Accuracy in planimetry and height of building footprints are evaluated using Euclidean distance between centroids and heights of buildings.

6.1 Euclidean distance between centroids

To verify the results and to ensure the algorithm implementation to obtain building footprints in real world accuracy assessment is very important. Although there are many researches on 3D photogrammetric point clouds data processing, however there are very little studies about the accuracy of the extraction and modelling. Euclidean distance plays a vital role in geospatial analysis by quantifying spatial relationships and enabling precise measurements between points, which is essential for effective building footprint delineation. Incorporating Euclidean distance metrics can facilitate a more specific understanding of spatial relationships, ultimately defining the accuracy of building footprint delineation in urban context.

In this paper to check the accuracy of the automated methods the centroids of polygon obtained by manual and automated methods are compared using the following formula.

$$c_x = 1/6A\sum^{N-1}(x_i+x_{i+1})\,(x_iy_{i+1}-x_{i+1}y_i)$$

$$c_y = 1/6A\sum^{N-1}(y_i+y_{i+1})\,(x_iy_{i+1}-x_{i+1}y_i)$$

Where $\mathbf{A}$ = Area of polygon ,x_i,y_i = Vertex coordinates of polygons, c_xc_y= Centroid of polygon.

Figure 6.1 shows the centroid of polygon generated using manual methods.

Figure 6.1. Centroid of polygon from manual building footprint

The Table 1 below and histogram in Figure 6.2 depicts the Euclidean distance between the centroids of manual and automatic building footprints from satellite data.

Euclidean distance(m)	Number of Polygons
0-2	10
2-4	90
4-6	300
6-8	189
8-10	71
10-12	39

Table 1. Euclidean distance variation with number of polygons

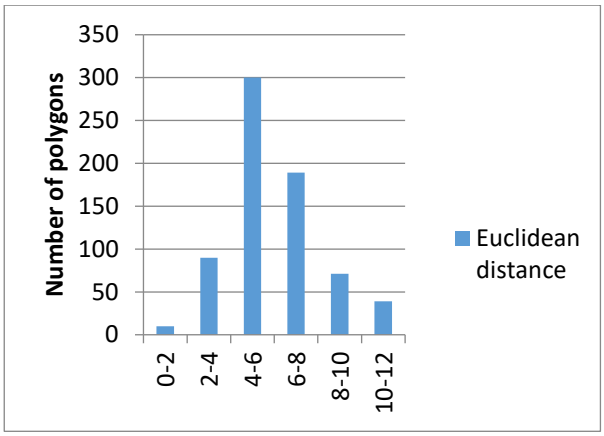


Figure 6.2. Euclidean distance variation with no of polygons

Figure 6.2 indicates that more number of building polygons have Euclidean distance variation of 4-8m between manual and automatic centroid polygons.This is mainly due to nDSM of buildings generated using automatic matching techniques which depends on the resolution of the stereo image in photogrammetry and point cloud spacing in LiDAR.

The Euclidean distance of manual and automatic footprints with area of buildings are shown as scatterplot in Figure 6.3 below.

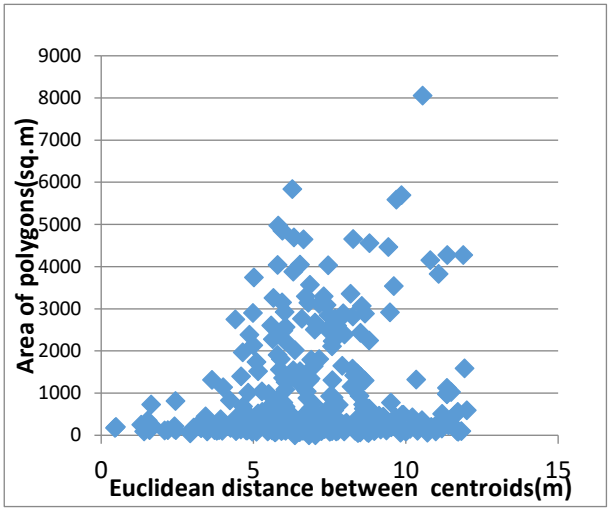


Figure 6.3. Euclidean distance variation with Area of polygons

Centroid Euclidean distance variation doesn't depend on absolute area of buildings and the variation is the same for buildings of area less the 1000sq.m and more than 1000sq.m to 5000sq.m within the test area, and it depends only on shape of building.

The manual and automatic footprints of LiDAR pointcloud are shown in Figure 6.4, Figure 6.5

Figure 6.4. Automatic building footprint from LiDAR

Figure 6.5. Manual building footprint from LiDAR

The centroids from automatic and manual footprints are shown in Figure 6.6

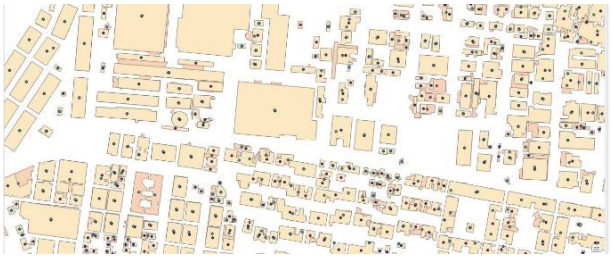


Figure 6.6. Centroid of polygons from manual building footprint and automatic building footprint

The Figure 6.7 show the histogram of Euclidean distance between centroids of manual and automatic building footprint obtained from LiDAR pointcloud.

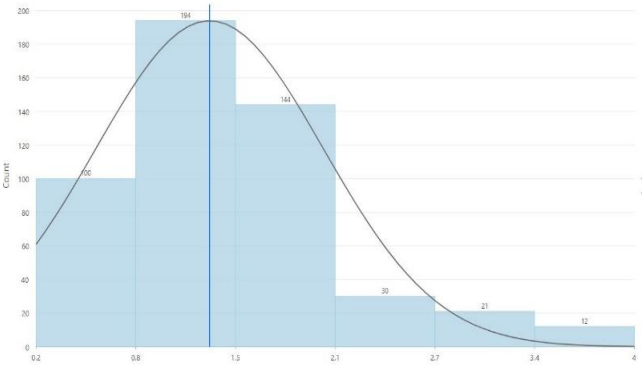


Figure 6.7. Histogram of Euclidean distance and building count

There were omissions in automatic building and the parameters like shrink factor has to be varied based on density of buildings for different pointclouds. In the LiDAR point cloud, since multireturn technology was available, roofs under trees had building corners. In contrast, it was not there in the photogrammetric point cloud, as it was only a surface model.

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-X-5-W2-2025-429-2025 | © Author(s) 2025. CC BY 4.0 License.

6.2 Height Comparison

The building heights derived from nDSM is validated against the building footprints and heights captured stereoscopically on a Digital Photogrammetric workstation for Geoeye data. Nearly 1300 building polygons were captured stereoscopically and taken as reference data for the validation of the building heights.

Height Difference	±1m	±2m	±3m
%Building Poygons	**54%**	**77%**	**86%**

Table 2. Statistics showing height difference of building polygons

From the statistics shown in Table 2 it is evident that 54% of the polygons have a height difference well within ±1m. 77% of the polygons have height difference within ±2m and 86% of the polygons with a height difference of ±3m. The remaining 14% of the polygons could be categorized under outliers since the building footprints differ due to the data capture, generalization, omission& commissions errors etc., in the data. The height variations can be also be due to the automatic DSM generation from high resolution satellite data through stereo matching technique.

CONCLUSION AND RECOMMENDATIONS

A detailed methodology for the extraction of building outline from high resolution data acquired in stereoscopic mode and LiDAR point cloud are explained in this paper. The method used has essentially following major steps:

1. Digital Surface Model generation using robust image matching techniques.
2. Norrmalised (nDSM) Digital Surface Model generation from object based image analysis using rulesets.
3. Segregation of trees and buildings using NDVI to generate building point clouds from satellite and filtering for LIDAR point cloud.
4. Roof Face Plane Detection, Boundary Detection, Outline Smoothing and Footprint Generation

The building footprint obtained is the major input for 3D city modeling.An accuracy assessment was done based on the comparison manual building footprints (by digitization of building footprint) and automatically generated by implementing the proposed methodology. The area and heights of the buildings obtained from manual footprint was compared with area of the automatically generated footprint.

This project utilized high resolution (50 cm) satellite imagery of GeoEye and LiDAR pointcloud. Approximately 90% footprints were within centroidal distance of 2m in LiDAR pointcloud and satellite images its within 4-8m for most footprints. This implies that dense point cloud (closer spacing) gives near real shape of building. To get high accuracy very dense point cloud is required.

Future study can analyse quantitatively the accuracy for different point cloud spacing.

With avalability of high resolution satellite data from future high resolution stereo missions, data of 2-3 cm GSD from UAV sensors, the methodology can be used to extract buildings footprints automatically. This would reduce the Turn AroundTime (TAT) for building feature extraction which will be very useful for disaster applications.

REFERENCES

Alobeid, A., Jacobsen, K., & Heipke, C. (n.d.). Building height estimation in urban areas from very high resolution satellite stereo images.

Alrajhi, M., Alam, Z., Khan, M. A., & Alobeid, A. (2016). Influence of GSD for 3D city modeling and visualization from aerial imagery. *ISPRS Archives*, *XLI-B3*, XXIII ISPRS Congress, Prague, 12–19 July 2016.

Awrangjeb, M. (2016). Using point cloud data to identify, trace, and regularize the outlines of buildings. *International Journal of Remote Sensing*, *37*(3), 551–579. https://doi.org/10.1080/01431161.2015.1131868

Awrangjeb, M., & Lu, G. (2014). Automatic building footprint extraction and regularisation from LiDAR point cloud data. In *Digital Image Computing: Techniques and Applications (DICTA)*. doi.org/10.1109/DICTA.2014.7008096

Axelsson, P. (1999). Processing of laser scanner data – Algorithms and applications. *ISPRS Journal of Photogrammetry and Remote Sensing*, *54*(2–3), 138–147.

Axelsson, P. (2000). DEM generation from laser scanner data using adaptive TIN models. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *33*(B4/1), 110–117.

Bennett, J. (n.d.). *OpenStreetMap: Be your own cartographer*. Packt Publishing.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2001). 33.3: Finding the convex hull. In *Introduction to algorithms* (2nd ed., pp. 955–956). MIT Press and McGraw-Hill.

Durieuxa, L., Lagabrielle, E., & Nelson, A. (2008). Method for monitoring building construction in urban sprawl areas using object-based analysis of Spot 5 images and existing GIS data. *ISPRS Journal of Photogrammetry and Remote Sensing*, *63*, 399–408.

Durieuxa, L., & Hirschmuller, H. (n.d.). Semi-global matching – Motivation, development and applications.

Erdem, M., & Anbaroglu, B. (2023). Reproducible extraction of building footprints from airborne LiDAR data: A demo paper. doi.org/10.1145/3589132.3625574

Gehrke, S., Morin, K., Downey, M., Boehrer, N., & Fuchs, T. (n.d.). Semi-global matching: An alternative to LiDAR for DSM generation?

Hu, J., You, S., & Neumann, U. (2003). Approaches to large-scale urban modeling. *IEEE Computer Graphics and Applications, 23*(6), 62–69.

Kong, G., Fan, H., & Lobaccaro, G. (2022). Automatic building outline extraction from ALS point cloud data using generative adversarial network. *Geocarto International*. doi.org/10.1080/10106049.2022.2102246

Kraus, T., Lehner, M., & Reinartz, P. (n.d.). Generation of coarse 3D models of urban areas from high resolution stereo satellite images.

Li, X., Qiu, F., Shi, F., & Tang, Y. (2022). A recursive hull and signal-based building footprint generation from airborne LiDAR data. *Remote Sensing, 14*(22), Article 5892. doi.org/10.3390/rs14225892

Nex, F., & Remondino, F. (2012). Automatic roof outlines reconstruction from photogrammetric DSM. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, I-3*, 25 August – 01 September 2012, Melbourne, Australia.

Nurunnabi, A., Teferle, N., Balado, J., Chen, M., Poux, F., & Sun, C. (2022). Robust techniques for building footprint extraction in aerial laser scanning 3D point clouds. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. doi.org/10.5194/isprs-archives-xlviii-3-w2-2022-43-2022

Proulx-Bourque, J.-S., McGrath, H., Bergeron, D., & Fortin, C. (2021). Extraction of building footprints from LiDAR: An assessment of classification and point density requirements. doi.org/10.1007/978-3-030-59109-0_11

Ribarsky, W., Wasilewski, T., & Faust, N. (2002). From urban terrain models to visible cities. *IEEE Computer Graphics and Applications, 22*.

Rottmann, P., Haunert, J.-H., & Dehbi, Y. (2022). Automatic building footprint extraction from 3D laserscans. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, X-4/W2*, 233–2022. doi.org/10.5194/isprs-annals-x-4-w2-2022-233-2022

Sithole, G., & Vosselman, G. (2004). Experimental comparison of filter algorithms for bare-earth extraction from airborne laser scanning point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing, 59*, 85–101.

Vostikolaei, F. S., & Jabari, S. (2024). Multimodal building footprint extraction from orthophoto and Lidar point clouds using deep learning framework. doi.org/10.1109/igarss53475.2024.10641225

Shen Wei. (2008). Building boundary extraction based on LIDAR point clouds data. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 37*(B3b), Beijing.

Wolf, P. R., & Dewitt, B. A. (2000). *Elements of photogrammetry: With application in GIS* (3rd ed.). McGraw-Hill Education (Asia).