A novel silhouette extraction method for binary images based on the Wall-Follower algorithm

Mario A. Aguirre-López
Autonomous University of Chiapas, Faculty of Sciences in Physics and Mathematics
Carretera Emiliano Zapata km 8.0
29050, Tuxtla Gutiérrez, Mexico
marioal1906@gmail.com

Omar Efraín Izaguirre-Prado
Autonomous University of Nuevo León, Faculty of Earth Sciences
Carretera a Cerro Prieto km 8.0
67700, Linares, Mexico
omar.izaguirrepr@uanl.edu.mx

Roberto Soto-Villalobos
Autonomous University of Nuevo León, Faculty of Earth Sciences
Carretera a Cerro Prieto km 8.0
67700, Linares, Mexico
roberto.sotovll@uanl.edu.mx

Filiberto Hueyotl-Zahuantitla*
Cátedra CONAHCyT-UNACH
Carretera Emiliano Zapata km 8.0
29050, Tuxtla Gutiérrez, Mexico
fhueyotl@conahcyt.mx
*Corresponding author

ABSTRACT

Silhouette extraction involves separating objects of interest from their background, which has several applications in image processing. Among the silhouette extraction techniques, contour tracing is commonly applied to images with a uniform background. This paper introduces a novel contribution to contour tracing techniques, utilizing the Wall-Follower Algorithm (WFA) to extract silhouettes with uniform backgrounds, or binary images. The algorithm is based on the analogy of a follower sequentially walking aside the external boundary of a wall, without separating a hand from it; then, the follower walks tagging silhouette pixels along the way until returning to the initial position and direction. Experimentation on vehicle technical drawings, satellite views of bodies of water and photographs of plants shows its effectiveness in producing high-quality silhouettes while showing some advantages over existing techniques. They include quickness in obtaining a solution, efficiency and ability to handle complex contours, and the option to simplify the results by reducing the percentage of saved points that trace the perimeter, based on object characteristics. The robustness of the algorithm suggests it as a promising alternative with diverse applications in image analysis, computer-aided design, and 3D object reconstruction, by extruding silhouettes, the latter being the main motivation for this contribution.

Keywords

contour tracing, silhouette extraction, wall-follower algorithm, image analysis, pixel following, technical drawing, CAD

1 INTRODUCTION

When processing an image to identify and object of interest from it, edge detection is a basic operation that uses the idea of detecting abrupt changes in image intensity. Classic methods to perform this task use first or second derivatives of the image. Examples of the first kind are the Canny, Roberts, Prewitt and Sobel algorithms which compute the magnitude and direction of the intensity changes at every pixel; while, the Marr-Hildreth and Harlick algorithms are examples of those that use second derivative, these include additional steps to reduce noise before detecting edges in the image. A good review of these methods and pseudo codes can be found in [SC15, ALM17]. Several of the above mentioned algorithms are implemented in well know software, such as the free-software GIMP (GNU Image Manipulation Program), Python and Matlab.

A step beyond edge detection is the silhouette extraction or contour tracing of a pattern (also called border tracing or boundary tracing). It is defined as the separation of the object of interest in the image (foreground) from the rest of them (background) [AYA+20]. Silhouette based methods have become an important tool in Computer Aided Design (CAD), for example, in object classification [MgS+14], technical illustrating [GSG+99, KBB17, PS23], and body feature extraction and recognition [PPJ14, LW11, CHK+06, DLR+24, DTGÇ06, ST02].

Contour tracing is a technique applied to digital images (representation as an array of $m \times n$ pixels), where each pixel having a certain value according to their intensity color. Contour tracing algorithms usually uses bi-level (binary) images, images that consists on pixels that can have only one of exactly two values: 1 when it is part of the pattern or 0 when it is part of the background. In order to extract the silhouette of a pattern, the identification of the boundary pixels is not enough, it is required an ordered sequence of the boundary pixels (a set of connected pixels). Contour tracing is a major contributor to the efficiency of the feature extraction process of silhouettes, for example to implement Freeman's chain-coded Curves, [FD77, LW11]. The most common contour tracing algorithms are the Square Tracing [Pra01], Moore-Neighbor Tracing [SHB08], the Radial Sweep [RAB12], Theo Pavlids' algorithm [Pav82], Snakes algorithm [KWT88], Amoeba algorithm [IV00], Topological-hierarchical algorithms [KC14]; among others so called Fast Contour-Tracing algorithms [BD18, SCS$^+$16].

A more complex method is introduced by [BH98], who proposed a curve tracing method fitting into the first category of both classifications, [ZQ04] and [AYA$^+$20]. It consists of three steps: i) locate a point of the surface by means of a search along a line of sight; ii) to find the silhouette by using a gradient method to walk along it; iii) to trace the silhouette by numerically integrating the corresponding equations and closing the form of the shape. As punctuated by the authors, this method highlights by the simplicity in construction and quickness in performance, however, it also has some disadvantages, mainly when dealing with pointed or sharped shapes.

In this context, our work provides an alternative method based on the Wall-Follower Algorithm (WFA) to extract silhouettes with known or uniform background. Details of the foreground that our technique can deal with are fully described in Section 2, together with its limitations.

The WFA was originally designed as a maze solver method. Its name arises because it reflects a person who tries to get out of the maze following the strategy of not removing one hand from a continuous wall, so that person follows a "hand on wall rule". Then, the method is a very effective but slow solver for perfect mazes, i.e. for those that have only two external continuous walls defining the entry and exit of the maze [NZ20]. Through years, the efficiency of the WFA has maintained in exploration which, in turn, has lead to the existence of some improved variants to solve specific types of mazes, such as: [DR06], who implemented a repulsive potential field with constrained active contours to set up large autonomous robots for navigation into diverse tunnel geometries, solving the problem of steering in narrow curved spaces; then, [DSG10] optimized the WFA for flood-mazes by sensing lateral walls to detect channels and avoid unnecessary fillers, so the greater the number of channels, the shorter the solving time; and more recently, [AAA$^+$23] developed an algorithm based on a modified Right-Left-Front hand rule to minimize the distance path while addressing the characteristic infinite loop-back problem of WFAs, which occurs when searching the entire area is required.

Clearly, the characterization of the silhouette is a mandatory task in several areas. However, to our understanding, no version of the WFA has been adapted for silhouette extraction, which is the primary focus of this study. As suggested above, it is well known that reconstructing 3D shapes from 2D images is a basic research area in computer vision, see for example [BCL01, BJL03] and references therein. The motivation for developing our algorithm is then to create a tool to generate 2D silhouette data which could then be used to extrude 3D shapes. In this sense, we oriented this work to vehicle shapes, in which technical draws in gray scale are commonly used. The structure of the paper is as follows: Section 2 describes the details of our method and its implementation in the statistical software R; Section 3 shows some study cases and discuss them; Section 4 highlights the concluding remarks of our work.

2 METHODOLOGY

2.1 The Wall-Follower Algorithm

In general terms, the WFA we programmed consists of a main routine (Algorithm 1) that sequentially calls two functions with the objective of tracing the contour of the foreground, by moving the "wall follower" in a right-hand lining procedure from a starting point up to the wall follower returns to its initial position with its initial direction. The procedure requires a matrix R, in which the original image is stored; a matrix S, with the same dimensions of R which will serve to store the silhouette; and a 2D-vector T to save pixel coordinates of the silhouette, arranging them as they were stored. In the algorithm, xy_{Start} represent the initial position at a side of the foreground and d_{Start} refers to the direction, which is parallel to the silhouette.

Algorithm 1 Pseudo-code for the WFA

procedure CONTOUR_LINER(R,S,T,xy_{Start},d_{Start})
  $xy = xy_{Start}$
  $d = d_{Start}$
  $finish = 0$
  $i = 0$
  while $finish = 0$ **do**
    Call `MOVE_TAG`
    Call `TAG_TURN`
    Update $finish$
return S,T

In the Algorithm 1, the function `MOVE_TAG` is in charge of moving forward one step or turning left when a wall is in front; while the function `TAG_TURN` aims to tagging one cell at right or turning right when that cell is not part of the wall. Figs. 1 and 2 help to schematize the cases faced by the above functions.

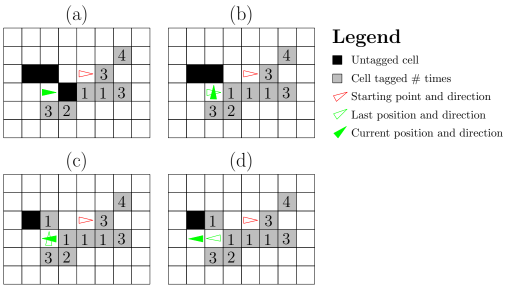

Figure 1: Visualization of the performance of the function `MOVE_TAG`. Cells in gray are tagged the number of times indicated inside them. Cells in black are part of the silhouette but they are still not tagged. (a) The follower in green is positioned after tagging by third time a cell by means of function `TAG_TURN`. (b) The cell in front is tagged by first time while the follower turns left. (c) The subsequent cell in front is tagged by first time while the follower turns left. (d) The follower can move forward.

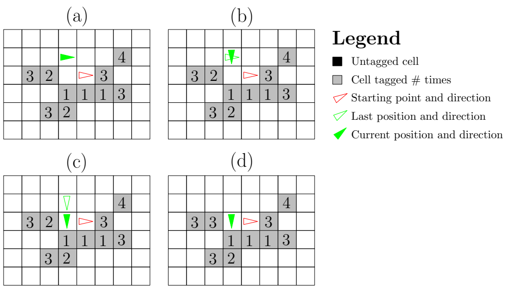

Figure 2: Visualization of the performance of the function `TAG_TURN`. Cells in gray are tagged the number of times indicated inside them. Cells in black are part of the silhouette but they are still not tagged. (a) The follower in green is positioned after tagging by second time a cell and moving forward by means of function `MOVE_TAG`. (b) The follower turns right since the cell in right is not part of the silhouette. (c) As a part of the same conditional, the follower moves forward. (d) The cell in right is sensed and tagged since it is part of the silhouette.

In detail, the first function to be called is shown in Algorithm 2, which is in charge to move the follower one cell in straight direction (L8), but considering that there could be scenarios where a wall is located in front of the follower, as schematized in Fig. 1. For that reason, the follower is able to turn left (or rotate in relative left direction) up to avoid walls in front (L3). The reader should be noticed that a wall implies that the associated cell is, indeed, part of the silhouette due to the assumption that we are drawing the outside contour. Thus, L5 tags that cell as a part of the silhouette by calling an auxiliary function shown in Algorithm 3. The reader could also note in Fig. 1 that up to two tags of different cells and one movement forward could be performed with calling this first function only once.

The auxiliary functions in Algorithm 3 (L2 and L4) tags and identifies if the follower has returned to the initial directional position (L5), modifying the index

---

Algorithm 2 Pseudo-code for the first function

1: **function** MOVE_TAG($R, S, i, T, xy, d, xy_{Start}, d_{Start}$)
2: Update the cell in front xyd
3: **while** $R[xyd]$ is part of the silhouette **do** ▷ If not possible to go straight
4: $xy^* = xyd$; $d^* = d_{Left}$
5: Call `TAG_SIMPLE` ▷ Tag that cell
6: $d = d_{Left}$ ▷ Turn left
7: Update the new cell in front xyd
8: $xy = xyd$ ▷ Move on one cell
return $S, T, xy, d, i, finish$

$finish$ to 1 (L6) in such a case, in order to break the main loop in Algorithm 5.

Algorithm 3 Pseudo-code for the auxiliary function

1: **function** TAG_SIMPLE($S, i, T, xy^*, d^*, xy_{Start}, d_{Start}$) ▷ tags the xy^*-cell
2: $S[xy^*] = S[xy^*] + 1$ ▷ by adding 1 to that cell,
3: $i = i + 1$
4: $T[i] = xy^*$ ▷ by adding the 2D-coordinates
5: **if** $xy^* = xy_{Start}$ & $d^* = d_{Start}$ **then**
6: $finish = 1$
return $S, T, i, finish$

Once the first function in Algorithm 1 has moved the follower to a new cell, the second function (Algorithm 4) looks for tagging the cell located at right-hand of the current xy-cell, considering the updated d-direction. This is only applied when the cell is part of the silhouette; in such a case a tag is produced (L6-L9) and the function ends. However, if the right-hand cell can not be tagged because the follower completed a past tag and forward movement without changing direction, i.e., it was at an external border of the silhouette such as shown in Fig. 2; then the follower needs to turn (or rotate to) right, while advancing one step without breaking the conditional loop (L10-L12). Therefore, this process is repeated until a cell is tagged (L3).

A special remark on this construction is that, since tagging is performed by adding one unit to the considered cell, a cell could be tagged more than once when the follower is at an external border, so that the value of each cell corresponds to the number of faces belonging to the outside contour of the foreground, up to four in 8-connected patterns. Indeed, in every step, the follower sens the cell at right of it without coming into the foreground then, it is feasible to trace the contour of any type of 8-connected patterns (foregrounds with an external silhouette whose cell components either share an edge or a vertex), including 4-connected (those that only share edges). On the other hand, the user has two ways of exporting the extracted silhouette. A matrix representation (S) that identifies the number outside

Algorithm 4 Pseudo-code for the second function

function TAG_TURN($R, S, i, T, xy, d, xy_{Start}, d_{Start}$)
  $t = 0$ ▷ auxiliary variable
  while $t = 0$ **do** ▷ If cell in right direction is not tagged
    Update d_{Right}
    Sens that cell xyd_{Right}
    if $R[xyd_{Right}]$ is part of the silhouette **then**
      $xy^* = xyd_{Right}$; $d^* = d$
      Call `TAG_SIMPLE` ▷ Tag that cell
      $t = 1$ ▷ And break the loop
    else
      $d = d_{Right}$ ▷ Turn right to look for a cell that is part of the silhouette
      $xy = xyd_{Right}$ ▷ And go straight
return $S, T, xy, d, i, finish$

faces of the silhouette, and a 2D-vector (T) which is able to delimit the directional contour.

This construction, although with some limitations, has good performance and interesting features. Next, we briefly list limitations and characteristics. The last are described in depth in Section 3. The algorithm is limited to:

- Extract silhouettes consisting of simple closed curves, since a single crossover or overlap produces an early stop due to the L5 condition of the Algorithm 3;
- Cases with a simple background, or one that could be removed by an external filter.
- Silhouettes whose set of points are fully contained in the image, i.e., can not be part of the limits because the follower walks next to the outer face of the silhouette.

In turn, the characteristics are:

- Directional contour, the cells of the extracted silhouette could be saved/indexed in an orderly manner with respect to adjacent cells.
- Silhouette identifying borders of the object with a maximum of three faces to be tagged, the algorithm identifies how many faces of a cell belong to the outside contour of the object.

2.2 Implementation

The complete process proposed for the extraction of a silhouette consists of five parts and is structured as in Algorithm 5. First, an image needs to be read and saved in RGB or gray scales with decimal color code; such a matrix is referred as R. Then, appropriate filters should be applied to R in order to obtain an image fulfilling the restrictions mentioned in Section 2.1. The third step is to initialize the matrix S and the vector T, where the silhouette points and the directional contour will be saved in, respectively. Thus, the fourth step is to locate the starting point xy_{Start} and direction d_{Start} that would initialize the WFA; this could be done by any search along a line of sight until detecting the outside pixel of the silhouette since the algorithm performance is independent from its starting point. Finally the calling of WFA is carried out in the fifth step.

Algorithm 5 Main pseudo-code (integrator algorithm)

Read image and associate it to matrix R
Apply filters to R
Create a matrix S for saving the silhouette, and save orderly the coordinates in a 2D-vector T
Initialize xy_{Start}, d_{Start} for the WFA by a search along a line of sight
Call Algorithm 1

3 RESULTS

We tested our method with some images fulfilling the restrictions mentioned in Section 2.1. They consist of different views of technical draws of vehicles, which were freely obtained from Free Cad Blocks [Fre24], see Figs. 3-5. A pre-processing was applied to the images: imperfections such as those in the right wheel of the truck in Fig. 4 (b) and (c), and in the (disjointed) mirrors of the truck with platform in Fig. 5 (a).

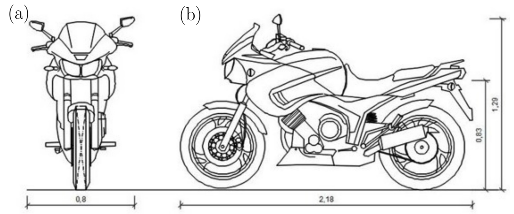

Figure 3: Front (a) and lateral (b) views of a motorcycle draw [Fre24].

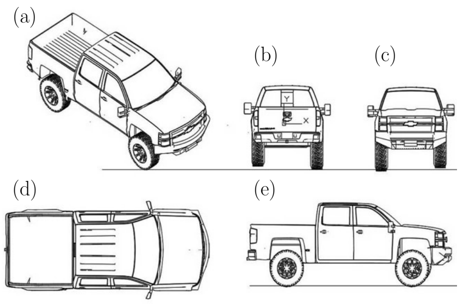

Figure 4: Perspective (a), rear (b), front (c), top (d), and lateral (e) views of a truck draw [Fre24].

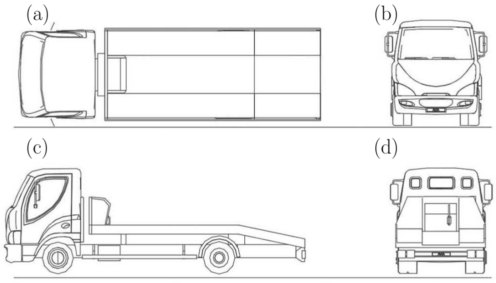

Figure 5: Top (a), front (b), lateral (c), and rear (d) views of a truck with platform draw [Fre24].

Following the process mentioned in Algorithm 5, images were read considering a gray scale from 0 (white) to 1 (black). A simple filter consisting of mapping all values < 0.8 to 0-value and the rest of them to 1 was applied. The search for xy_{Start} and d_{Start} was initialized by testing cells of the image in downward direction from the top-center; when an analyzed cell was found to belong to the silhouette, then the search stopped, as schematized in Fig. 6. Finally, the Algorithm 1 was called.

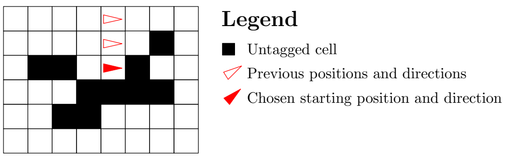

Figure 6: Visualization scheme of the performance of searching for a starting point. The search starts from the columns closest to the top-center of the image.

Figs. 7-9 show the resulting silhouettes of some representative views: Fig. 7 (a) and (b) for cases (a) and (b) in Fig. 3; Fig. 8 (a) and (b) for cases (a) and (e) in Fig. 4, respectively; Fig. 9 (a) and (b) for the cases (a) and (c) in Fig. 5, respectively. Two silhouettes are plotting per graphic, indicating the number of points defining them. Continuous gray lines refer to the full contour obtained by applying our technique. Red dots refer to a simplified contour considering only silhouette cells that are tagged two or more times with our technique.

The full silhouettes provide high quality solutions, capturing the directional contour of the body in such a manner that the target object can be easily recognized. On the other hand, the simplified silhouettes reproduce a contour free of straight lines, achieving to reduce considerably the number of points, mainly when flat surfaces are present. However, some details of the original images could be lost when large flat zones are combined with right angles, like those ones at the connection of the truck with the platform in Fig. 9 (a) and (b). The resulting silhouette would not be reliable reconstruction of the vehicle.

Table 1 summarizes the percentages of points saved when utilizing the simplified approximation for all the

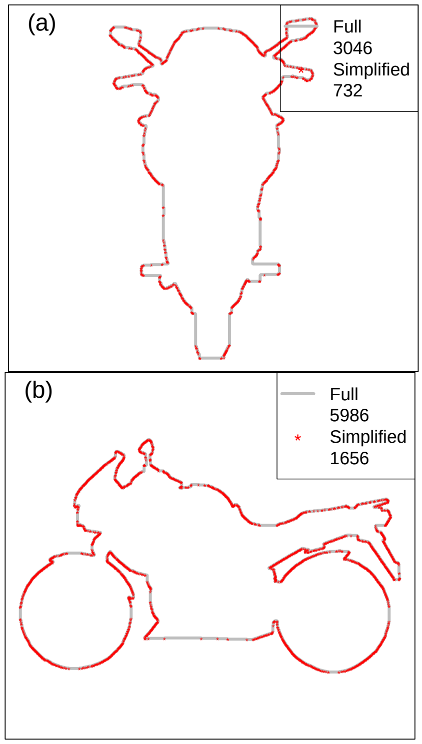

Figure 7: Full and simplified silhouette extraction of the motorcycle draws shown in Fig. 3. (a) Front view. (b) Lateral view.

analyzed cases. Since the structure of the truck cases is similar, full reconstructions of them are achieved with almost the same points, around 3000. The lateral view of the motorcycle requires about twice as many points. The behavior of the simplified reconstructions contrasts with this, depending mainly of the presence of (horizontal and vertical) flat zones, as suggested before. Such an example, the full reconstruction shown in Fig. 9 (a) is reduced in 95%, and the most of the points in the simplified reconstruction take place at the mirrors of the truck.

Thus, the reconstruction to consider for a specific application will depend on the flatness zones of the analyzed object and the type of edges at the borders of those zones, being useful a simplified reconstruction when there are cells exposing more than one face at the edges. In this context, external factors such as (original or added) noise at the edges could be helpful to avoid the coarseness problem. This, and other lines of research like the identification of shadows, the completeness of edge discontinuities and the extraction of a second internal silhouette are proposed as future extension of this work.

Now, beyond the high potential use of our algorithm for technical draws and meshing of vehicles, we applied the technique to other kind of images in order to discuss its feasibility in different scenarios. Fig. 10 shows two

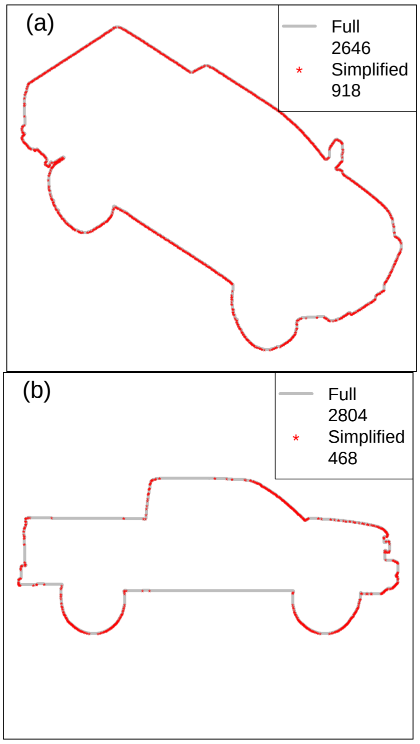

Figure 8: Full and simplified silhouette extraction of the truck draws shown in Fig. 4. (a) Perspective view. (b) Lateral view.

Object	View	Full (# points)	Simplified (%)
Moto	Front	3,046	24%
Moto	Lateral	5,986	28%
Truck1	Perspective	2,646	35%
Truck1	Lateral	2,804	17%
Truck2	Top	3,212	05%
Truck2	Lateral	2,886	12%

Table 1: Comparison of the number of points for each extracted silhouette. Simplified silhouettes are written as a percentage of their corresponding full versions. Truck1 corresponds to the single truck, while Truck2 to the truck with platform.

satellite photos of water bodies in Mexico. Their reconstructions are shown in Fig. 11. Even though the original images have not a uniform or known background, there is a high color contrast (for both images) between the soil and the water bodies, which allows a good performance of our algorithm, either through a full reconstruction or a simplified one (about 25% of the full one).

On the other hand, Fig. 12 shows two photos of plants with uniform background but many internal holes. As a consequence of this, only their external silhouettes are successfully extracted with details that increase the number of points (up to 28,000 for the full version of the palm case), but the internal details are lost, as occurred with all the contour tracing algorithms dealing

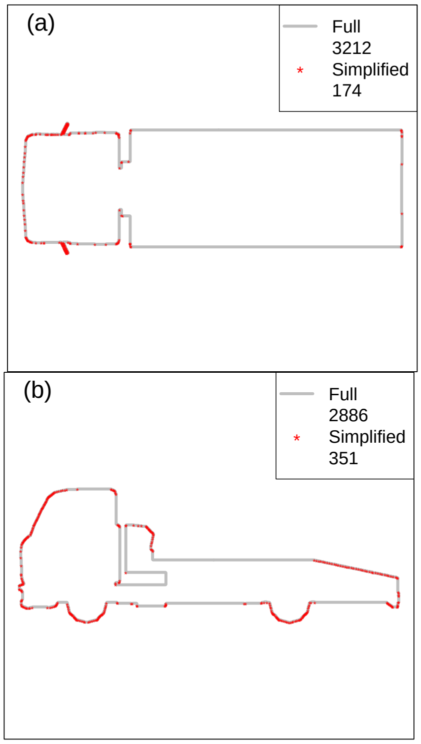

Figure 9: Full and simplified silhouette extraction of the truck draws shown in Fig. 5. (a) Top view. (b) Lateral view.

with such a kind of images. The weight ratio between full and simplified reconstructions is also 20-25%.

Figure 10: Satellite views of water bodies. (a) Dam Cerro Prieto in Linares, Nuevo León, Mexico. (b) Natural water body in China, Nuevo León, Mexico.

Beyond the characteristics of independence from the choice of the starting point and the directional contour, it is important to remark three advantages of our method over the reported techniques mentioned in Section 1:

- Quickness: taking into account that the Square Tracing algorithm is the one that requires the minimum of cell inspections among the existing techniques, while it is the most similar to ours, having as main difference that the WFA does not need to enter and leave the contour, since the follower always remain outside it.

- Complex contours: similarly to the Moore-Neighbor Tracing, Redial Sweep, Theo Pavlidi's,

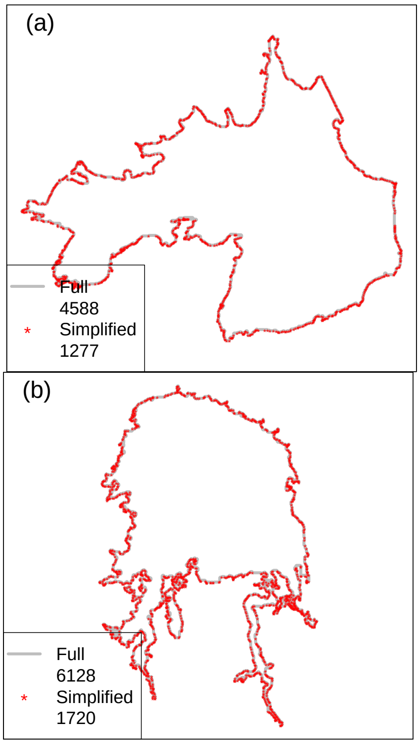

Figure 11: Full and simplified silhouette extraction of the two dams shown in Fig. 10.

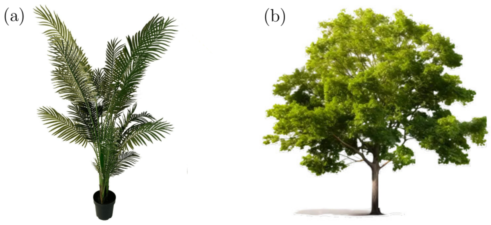

Figure 12: Photos of plants with uniform background. (a) A palm. (b) A tree.

and Fast Backward Contour Tracing algorithms, our implementation of the WFA is able to deal with 8-connected patterns that are not 4-connected, which is not possible for the Square Tracing algorithm.

- Simplified contour: our technique posses the duality of a) delimiting the complete contour, providing the 100% of the perimeter such as the Square Tracing algorithm; b) or well, it can simplify the contour, reducing the number of points that represent the perimeter by up to 83% for the analyzed cases, which possessed a complex geometry. This is a significant characteristic for saving computational storage when addressing high resolution images. In this sense, our algorithm can compete with traditional Freeman's chain codes (8- and 4-directional) [Fre61, FD77] and chain code compression schemes [ÅML15]. In turn, our results contrast with those of the experiments shown in [BD18] for the above mentioned algorithms, in which the Fast Backward Contour Tracing algorithm provided the most reduced contours, only up to 35%.

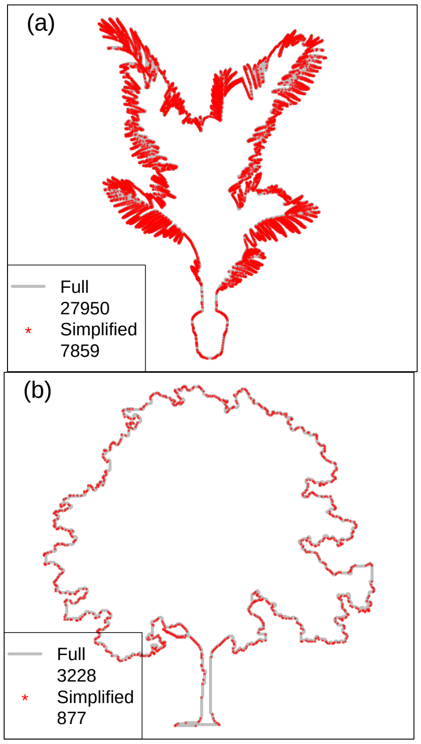

Figure 13: Full and simplified silhouette extraction of the plants shown in Fig. 12.

It is important to note that the simplified silhouette obtained as a vector of ordered points could be used as entry data for 3D automotive design in CAD software by extruding the silhouette data on each direction and then, intersecting the extruded objects, as shown in Fig. 14. This is a line of research that could be useful for novice designers and engineers looking for a simple and specialized method, emphasizing usability of the resulting designs from technical draws.

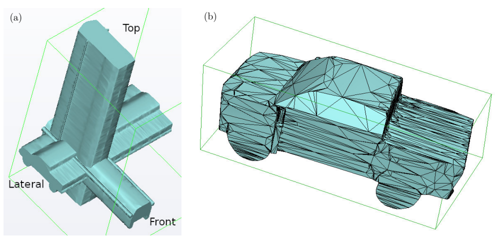

Figure 14: Application of ordered silhouettes in the 3D reconstruction of Truck1 in FreeCAD software, with cases the lateral, front, upper views shown in Fig. 4. (a) Extrusion of the silhouettes. (b) Final object.

4 CONCLUSIONS

We developed a technique to extract an external silhouette from a binary image, based on the Wall-Follower algorithm. The method is robust under considerable entry data conditions and presents some advantages over existing similar techniques, such as: independence of the starting point, quickness, reliability when dealing with complex and spiked contours, and the possibility of minimizing the number of points to trace the final contour. The property of the algorithm to store the complete contour and at the same time the simplified silhouette (which can be considered as a compressed form), as a matrix and as ordered vector elements, enhances its application in various areas that require the use of silhouettes.

5 FUNDING

This research was funded by Consejo Nacional de Humanidades, Ciencias y Tecnologías (CONAHCyT), grant number 839412.

6 ACKNOWLEDGMENTS

The third author thanks FCFM-UNACH and the support from CONAHCyT through the program "Investigadoras e investigadores por México", Cátedra 873.

7 REFERENCES

[AAA+23] Shatha Alamri, Hadeel Alamri, Wejdan Alshehri, Shuruq Alshehri, Ahad Alaklabi, and Tareq Alhmiedat. An autonomous maze-solving robotic system based on an enhanced wall-follower approach. Machines, 11(2), 2023.

[ALM17] Naurin Afrin, Wei Lai, and Nabeel Mohammed. Performance analysis of corner detection algorithms based on edge detectors. Computer Science Research Notes - CSRN, CSRN 2701:21–28, May 2017.

[AYA+20] Guido Ascenso, Moi Hoon Yap, Thomas Allen, Simon S. Choppin, and Carl Payton. A review of silhouette extraction algorithms for use within visual hull pipelines. Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization, 8(6):649–670, 2020.

[BCL01] Andrea Bottino, L. Cavallero, and Aldo Laurentini. Interactive reconstruction of 3-d objects from silhouettes. In Journal of WSCG, Vol.09, No. 1-3, pages 230–236, 01 2001.

[BD18] Yuriy Batko and Vitalii Dyminskyi. Fast contour tracing algorithm based on a backward contour tracing method. In Mykola Dyvak and Libor DostÃ¡lek, editors, Proceedings of the International Conference Advanced Computer Information Technologies, ACIT 2018, Ceske Budejovice, Czech Republic, June 1-3, 2018, volume 2300 of CEUR Workshop Proceedings, pages 219–222. CEUR-WS.org, 2018.

[BH98] David Bremer and John F. Hughes. Rapid approximate silhouette rendering of implicit surfaces. In Proceesings of Implicit Surfaces 98, 1998.

[BJL03] Andrea Bottino, Luc Jaulin, and Aldo Laurentini. Finding feasible parameter sets for shape from silhouettes with unknown position of the viepoints. In Proceedings of the The 11-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2003, WSCG 2003, Plzen, Czech Republic, Februay 7-13, 2003.

[CHK+06] Xi Chen, Zhihai He, J.M. Keller, D. Anderson, and M. Skubic. Adaptive silhouette extraction in dynamic environments using fuzzy logic. In 2006 IEEE International Conference on Fuzzy Systems, pages 236–243, 2006.

[DLR+24] Arindam Dutta, Rohit Lal, Dripta S. Raychaudhuri, Calvin-Khang Ta, and Amit K. Roy-Chowdhury. Poise: Pose guided human silhouette extraction under occlusions. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pages 6153–6163, January 2024.

[DR06] Elliot S. Duff and Jonathan M. Roberts. Wall Following with Constrained Active Contours, pages 51–60. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[DSG10] Hongshe Dang, Jinguo Song, and Qin Guo. An efficient algorithm for robot maze-solving. In 2010 Second International Conference on Intelligent Human-Machine Systems and Cybernetics, volume 2, pages 79–82, 2010.

[DTGÇ06] Yiğithan Dedeoğlu, B. Uğur Töreyin, Uğur Güdükbay, and A. Enis Çetin. Silhouette-based method for object classification and human action recognition in video. In Thomas S. Huang, Nicu Sebe, Michael S. Lew, Vladimir Pavlović, Mathias Kölsch, Aphrodite Galata, and Branislav Kisačanin, editors, Computer Vision in Human-Computer Interaction, pages 64–77, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[FD77] Freeman and Davis. A corner-finding algorithm for chain-coded curves. IEEE Transactions on Computers, C-26(3):297–303, 1977.

[Fre61] Herbert Freeman. On the encoding of arbitrary geometric configurations. IRE Transactions on Electronic Computers, EC-10(2):260–268, 1961.

[Fre24] Free CAD Blocks. CAD library. https://freecadblock.com/, 2024. Online; accessed March 1, 2024.

[GSG+99] Bruce Gooch, Peter-Pike J Sloan, Amy Gooch, Peter Shirley, and Richard Riesenfeld. Interactive technical illustration. In Proceedings of the 1999 symposium on Interactive 3D graphics, pages 31–38, 1999.

[IV00] G. Iannizzotto and L. Vita. Fast and accurate edge-based segmentation with no contour smoothing in 2-d real images. IEEE Transactions on Image Processing, 9(7):1232–1237, 2000.

[KBB17] Hakki Can Karaimer, Ipek Baris, and Yalin Bastanlar. Detection and classification of vehicles from omnidirectional videos using multiple silhouettes. 20(3):893–905, 2017.

[KC14] Pisut Koomsap and Natthavika Chansri. Topological hierarchy-contour tracing algorithm for nests of interconnected contours. The International Journal of Advanced Manufacturing Technology, 70:1247–1266, 02 2014.

[KWT88] Michael Kass, Andrew P. Witkin, and Demetri Terzopoulos. Snakes: Active contour models. Int. J. Comput. Vis., 1(4):321–331, 1988.

[LW11] Yueh-Ling Lin and Mao-Jiun J. Wang. Automated body feature extraction from 2d images. Expert Systems with Applications, 38(3):2585–2591, 2011.

[MgS+14] Begüm Mutlu, Murat Haciömeroğlu, Guzel Mehmet Serdar, Mehmet Dikmen, and Hayri Sever. Silhouette extraction from street view images. International Journal of Advanced Robotic Systems, 11(7):114, 2014.

[NZ20] R. Niemczyk and Stanisław Zawiślak. Review of maze solving algorithms for 2d maze and their visualisation. In Stanisław Zawiślak and Jacek Rysiński, editors, Engineer of the XXI Century, pages 239–252, Cham, 2020. Springer International Publishing.

[Pav82] T. Pavlidis. ALGORITHMS FOR GRAPHICS AND IMAGE PROCESSING. Computer Science Press, 1982.

[PPJ14] R. Pulido, F. Paulano, and J.J. Jiménez. Reconstruction & interaction with 3d simplified bone models. In Proceedings of the 22nd International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision 2014, 2014.

[Pra01] W.K. Pratt. Digital Image Processing: PIKS Inside. Wiley, 2001.

[PS23] A. Padinjarathala and R. Sadleir. Using the Adaptive HistoPyramid to Enhance Performance of Surface Extraction in 3D Medical Image Visualisation. Computer Science Research Notes, 3301:331–339, 2023.

[RAB12] Pratiksha R. Reddy, Vadipina Amarnadh, and Mekala Bhaskar. Evaluation of stopping criterion in contour tracing algorithms. 2012.

[SC15] Haldo Spontón and Juan Cardelino. A Review of Classic Edge Detectors. Image Processing On Line, 5:90–123, 2015. https://doi.org/10.5201/ipol.2015.35.

[SCS+16] Jonghoon Seo, Seungho Chae, Jinwook Shim, Dongchul Kim, Cheolho Cheong, and Tack-Don Han. Fast contour-tracing algorithm based on a pixel-following method for image sensors. Sensors, 16(3), 2016.

[SHB08] Milan Sonka, Vaclav Hlavac, and Roger Boyle. Image processing, analysis and and machine vision (3. ed.). 01 2008.

[ST02] Cristian Sminchisescu and Alexandru Telea. Human pose estimation from silhouettes - a consistent approach using distance level sets. In Journal of WSCG, Vol.11, No. 1-3, pages 413–420, 02 2002.

[ZQ04] Gang Zeng and Long Quan. Silhouette extraction from multiple images of an unknown background. In Proceedings of the Asian Conference of Computer Vision, Citeseer. Citeseer, 2004.

[ÅML15] Borut Å½alik, Domen Mongus, and Niko Lukač. A universal chain code compression method. Journal of Visual Communication and Image Representation, 29:8–15, 2015.