UNIVERSIDAD AUTÓNOMA
DE AGUASCALIENTES

CENTRO DE CIENCIAS BÁSICAS

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN.

**TESIS**

# TÉCNICAS DE INTELIGENCIA ARTIFICIAL Y MATEMÁTICAS APLICADAS AL RECONOCIMIENTO DE OBJETOS EN 2D Y 3D.

PRESENTA

Osvaldo Arturo Tapia Dueñas

PARA OPTAR POR EL GRADO DE DOCTOR EN CIENCIAS APLICADAS Y TECNOLOGÍA

TUTORES

Dr. Hermilo Sánchez Cruz

Dr. Hiram Habid López Valdez

Asesora

Dra. Ivonne Bazán Trujillo

Mayo 2023, Aguascalientes, Ags.

CARTA DE VOTO APROBATORIO
INDIVIDUAL

**M. en C. Jorge Martín Alférez Chávez**
DECANO (A) DEL CENTRO **DE CIENCIAS BÁSICAS**

PRESENTE

Por medio del presente como ***TUTOR*** designado del estudiante ***OSVALDO ARTURO TAPIA DUEÑAS*** con ID *152131* quien realizó *la tesis* titulado: ***TÉCNICAS DE INTELIGENCIA ARTIFICIAL Y MATEMÁTICAS APLICADAS AL RECONOCIMIENTO DE OBJETOS EN 2D Y 3D***, un trabajo propio, innovador, relevante e inédito y con fundamento en el Artículo 175, Apartado II del Reglamento General de Docencia doy mi consentimiento de que la versión final del documento ha sido revisada y las correcciones se han incorporado apropiadamente, por lo que me permito emitir el **VOTO APROBATORIO**, para que *el* pueda proceder a imprimir*la* así como continuar con el procedimiento administrativo para la obtención del grado.

Pongo lo anterior a su digna consideración y sin otro particular por el momento, me permito enviarle un cordial saludo.

**ATENTAMENTE**
**"Se Lumen Proferre"**
**Aguascalientes, Ags., a 17 día de Julio de *2023*.**

***DR. HERMILO SÁNCHEZ CRUZ***
**Cotutor de *tesis***

c.c.p.- Interesado

CARTA DE VOTO APROBATORIO
INDIVIDUAL

**M. en C. Jorge Martin Alférez Chávez**
DECANO (A) DEL CENTRO **DE CIENCIAS BÁSICAS**

PRESENTE

Por medio del presente como ***TUTOR*** designado del estudiante ***OSVALDO ARTURO TAPIA DUEÑAS*** con ID *152131* quien realizó *la tesis* titulado: ***TÉCNICAS DE INTELIGENCIA ARTIFICIAL Y MATEMÁTICAS APLICADAS AL RECONOCIMIENTO DE OBJETOS EN 2D Y 3D***, un trabajo propio, innovador, relevante e inédito y con fundamento en el Articulo 175, Apartado II del Reglamento General de Docencia doy mi consentimiento de que la versión final del documento ha sido revisada y las correcciones se han incorporado apropiadamente, por lo que me permito emitir el **VOTO APROBATORIO**, para que *el* pueda proceder a imprimir*la* así como continuar con el procedimiento administrativo para la obtención del grado.

Pongo lo anterior a su digna consideración y sin otro particular por el momento, me permito enviarle un cordial saludo.

**ATENTAMENTE**
**"Se Lumen Proferre"**
**Aguascalientes, Ags., a 17 día de Julio de *2023*.**

***DR. HIRAM HABID LÓPEZ VALDEZ***
**Cotutor de *tesis***

c.c.p.- Interesado
c.c.p.- Secretaría Técnica del Programa de Posgrado

CARTA DE VOTO APROBATORIO
INDIVIDUAL

**M. en C. Jorge Martín Alférez Chávez**
DECANO (A) DEL CENTRO **DE CIENCIAS BÁSICAS**

PRESENTE

Por medio del presente como ***ASESOR*** designado del estudiante ***OSVALDO ARTURO TAPIA DUEÑAS*** con ID *152131* quien realizó *la tesis* titulado: ***TÉCNICAS DE INTELIGENCIA ARTIFICIAL Y MATEMÁTICAS APLICADAS AL RECONOCIMIENTO DE OBJETOS EN 2D Y 3D***, un trabajo propio, innovador, relevante e inédito y con fundamento en el Artículo 175, Apartado II del Reglamento General de Docencia doy mi consentimiento de que la versión final del documento ha sido revisada y las correcciones se han incorporado apropiadamente, por lo que me permito emitir el **VOTO APROBATORIO**, para que *el* pueda proceder a imprimir*la* así como continuar con el procedimiento administrativo para la obtención del grado.

Pongo lo anterior a su digna consideración y sin otro particular por el momento, me permito enviarle un cordial saludo.

**ATENTAMENTE**
**"Se Lumen Proferre"**
**Aguascalientes, Ags., a 17 día de Julio de *2023*.**

***DRA. IVONNE BAZÁN TRUJILLO***
**Asesor de *tesis***

c.c.p.- Interesado

UNIVERSIDAD AUTÓNOMA DE AGUASCALIENTES

# DICTAMEN DE LIBERACIÓN ACADÉMICA PARA INICIAR LOS TRÁMITES DEL EXAMEN DE GRADO

POSGRADOS uaa

Fecha de dictaminación dd/mm/aaaa: 18/07/2023

**NOMBRE:** Osvaldo Arturo Tapia Dueñas **ID** 152131

**PROGRAMA:** Doctorado en Ciencias Aplicadas y Tecnología **LGAC (del posgrado):** Inteligencia Artificial Y Modelación Estadística

**TIPO DE TRABAJO:** ( X ) Tesis ( ) Trabajo Práctico

**TITULO:** Técnicas de Inteligencia Artificial y Matemáticas Aplicadas al Reconocimiento de Objetos en 2D Y 3D

**IMPACTO SOCIAL (señalar el impacto logrado):** La tesis contribuye a la simplificación y compresión de imágenes y objetos en 2D y 3D, logrando una disminución del tiempo de transmisión digital de dicha información. Adicionalmente, presenta una aplicación en el área de imágenes médicas de la metodlogía propuesta. Las metodologías y aplicaciones desarrolladas durante este trabajo pueden ser explotados en diversos campos de la

**INDICAR SI NO N.A. (NO APLICA) SEGÚN CORRESPONDA:**

| | |
|---|---|
| | ***Elementos para la revisión académica del trabajo de tesis o trabajo práctico:*** |
| *SI* | El trabajo es congruente con las LGAC del programa de posgrado |
| *SI* | La problemática fue abordada desde un enfoque multidisciplinario |
| *SI* | Existe coherencia, continuidad y orden lógico del tema central con cada apartado |
| *SI* | Los resultados del trabajo dan respuesta a las preguntas de investigación o a la problemática que aborda |
| *SI* | Los resultados presentados en el trabajo son de gran relevancia científica, tecnologíca o profesional según el área |
| *SI* | El trabajo demuestra más de una aportación original al conocimiento de su área |
| *SI* | Las aportaciones responden a los problemas prioritarios del país |
| *NO* | Generó transferecia del conocimiento o tecnológica |
| *SI* | Cumple con la ética para la investigación (reporte de la herramienta antiplagio) |
| | ***El egresado cumple con lo siguiente:*** |
| *SI* | Cumple con lo señalado por el Reglamento General de Docencia |
| *SI* | Cumple con los requisitos señalados en el plan de estudios (créditos curriculares, optativos, actividades complementarias,estancia, predoctoral, etc) |
| *SI* | Cuenta con los votos aprobatorios del comité tutoral, en caso de los posgrados profesionales si tiene solo tutorpodrá liberar solo el tutor |
| *N.A.* | Cuenta con la carta de satisfacción del Usuario |
| *SI* | Coincide con el título y objetivo registrado |
| *SI* | Tiene congruencia con cuerpos académicos |
| *SI* | Tiene el CVU del Conacyt actualizado |
| *SI* | Tiene el artículo aceptado o publicado y cumple con los requisitos institucionales (en caso que proceda) |
| | ***En caso de Tesis por artículos científicos publicados*** |
| *SI* | Aceptación o Publicación de los articulos según el nivel del programa |
| *SI* | El estudiante es el primer autor |
| *SI* | El autor de correspondencia es el Tutor del Núcleo Académico Básico |
| *SI* | En los artículos se ven reflejados los objetivos de la tesis, ya que son producto de este trabajo de investigación. |
| *SI* | Los artículos integran los capítulos de la tesis y se presentan en el idioma en que fueron publicados |
| *SI* | La aceptación o publicación de los artículos en revistas indexadas de alto impacto |

Con base a estos criterios, se autoriza se continúen con los trámites de titulación y programación del examen de grado:

**Sí** x

**No** ____

## FIRMAS

**Elaboró:**

* NOMBRE Y FIRMA DEL CONSEJERO SEGÚN LA LGAC DE ADSCRIPCION: Dr. Jaime Muñoz Arteaga

NOMBRE Y FIRMA DEL SECRETARIO TÉCNICO: Dr. Angel Eduardo Muñoz Zavala

*** En caso de conflicto de intereses, firmará un revisor miembro del NAB de la LGAC correspondiente distinto al tutor o miembro del comité tutoral, asignado por el Decano**

**Revisó:**

NOMBRE Y FIRMA DEL SECRETARIO DE INVESTIGACIÓN Y POSGRADO: Dr. Juan Jáuregui Rincón

**Autorizó:**

NOMBRE Y FIRMA DEL DECANO: Mtro. Jorge Martín Alférez Chávez

***Nota: procede el trámite para el Depto. de Apoyo al Posgrado***

En cumplimiento con el Art. 105C del Reglamento General de Docencia que a la letra señala entre las funciones del Consejo Académico: .... Cuidar la eficiencia terminal del programa de posgrado y el Art. 105F las funciones del Secretario Técnico, llevar el seguimiento de los alumnos.

Signal Processing: Image Communication 91 (2021) 116080

Contents lists available at ScienceDirect

Signal Processing: Image Communication

journal homepage: www.elsevier.com/locate/image

# Context-free grammars to detect straight segments and a novel polygonal approximation method

Osvaldo A. Tapia-Dueñas, Hermilo Sánchez-Cruz *

*Universidad Autónoma de Aguascalientes, Aguascalientes 20131, Ags., México*

ARTICLE INFO

*Keywords:*
Break points
Dominant points
Shortest path
Digital straight segments
Lost pixels
Tolerable error

ABSTRACT

A new method for the polygonal approximation is presented. The method is based on the search for break points through a context-free grammar, that accepts digital straight segments with loss of information, as well as the decrease in the error committed employing the comparison of a tolerable error. We present an application of our method to different sets of objects widely used, as well as a comparison of our results with the best results reported in the literature, proving that our method achieves better values of error criteria. Besides, a new way to find polygonal approximations, with context-free grammars to recognize digital straight segments without loss of pixels, it is also addressed.

## 1. Introduction

In the topic of polygonal approximation for the shape of binary objects, in recent decades a large number of articles have been published. These methods seek to obtain an efficient tradeoff between different error criteria [1–15].

In 1954, Attneave [16] studied how the human being compresses the information he receives from the world through visual perception. This was achieved by considering perception as an information management process, where much of the perceived information is redundant, since the points with high curvature, that make up the contour of a shape, are abundant in information content, therefore, they are fit and sufficient to represent contour shapes. An example of this is reflected in the experiment of his classic cat image, in which he identifies points with a high curvature in a snapshot and is joined by segments of continuous straight lines.

Following the Attneave experiment, various methods have been proposed for the presentation of contours of 2D objects by polygonal approximation, in which curve points (known as *dominant points*) and line segments are used to join them. This coding causes loss since it introduces some distortion errors. The error is permissible as long as the visual alteration is considered insignificant. Some algorithms look for minimize the number of dominant points (min-# problem) [17–19], Prasad [20] proposes a non-parametric framework to identify the dominant points using the Masood [9], RDP [21,22] and Carmona-Poyato et al. [10] methods. In [23] is proposed a non-optimal but unsupervised algorithm, called ICT-RDP, for the generation of polygonal approximations based on the convex hull. Others focus on decreasing the approximation error for a predefined number of vertices (problem min-$\varepsilon$) [11,13,24,25], Sarkar [3] uses the F8 chain code in addition to four propositions to determine the locations. Carmona-Poyato et al. [26] use a measure to identify what break points should be suppressed taking into account an allowable error. Some techniques seek to solve both the min-# and the min-$\varepsilon$ problems [4,5,15]. Masood [7] identifies the break points using the F8 chain code, in which recursively removes and rearranges the points. Kalaivani and Ray [15] propose a simple and easy technique for the detection of break points through the strategy of division (local deviation) and fusion (global deviation). As reported in [15], in some of these works, the number of dominant points is improved by sacrificing the integral square error (ISE), or vice versa. To determine polygonal approximations of a shape in [27], they develop an algorithm starting from coarse-level to more refined-level representation by varying the number of polygon sides. Ngo [28] proposes a fully discrete structure, based on the notion of blurred segments, to study the geometrical features on such curves and apply it in a process of polygonal approximation. Kai, Shurong and Zhongjian [29] develop a non-uniform rational B-spline interpolation scheme (NURBS). They fit a NURBS curve efficiently through an optimal polygonal approximation based on Mixed Integer Programming (MIP). Other authors use artificial intelligence, for example in [30], authors apply a coarse-grained parallel genetic algorithm (CGPGA) to solve polygonal approximation problem.

Some methods try to reduce the execution time [6,31,32]. Chetverikov and Szabó [33] propose a two-phase method, the first consists of using either inscribed triangles of the same size where vertices touch three points of the original contour or an angle where intersect two lines that touch two points of the original contour, the second phase is the elimination of dominant points using the Euclidean

---

* Corresponding author.
*E-mail address:* hsanchez@correo.uaa.mx (H. Sánchez-Cruz).

https://doi.org/10.1016/j.image.2020.116080
Received 25 May 2020; Received in revised form 26 September 2020; Accepted 22 November 2020

Multimedia Tools and Applications (2023) 82:9491–9515
https://doi.org/10.1007/s11042-022-13588-3

# 3D object simplification using chain code-based point clouds

**Osvaldo A. Tapia-Dueñas[1] · Hermilo Sánchez-Cruz[1] · Hiram H. López[2]**

Received: 1 December 2021 / Revised: 25 February 2022 / Accepted: 18 July 2022 /

## Abstract

This work aims to obtain a sequence of 3D point clouds associated with a 3D object that reduces the volume data and preserves the shape of the original object. The sequence contains point clouds that give different simplifications of the object, from a very fine-tuned representation to a simple and sparse one. Such a sequence is important because it satisfies different needs, from a faithful representation with a low reduction of points to a significant data reduction that only preserves the main properties of the object. We construct the sequence in the following way. We first obtain a voxelization of the original 3D object. Then, we organize the voxels by slices to get a single chain code that represents the original 3D object. The point clouds depend on the key points of the chain code. The Hausdorff distance and the average geometric error prove that the point clouds are invariant under rigid rotations and maintain the shape of the object. Our results indicate that the proposed method has an average efficiency of 60% regarding the state-of-the-art simplification methods.

**Keywords** Chain code · Voxelization · 3D shape · Data reduction · Key points

## 1 Introduction

During the last years, numerous works in the literature have studied the representation of a three-dimensional (3D) object. One of the main reasons is the growing demand for applications related to these objects, such as digitization, storage, and recognition. We focus on the representations given in terms of point clouds or voxels.

---

✉ Hermilo Sánchez-Cruz
hermilo.sanchez@edu.uaa.mx

Osvaldo A. Tapia-Dueñas
osvaldo.tapia@edu.uaa.mx

Hiram H. López
h.lopezvaldez@csuohio.edu

[1] Departamento de Ciencias de la Computación, Universidad Autónoma de Aguascalientes, Aguascalientes, 20100, Aguascalientes, México

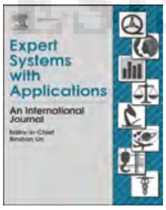
[2] Department of Mathematics and Statistics, Cleveland State University, Cleveland, 44115, OH, USA

Expert Systems With Applications 216 (2023) 119449

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



# Chain code strategy for lossless storage and transfer of segmented binary medical data

Erdoğan Aldemir [a,*], Osvaldo Arturo Tapia Dueñas [b], Ali Emre Kavur [a], Gulay Tohumoglu [c], Hermilo Sánchez-Cruz [b], Mustafa Alper Selver [c]

[a] *The Graduate School of Natural and Applied Sciences., Dokuz Eylül University, Tinaztepe Campus. No: 22, Izmir, Turkey*
[b] *Departamento de Ciencias de la Computación. Centro de Ciencias Básicas. Universidad Autónoma de Aguascalientes, Aguascalientes 20131, Ags., México*
[c] *Electrical and Electronics Engineering. Dept, Dokuz Eylül University., Tınaztepe Camp. No:22, Izmir, Turkey*

ARTICLE INFO

*Keywords:*
DICOM
Telemedicine
Lossless compression
Chain code
Visualization

ABSTRACT

Obtaining a 3D medical visualization is a tedious process requiring several processing steps (such as segmentation) and assigning various rendering parameters (such as color and opacity). Current systems use video/image exporting or snapshots to save results. Such vendor-dependent tools not only prevent the possibility of further interactions but also creates additional large-size data that is problematic to store in PACS over time and hard to transfer for teleradiology applications. To overcome, alternative strategies propose a representation of the visualizations, which only store segmentation masks that contains the binary form of segmented data. Unfortunately, existing compression methods are limited to effectively compress the volumetric data. In this study, lossless storage of binary segmented data is effectively performed by two newly-proposed chain code approaches. Particularly-two novel contributions are presented: 1. The dictionary of normalized angle difference is improved as a new chain symbol coding procedure, namely normalized angle difference, by adding new symbols to the dictionary aiming to generate a low-entropy symbol sequence for medical volumes. 2. A new volumetric approach that utilizes 26 symbols to encode volumetric data is developed. Each slice is visited, and the contour of the segmented object is codified such that eight different vectors for each slice (pointing to one of the four faces of each voxel, plus four towards one of its edges) are obtained. The developed methods are tested on diverse volumetric segmented data and compared to existing standards. It is shown that the proposed methods outperform well-established techniques.

## 1. Introduction

As medical imaging modalities advance to provide more detailed acquisitions, the amount of generated data is increasing drastically (Andriole et al., 2011; Gunderman and Chou, 2016; Norbash et al., 2014). This requires new tools and techniques for efficient data storage and fast and reliable distribution of data over teleradiology networks. The latter's importance is recognized, especially during the pandemic, when radiologists mostly worked remotely. Such conditions also require the remote collaboration of other clinicians with the radiologist(s) for diagnostic planning. Especially in complex cases, high-quality communication is necessary for handling multi-dimensional medical data, and to provide that, effective compression and distribution strategies are needed (Aldemir et al., 2020; Fischer et al., 2010, 2015).

3D visualization became a vital tool for analyzing tomographic medical data. Unfortunately, the generation of a clinically interpretable rendering requires a tedious procedure including several processing steps (such as segmentation) and adjustment of various parameters (i.e., optical and appearance-related parameters such as color and opacity) (Fischer et al., 2015). Moreover, reaching a consensus usually requires an iterative optimization process, during which the segmentation methods are executed several times prior to reaching desired

---

Peer review under responsibility of Submissions with the production note 'Please add the Reproducibility Badge for this item' the Badge and the following footnote to be added:The code (and data) in this article has been certified as Reproducible by the CodeOcean: https://codeocean.com. More information on the Reproducibility Badge Initiative is available at https://www.elsevier.com/physicalsciencesandengineering/computerscience/journals.

[*] Corresponding author.

*E-mail addresses:* erdogan.aldemir@deu.edu.tr (E. Aldemir), osvaldo.tapia@edu.uaa.mx (O. Arturo Tapia Dueñas), gulay.tohumoglu@deu.edu.tr (G. Tohumoglu), hermilo.sanchez@edu.uaa.mx (H. Sánchez-Cruz), alper.selver@deu.edu.tr (M. Alper Selver).

https://doi.org/10.1016/j.eswa.2022.119449
Received 21 March 2022; Received in revised form 21 November 2022; Accepted 16 December 2022

0957-4174/

# Agradecimientos

Esta tesis es el resultado de la generosa ayuda, apoyo e inspiración de diversas personas. En particular, deseo expresar mi profundo agradecimiento a mi tutor, el Dr. Hiram Habid López Valdez por brindarme la oportunidad de realizar una estancia de investigación en la Cleveland State University. Su guía invaluable me permitió obtener una oferta de trabajo como Assistant Professor en el Departamento de Mathematics, Computer Science, and Data Science en John Carroll University. Además, agradezco sinceramente todos los consejos y el arduo trabajo invertido en la publicación de artículos.

Quiero extender mi más sincero agradecimiento a mi tutor, el Dr. Hermilo Sánchez Cruz, por su dedicación, apoyo, aliento, orientación, paciencia y vasto conocimiento. Gracias por brindarme la oportunidad de trabajar junto a usted en un entorno óptimo de investigación, donde adquirí valiosos conocimientos y consejos. También le agradezco su valiosa contribución en la redacción de esta tesis y en la elaboración de artículos de investigación presentados en revistas indexadas.

Además, deseo expresar mi gratitud al CONACYT por el apoyo financiero otorgado, el cual fue fundamental para la realización de este trabajo. Por último, agradezco a la Universidad Autónoma de Aguascalientes por permitirme formar parte de este programa y brindarme un entorno propicio para mi desarrollo académico y profesional.

# Índice

# 1 Resumen

La tesis doctoral se centra en la visión por computadora y aborda la simplificación y compresión de datos en imágenes y objetos 2D y 3D. El primer capítulo presenta un enfoque novedoso basado en gramática libre de contexto para detectar segmentos rectos en imágenes y propone un método de aproximación poligonal para representar los segmentos lineales detectados. En el segundo capítulo, la gramática libre de contexto se extiende al caso tridimensional y se propone una técnica para simplificar objetos 3D utilizando nubes de puntos y códigos de cadena. El tercer capítulo propone un método de codificación de cadenas para comprimir imágenes médicas tridimensionales utilizando códigos de cadena y trayectorias helicoidales. Los resultados experimentales demuestran la efectividad y eficiencia de los métodos propuestos, avanzando en la vanguardia de la simplificación y compresión de datos en visión por computadora. La tesis ofrece nuevos enfoques y técnicas para la simplificación y compresión de datos en imágenes y objetos 2D y 3D, con aplicaciones en diversos campos.

# 2 Abstract

The doctoral thesis focuses on computer vision and addresses the simplification and compression of data in 2D and 3D images and objects. The first chapter presents a novel approach based on context-free grammar to detect straight segments in images and proposes a polygonal approximation method to represent the detected linear segments. In the second chapter, the context-free grammar extends to the three-dimensional case, and we proffer a technique to simplify 3D objects using point clouds and string codes. The third chapter proposes a string encoding method for compressing three-dimensional medical images using string codes and helical paths. Experimental results demonstrate the effectiveness and efficiency of the proposed methods, Advancing the cutting edge in data simplification and compression in computer vision. The thesis offers new approaches and techniques for data simplification and compression in 2D and 3D images and objects, with applications in various fields.

# 3 Introducción

La visión por computadora es una rama de la ciencia de la computación que se enfoca en desarrollar algoritmos y sistemas que permitan a las máquinas interpretar y comprender la información visual de las imágenes, vídeos y objetos 3D, de la misma manera que lo hace el cerebro humano.

En otras palabras, la visión por computadora busca enseñar a las computadoras a "ver" y "entender" las imágenes, vídeos y objetos 3D, realizando tareas como clasificación, simplificación, compresión, reconstrucción, entre otras. Esta tecnología tiene aplicaciones en diversas áreas, incluyendo robótica, reconocimiento facial, vigilancia, medicina y la industria automotriz, por mencionar algunas. Por ejemplo, la detección automática de tumores en imágenes médicas puede salvar vidas al permitir un diagnóstico temprano y un tratamiento eficaz. De manera similar, la detección automática de defectos en piezas de maquinaria en una línea de producción puede mejorar la calidad y la eficiencia de la producción.

La simplificación y compresión de datos es una tarea importante en visión por computadora. La simplificación de datos implica reducir la complejidad de una representación de datos sin perder demasiada información importante, mientras que la compresión de datos se enfoca en reducir el tamaño de los datos para que puedan ser almacenados o transmitidos de manera más eficiente.

En el contexto de la visión por computadora y el procesamiento de imágenes, la simplificación y compresión de datos son importantes porque las imágenes,vídeos y objetos 3D suelen ser muy grandes y complejos. Por ejemplo, una imagen de alta resolución puede contener millones de píxeles, lo que la hace difícil de procesar o almacenar. Además, estas representaciones visuales del mundo real se generan y se transmiten a través de redes de comunicaciones, por lo que su tamaño debe ser reducido para que se puedan transmitir de manera eficiente.

El presente trabajo de tesis se enfoca en la simplificación de objetos en 2D y 3D, así como en la compresión de imágenes médicas que representan objetos tridimensionales. En el segundo capítulo se describe la simplificación de imágenes binarias mediante la aproximación poligonal [1]. Para lograr esto, utilizamos nuestra gramática libre de contexto presentada en [2], que nos permitió detectar segmentos rectos en las imágenes. Luego, encontramos la aproximación poligonal óptima utilizando nuestros métodos para modificar los puntos y eliminar aquellos que no presentan información importante.

En el tercer capítulo se extiende la gramática libre de contexto al caso tridimensional [3], utilizando un nuevo parámetro denominado $\delta$ para controlar la cantidad de capas sobre las cuales se aplica la gramática. Además, se compró la invarianza ante rotación para asegurar que la secuencia de nubes de puntos 3D represente simplificaciones consistentes del objeto original, independientemente de su orientación. Este método permite obtener diferentes densidades de la nube de puntos, desde la más escasa hasta la más densa, modificando los valores de cuatro parámetros. Los resultados muestran que esta secuencia de nubes de puntos 3D representa diferentes simplificaciones del objeto original y cumple distintas necesidades, desde una representación detallada con una reducción mínima de puntos hasta una simplificación significativa que solo preserva las propiedades principales del objeto.

En el cuarto capítulo, se presenta la propuesta de dos métodos de codificación de cadena para la compresión de imágenes médicas tridimensionales [4]. El primer método mejora un diccionario de codificación de ángulos para generar una secuencia de símbolos más compacta, mientras que el segundo utiliza un camino helicoidal definido previamente [5] para representar objetos 3D con menos coordenadas. Los resultados obtenidos muestran que ambos métodos superan a las técnicas ya existentes en términos de eficiencia y calidad de la reconstrucción.

En resumen, este trabajo aborda la simplificación de objetos en 2D y 3D, así como la compresión de imágenes médicas tridimensionales, proponiendo métodos novedosos que mejoran el estado del arte en estas áreas.

# 4 Artículos derivados del trabajo de tesis

Osvaldo A Tapia-Dueñas and Hermilo Sánchez-Cruz. “Context-free grammars to detect straight segments and a novel polygonal approximation method”. In: *Signal Processing: Image Communication* 91 (2021), p. 116080.

Osvaldo A Tapia-Dueñas, Hermilo Sánchez-Cruz, and Hiram H López. “3D object simplification using chain code-based point clouds”. In: *Multimedia Tools and Applications* (2022), pp. 1–25.

Erdoğan Aldemir et al. “Chain code strategy for lossless storage and transfer of segmented binary medical data”. In: *Expert Systems with Applications* 216 (2023), p. 119449.

# 5 Gramáticas libres de contexto para detectar segmentos rectos y un nuevo método de aproximación poligonal

A continuación se presenta el artículo publicado, cuyo título es: **Context-free grammars to detect straight segments and a novel polygonal approximation method**

Signal Processing: Image Communication 91 (2021) 116080

Contents lists available at ScienceDirect

Signal Processing: Image Communication

journal homepage: www.elsevier.com/locate/image

# Context-free grammars to detect straight segments and a novel polygonal approximation method

Osvaldo A. Tapia-Dueñas, Hermilo Sánchez-Cruz *

*Universidad Autónoma de Aguascalientes, Aguascalientes 20131, Ags., México*

## ARTICLE INFO

*Keywords:*
Break points
Dominant points
Shortest path
Digital straight segments
Lost pixels
Tolerable error

## ABSTRACT

A new method for the polygonal approximation is presented. The method is based on the search for break points through a context-free grammar, that accepts digital straight segments with loss of information, as well as the decrease in the error committed employing the comparison of a tolerable error. We present an application of our method to different sets of objects widely used, as well as a comparison of our results with the best results reported in the literature, proving that our method achieves better values of error criteria. Besides, a new way to find polygonal approximations, with context-free grammars to recognize digital straight segments without loss of pixels, it is also addressed.

## 1. Introduction

In the topic of polygonal approximation for the shape of binary objects, in recent decades a large number of articles have been published. These methods seek to obtain an efficient tradeoff between different error criteria [1–15].

In 1954, Attneave [16] studied how the human being compresses the information he receives from the world through visual perception. This was achieved by considering perception as an information management process, where much of the perceived information is redundant, since the points with high curvature, that make up the contour of a shape, are abundant in information content, therefore, they are fit and sufficient to represent contour shapes. An example of this is reflected in the experiment of his classic cat image, in which he identifies points with a high curvature in a snapshot and is joined by segments of continuous straight lines.

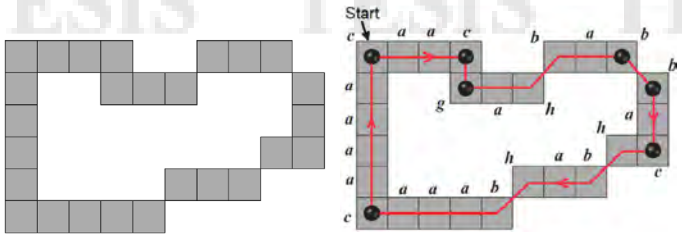
Following the Attneave experiment, various methods have been proposed for the presentation of contours of 2D objects by polygonal approximation, in which curve points (known as *dominant points*) and line segments are used to join them. This coding causes loss since it introduces some distortion errors. The error is permissible as long as the visual alteration is considered insignificant. Some algorithms look for minimize the number of dominant points (min-# problem) [17–19], Prasad [20] proposes a non-parametric framework to identify the dominant points using the Masood [9], RDP [21,22] and Carmona-Poyato et al. [10] methods. In [23] is proposed a non-optimal but unsupervised algorithm, called ICT-RDP, for the generation of polygonal approximations based on the convex hull. Others focus on decreasing the approximation error for a predefined number of vertices (problem min-$\varepsilon$) [11,13,24,25], Sarkar [3] uses the F8 chain code in addition to four propositions to determine the locations. Carmona-Poyato et al. [26] use a measure to identify what break points should be suppressed taking into account an allowable error. Some techniques seek to solve both the min-# and the min-$\varepsilon$ problems [4,5,15]. Masood [7] identifies the break points using the F8 chain code, in which recursively removes and rearranges the points. Kalaivani and Ray [15] propose a simple and easy technique for the detection of break points through the strategy of division (local deviation) and fusion (global deviation). As reported in [15], in some of these works, the number of dominant points is improved by sacrificing the integral square error (ISE), or vice versa. To determine polygonal approximations of a shape in [27], they develop an algorithm starting from coarse-level to more refined-level representation by varying the number of polygon sides. Ngo [28] proposes a fully discrete structure, based on the notion of blurred segments, to study the geometrical features on such curves and apply it in a process of polygonal approximation. Kai, Shurong and Zhongjian [29] develop a non-uniform rational B-spline interpolation scheme (NURBS). They fit a NURBS curve efficiently through an optimal polygonal approximation based on Mixed Integer Programming (MIP). Other authors use artificial intelligence, for example in [30], authors apply a coarse-grained parallel genetic algorithm (CGPGA) to solve polygonal approximation problem.

Some methods try to reduce the execution time [6,31,32]. Chetverikov and Szabó [33] propose a two-phase method, the first consists of using either inscribed triangles of the same size where vertices touch three points of the original contour or an angle where intersect two lines that touch two points of the original contour, the second phase is the elimination of dominant points using the Euclidean

---

* Corresponding author.
*E-mail address:* hsanchez@correo.uaa.mx (H. Sánchez-Cruz).

https://doi.org/10.1016/j.image.2020.116080
Received 25 May 2020; Received in revised form 26 September 2020; Accepted 22 November 2020



(a) (b)

**Fig. 1.** An example of a shape: (a) only its contour and (b) its break points.

distance. Nain, et al. [34], propose a method to detect break points in images with noise, they use the F8 chain code and the slope of straight lines. Other techniques investigate how to be robust to noise [35–37].

Another application of the polygonal approximation approach is for detection [38,39], Chabat, et al. [40] obtain the coordinates of the corners by detecting them based on intensity patterns that are anisotropic in several directions. Polygonal approximation is also used for recognition [37,41], Singh et al. [42] use the RDP algorithm [21,22] to obtain the break points needed in character recognition.

It is worth mentioning that while the use of the so-called *key points* focuses on segmentation and classification problems in images, the problem of detecting dominant points is carried out on closed curves, *i.e.*, object contours extracted from an image. On the one hand, a number of works have appeared for the segmentation and extraction of object contours, *e.g.*, authors in [43] propose an edge-based superpixel similarity measurement, which globally evaluates the similarity between superpixels by binary edge maps. Convolutional neural networks (CNN) have also been used for the same purpose [44–50]. On the other hand, there are studies that use key points to detect contours, relying on modeling the problem using graph theory [51]. In the present research, once a contour of a scene has been extracted from an image, the issue of dealing with the optimal search for the so-called dominant points is tackled.

Although there is previous work to solve graph-based problems using neural networks [52]. Recently, research in deep learning for graph-based problems optimization has been developed [53–55]. Even more, although there is recent research that addresses the problem of key point detection using deep learning, until now, CNN-based methods have not been developed to search for dominant point detection, nor have they been developed to optimize the error criteria that we have addressed.

This paper proposes a new method for solving the min-# and min-$\varepsilon$ problems using the language proposed in [56], the Dijkstra algorithm [57] for the rearrangement of break points and a novel method for the elimination of break points.

In Section 2 the main concepts and definitions used in this paper are presented, whereas in Section 3 our proposed method to obtain the dominant points is explained. On the other hand, in Section 4 the application of our proposed method as well as the results are presented. Finally, some conclusions and further work are given in Section 5. To reproduce our results an open-source software is available.[1]

## 2. Concepts and definitions

In this section, a set of concepts and definitions that are used throughout the paper are given.

**Definition 2.1.** The points of a digital contour shape are given by

$$C = \{P_i(x_i, y_i) | i = 1, 2, \dots n\}, \tag{1}$$

such that $P_{i-1}$ and $P_{i+1}$ are neighboring points of $P_i$, and the contour shape is of size $n$.

**Definition 2.2.** A subset of $C$, called break points, is given by

$$B = \{P^k \in C | k = 1, 2, \dots m, m \leq n\}, \tag{2}$$

**Definition 2.3.** The final break points achieved from a series of procedures to reduce them, are called dominant points (DP).

On the other hand, the following definitions are important for handling the error criteria calculated in this paper.

**Definition 2.4.** Compression ratio is defined as the ratio between the size of the contour shape and the number of DP, *i.e.*,

$$\mathrm{CR} = \frac{n}{\mathrm{DP}}. \tag{3}$$

**Definition 2.5.** Given two break points $(x_k, y_k)$ and $(x_{k+1}, y_{k+1})$, a continuous-line segment is defined. The distance between this segment and the points of the contour cells is given by

$$d^2(p_i, \overline{p_k p_{k+1}}) = \frac{((x_i - x_k)(y_{k+1} - y_k) - (y_i - y_k)(x_{k+1} - x_k))^2}{(x_i - x_{k+1})^2 + (y_i - y_{k+1})^2}. \tag{4}$$

**Definition 2.6.** The integral square error is the summation of the square distance from the edges of approximating polygon to the contour points, and is given by

$$\mathrm{ISE} = \sum_{i=1}^{n} d_i^2. \tag{5}$$

where $d_i$ is the distance from the $i$-th point cell of the contour to the approximating polygon.

**Definition 2.7.** The figure of merit (FOM) defined by [3] is shown in Eq. (6).

$$\mathrm{FOM} = \frac{n}{(\mathrm{DP})(\mathrm{ISE})}, \tag{6}$$

**Definition 2.8.** The weighted sum of squared error is defined in [17,58].

$$\mathrm{WE}_i = \frac{\mathrm{ISE}}{\mathrm{CR}^i}, \tag{7}$$

where authors in [17] use $i = \{1, 2, 3\}$.

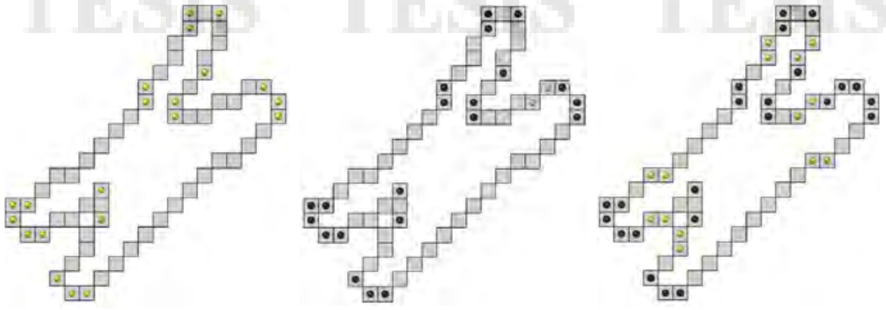
Although different error criteria are used to prove polygonal approximation methods, in our work we focus on look for a fair tradeoff between the number of dominant points (DP), compression ratio (CR), integral square error (ISE), figure of merit (FOM) and weighted sum of squared error (WE). In addition, we analyze the number of lost pixels (LP), caused when enough number of DP are used to recover the original contour with lossless of information.

## 3. Proposal method to obtain dominant points

In this section, we present our method for obtaining the DP, which is achieved, first, by obtaining the break points, then, by eliminating those that are unnecessary, and finally, by rearranging the remaining ones in an optimal way.

---

[1] https://github.com/OsvoT/Dominant-Points.



(a) (b) (c)

**Fig. 2.** First set of break points for *Chromosome*, when: (a) $r = 2$, (b) $r = 1$, (c) $r = 0$.

**Table 1**
Comparison of error criteria due to different methods.

| Shape | Method | DP | ISE | CR | FOM | WE | WE$_2$ |
|---|---|---|---|---|---|---|---|
| *Chromosome* ($n = 60$) | Wu [58] | 16 | 4,700 | 3,750 | 0,798 | 1,253 | 0,334 |
| | Marji and Siy [25] | **10** | 10,010 | 6,000 | 0,599 | 1,668 | 0,278 |
| | Masood [8] | 15 | 3,880 | 4,000 | 1,031 | 0,970 | 0,243 |
| | Carmona-Poyato et al. [10] | 15 | 4,270 | 4,000 | 0,937 | 1,068 | 0,267 |
| | Nguyen and Debled-Rennesson [13] | 18 | 4,060 | 3,333 | 0,821 | 1,218 | 0,365 |
| | Parvez [19] | 11 | 7,090 | 5,455 | 0,769 | 1,300 | **0,238** |
| | Kalaivani and Ray [15] | 11 | 7,770 | 5,455 | 0,702 | 1,425 | 0,261 |
| | Proposed | 16 | **3,546** | 3,750 | **1,058** | **0,945** | 0,252 |
| *Leaf* ($n = 120$) | Wu [58] | 24 | 15,930 | 5,000 | 0,314 | 3,186 | 0,637 |
| | Marji and Siy [25] | **17** | 28,670 | 7,059 | 0,246 | 4,062 | 0,575 |
| | Masood [8] | 23 | 9,460 | 5,217 | 0,552 | 1,813 | **0,348** |
| | Carmona-Poyato et al. [10] | 23 | 10,680 | 5,217 | 0,489 | 2,047 | 0,392 |
| | Nguyen and Debled-Rennesson [13] | 33 | **5,56** | 3,636 | **0,654** | **1,529** | 0,420 |
| | Parvez [19] | 21 | 11,980 | 5,714 | 0,477 | 2,097 | 0,367 |
| | Kalaivani and Ray [15] | 20 | 14,720 | 6,000 | 0,408 | 2,453 | 0,409 |
| | Proposed | 24 | 8,994 | 5,000 | 0,556 | 1,799 | 0,360 |
| *Semicircle* ($n = 102$) | Wu [58] | 26 | 9,040 | 3,923 | 0,434 | 2,304 | 0,587 |
| | Marji and Siy [25] | 15 | 22,700 | 6,800 | 0,300 | 3,338 | 0,491 |
| | Masood [8] | 26 | **4,05** | 3,923 | **0,969** | **1,032** | **0,263** |
| | Carmona-Poyato et al. [10] | 26 | 4,910 | 3,923 | 0,799 | 1,252 | 0,319 |
| | Nguyen and Debled-Rennesson [13] | 25 | 5,420 | 4,080 | 0,753 | 1,328 | 0,326 |
| | Parvez [19] | 15 | 18,220 | 6,800 | 0,373 | 2,679 | 0,394 |
| | Kalaivani and Ray [15] | **13** | 27,580 | 7,846 | 0,284 | 3,515 | 0,448 |
| | Proposed | 26 | 4,054 | 3,923 | 0,968 | 1,033 | **0,263** |
| *Infinity* ($n = 45$) | Wu [58] | 13 | 5,780 | 7,846 | 1,357 | 0,737 | 0,094 |
| | Masood [8] | 11 | 2,900 | 9,273 | 3,197 | 0,313 | 0,034 |
| | Carmona-Poyato et al. [10] | 10 | 5,290 | 10,200 | 1,928 | 0,519 | 0,051 |
| | Parvez [19] | **7** | 7,690 | 14,571 | 1,895 | 0,528 | 0,036 |
| | Kalaivani and Ray [15] | 9 | 5,260 | 11,333 | 2,155 | 0,464 | 0,041 |
| | Proposed | 12 | **2,395** | 8,500 | **3,549** | **0,282** | **0,033** |

### 3.1. Break points detection

To obtain our first set of break points, we find the endpoints of the digital straight segments (DSS). To do this, we encode the shape of the contour using the Freeman's Angle chain code of eight directions (AF8) [56,59]. Then, we obtain the break points by looking for the symbol strings that belong to the subset of a context-free grammar [56], given by Eq. (8), which geometrically represents DSS.

$$L = \{xa^p(bha^q)^r, xa^p(hba^q)^r \mid x \in \{a, b, c, d, e, f, g, h\}\}, \tag{8}$$

where $p, q, r$ are integers that indicate the number of times the symbol or substring in parentheses appears concatenated, $x$ is the label for the break points and $a, b, \dots, h$ are symbols of the AF8 code. Consider a contour of size $n$, like the one shown in Fig. 1(a). Starting with the upper and leftmost pixel, its AF8 chain code is $S_{AF8} = caacgahbabbachbahbaaacaaaa$, which also can be written as $S_{AF8} = ca^2cgahbabbachbahba^3ca^4$. Fig. 1(b) shows the path followed to obtain the chain code. As can be observed, the chain code has the same form of any of the strings of Eq. (8). It can be written as $S_{AF8} = \underbrace{xa^2}\underbrace{x}\underbrace{xahba}\underbrace{x}\underbrace{xa}\underbrace{xhbahba^3}\underbrace{xa^4} = \underbrace{xa^p}\underbrace{x}\underbrace{xa^p(hba^q)^r}\underbrace{x}\underbrace{xa^p}\underbrace{x(hba^q)^r}\underbrace{xa^p}$, where every $x$ symbol represents a break point (*i.e.*, there are seven DSS in this example) and parameters $p, q$ and $r$ satisfy $0 \le p \le 4$, $0 \le q \le 3$, $0 \le r \le 2$. Fig. 1(b) shows the obtained break points depicted in dark circles.

When looking for the polygon that adequately represents the shape of the object, one of the problems to face is to decrease the ISE as much as possible, without increasing the number of DP too much. Part of our method is to impose a tolerable error, even close to those obtained by recent authors.

We apply the detection of break points as follows.

Calculate the maximum values of $p$, $q$ and $r$: $p_{max}, q_{max}$ and $r_{max}$, respectively.

Find the first set of break points as explained above. Let $d(P_k, P_{k+1})$ be the Euclidean distance between two break points, and $s(P_k, P_{k+1})$ the perimeter chord that connects the same points. Define a tolerable error, T, and share it with the errors committed by the break points, using the Eq. (9).

$$\text{ISE}_{k,k+1} \le \frac{\text{T}}{sd}, \tag{9}$$

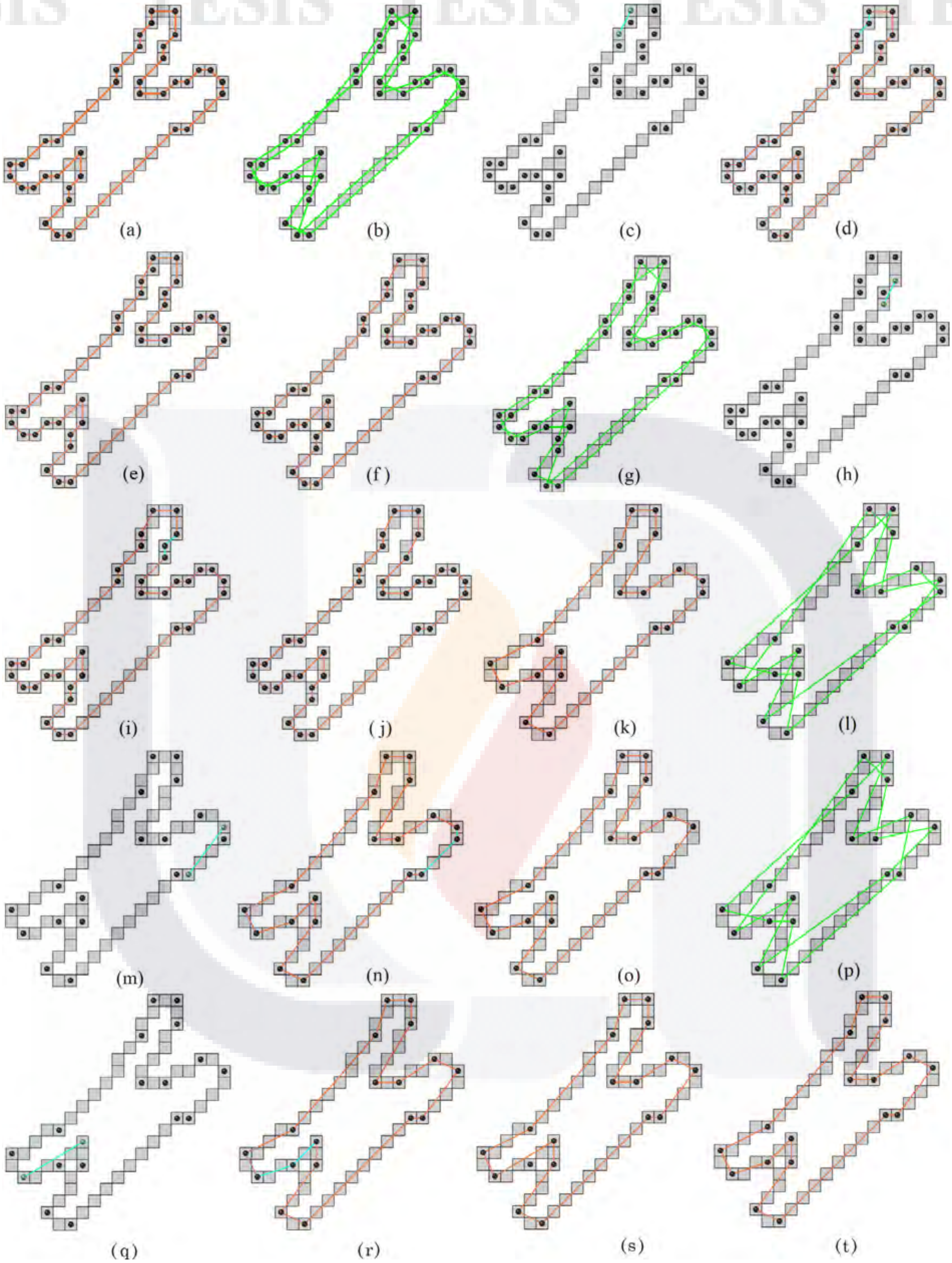


**Fig. 3.** Break points suppression example.

where T is associated to each segment $\overline{P_k, P_{k+1}}$, *i.e.*, the value of T is approximately distributed to each error area by the product $sd$. The denominator $sd$, comes from the knowledge that the area subtended by an arc $s$, with radius $r$, is smaller than the product $rs$. Particularly, the area that subtends half a circle is $A = \pi r^2/2$, while the product of the arc by the diameter is $A' = 2\pi r^2$, where clearly $A \leq A'$. Therefore, in general, $sd$ is greater than the error committed for the segment $\overline{P_k, P_{k+1}}$. Of course, T allows to calculate the details of the shape as it is handled. The smaller T is, the more break points appear and therefore a more detailed representation of the original contour can be obtained.

For segments that do not satisfy the inequality of Eq. (9) do: if $r/2 \leq 0.5$ then $r \rightarrow 0$, otherwise $r \rightarrow round(r/2)$, where round() rounds to the floor when decimal part of $r/2$ is smaller than 5, otherwise rounds to the ceiling. This assignment allows the algorithm to work well on low curvatures, causing more break points to appear.

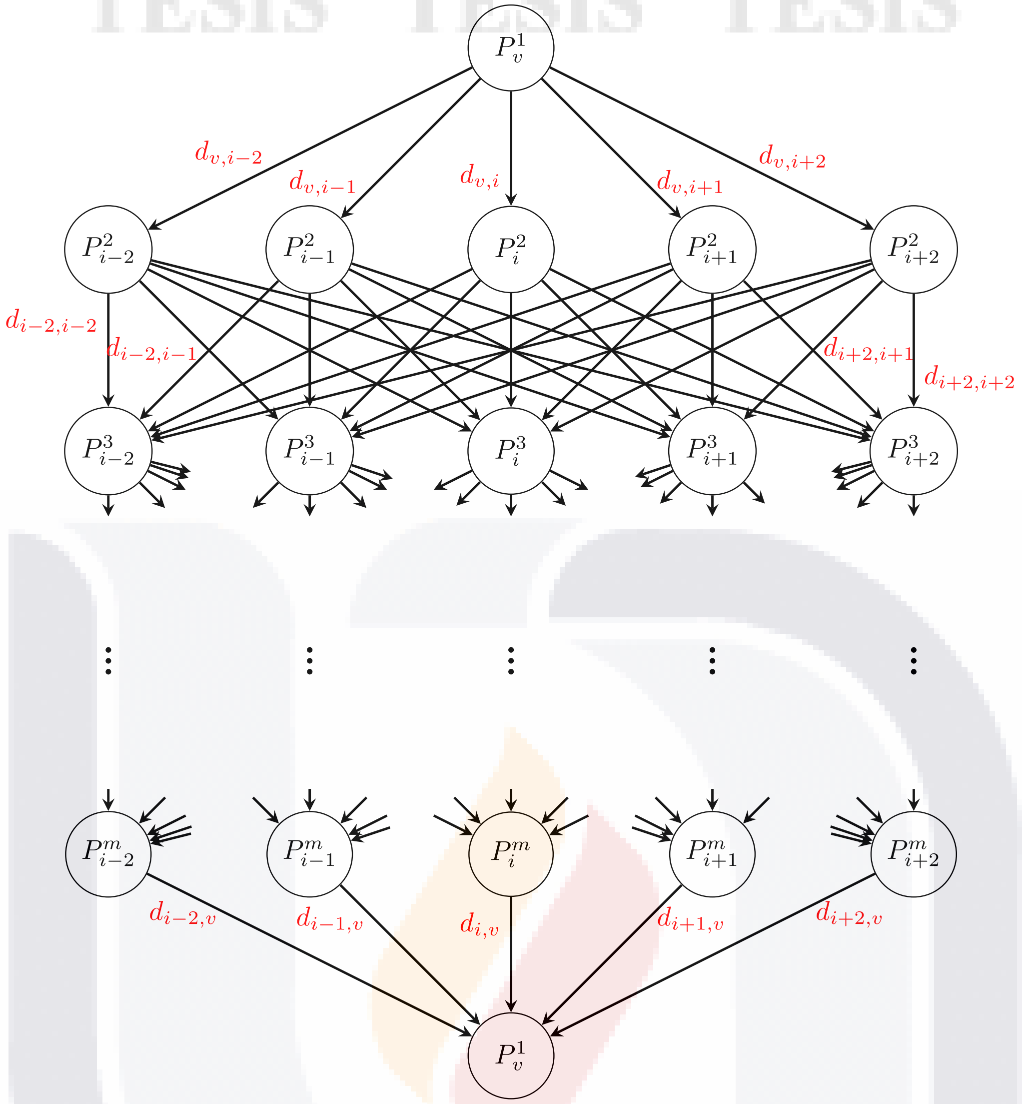


**Fig. 4.** Representation of the problem for obtaining the best polygonal approximation.

**Table 2**
Comparisons of the proposed method with other polygonal approximation methods.

| Shape | Method | DP | ISE | CR | FOM | WE | $WE_2$ |
|---|---|---|---|---|---|---|---|
| *Shark* | Algo 1 | 23 | 76,68 | 12,74 | 0,166 | 6,019 | 0,472 |
| $n = 293$ | APS | 21 | 72,70 | 13,95 | 0,192 | 5,211 | 0,373 |
| | FDP | **19** | 105,00 | 15,42 | 0,147 | 6,809 | 0,442 |
| | Proposed | 20 | **66,91** | 14,65 | **0,219** | **4,567** | **0,312** |
| *Cup* | Algo 1 | 21 | 160,70 | 19,29 | 0,120 | 8,333 | 0,432 |
| $n = 405$ | APS | **11** | 238,45 | 36,82 | 0,154 | 6,476 | 0,176 |
| | FDP | 17 | 159,56 | 23,82 | 0,149 | 6,697 | 0,281 |
| | Proposed | 12 | **99,39** | 33,75 | **0,340** | **2,945** | **0,087** |
| *Stingray* | Algo 1 | 25 | 118,66 | 13,12 | 0,111 | 9,044 | 0,689 |
| $n = 328$ | APS | 23 | 121,38 | 14,26 | 0,117 | 8,511 | 0,597 |
| | FDP | **20** | 165,40 | 16,40 | 0,099 | 10,085 | 0,615 |
| | Proposed | **20** | **106,47** | 16,40 | **0,154** | **6,492** | **0,396** |
| *Heart* | Algo 1 | 37 | 71,21 | 8,19 | 0,115 | 8,695 | 1,062 |
| $n = 303$ | APS | **17** | 85,67 | 17,82 | 0,208 | 4,807 | **0,270** |
| | FDP | 30 | 68,01 | 10,10 | 0,149 | 6,734 | 0,667 |
| | Proposed | 20 | **63,36** | 15,15 | **0,239** | **4,182** | 0,276 |

As an example, consider the *Chromosome* shape given in Fig. 2. Starting from the top and leftmost pixel, its AF8 code is given by

$C = \boldsymbol{c}a\boldsymbol{c}abh\boldsymbol{bahg}ahbhb\boldsymbol{bbb}aabhaaaaaaaa\boldsymbol{bbc}ahb\boldsymbol{haf}abh\boldsymbol{bbbch}abhaaaa\boldsymbol{hb}ahb\boldsymbol{h}$, whose length is $n = 60$.

Note that the symbols in bold correspond to the $x$ symbol of Eq. (8). This is represented in pixels with yellow balls in Fig. 2(a). Also, the maximum number of concatenations of $a$, after one $x$, is two, while after a substring $hb$ (or $bh$) is eight. We also note that the maximum number of concatenated substrings containing $hb$ (or $bh$) is two; thus, $(p, q, r) = (2, 8, 2)$. On the other hand, there may be segments such that the ISE is greater than the tolerable error shared between the break points, given by Eq. (9), with which we have the possibility of obtaining more break points. Therefore, we apply: if $r/2 \leq 0.5$ then $r \rightarrow 0$, otherwise $r \rightarrow round(r/2)$. Fig. 2(b) shows, in yellow balls, how more break points arise. We repeat the above procedure until Eq. (9) fails. Fig. 2(c) shows the case in which $r = 0$.

At the end of the process, Eq. (10) must be satisfied.

$$\mathrm{ISE}_{1,2} + \mathrm{ISE}_{2,3} \dots \mathrm{ISE}_{m,1} \leq \mathrm{T}. \tag{10}$$

### 3.2. Break points elimination

To obtain a final small number of DP, properly reduce the number of break points as follows:

1. For break points $P^1, P^2, \dots P^{m-1}, P^m$ make a list, $List1 = [\mathrm{ISE}_{1,2}, \mathrm{ISE}_{2,3}, \dots, \mathrm{ISE}_{m,1}]$.
   For break points $P^1, P^3, P^2, P^4, \dots, P^{m-1}, P^1, P^m, P^2$ make another list, $List2 = [\mathrm{ISE}_{1,3}, \mathrm{ISE}_{2,4}, \dots, \mathrm{ISE}_{m-1,1}, \mathrm{ISE}_{m,2}]$.
2. From $List2$, choose the smallest $\mathrm{ISE}_{k,k+2}$ that we define as $\mathrm{ISE}_{Min}$.

**Table 3**
Comparison of error criteria using a third set of shapes.

Fernández-García et al. [14]:

| Shape | $n$ | DP | CR | ISE | FOM | WE | $WE_2$ |
|---|---|---|---|---|---|---|---|
| *Bell-7* | 406 | 23 | 17,652 | 166,207 | 0,106 | 9,416 | 0,533 |
| *Device6-9* | 1590 | 33 | 48,18 | 370,277 | 0,130 | 7,685 | 0,160 |
| *Ray-17* | 689 | 35 | 19,686 | 242,437 | 0,081 | 12,315 | 0,626 |
| *Truck-07* | 277 | 40 | 6,925 | 23,983 | 0,289 | 3,463 | 0,500 |
| *Bell-10* | 1202 | 42 | 28,619 | 694,800 | 0,041 | 24,278 | 0,848 |

Kalaivani and Ray [15]:

| Shape | $n$ | DP | CR | ISE | FOM | WE | $WE_2$ |
|---|---|---|---|---|---|---|---|
| *Bell-7* | 406 | **20** | 20,300 | 166,855 | 0,122 | 8,219 | 0,405 |
| *Device6-9* | 1590 | 33 | 48,182 | 308,990 | 0,156 | 6,413 | 0,133 |
| *Ray-17* | 689 | **30** | 22,967 | 318,591 | 0,072 | 13,872 | 0,604 |
| *Truck-07* | 277 | 40 | 6,925 | 22,793 | 0,304 | 3,291 | 0,475 |
| *Bell-10* | 1202 | 36 | 33,389 | 932,880 | 0,036 | 27,940 | 0,837 |

Proposed:

| Shape | $n$ | DP | CR | ISE | FOM | WE | $WE_2$ |
|---|---|---|---|---|---|---|---|
| *Bell-7* | 406 | **20** | 20,35 | **147,206** | **0,238** | **4,197** | **0,278** |
| *Device6-9* | 1590 | **32** | 49,688 | **287,8801** | **0,173** | **5,794** | **0,117** |
| *Ray-17* | 689 | **30** | 22,967 | **221.946** | **0,104** | **9,664** | **0,421** |
| *Truck-07* | 277 | **36** | 7,694 | **21,917** | **0,351** | **2,848** | **0,370** |
| *Bell-10* | 1202 | **35** | 34,4 | **573.018** | **0,060** | **16,658** | **0,484** |

**Table 4**
Comparison of error criteria due to different methods.

| | DP | ISE | CR | FOM | WE | $WE_2$ | | DP | ISE | CR | FOM | WE | $WE_2$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *Africa*; $n = 291$ | | | | | | | *Plane1*; $n = 462$ | | | | | | |
| Carmona | **26** | 93,62 | 11,19 | 0,12 | 8,36 | 0,75 | Carmona | **37** | 104,82 | 12,49 | 0,12 | 8,39 | 0,67 |
| RDP | 38 | 37,02 | 7,66 | 0,21 | 4,83 | 0,63 | RDP | 40 | 67,57 | 11,55 | 0,17 | 5,85 | 0,51 |
| Masood | 39 | 25,89 | 7,46 | 0,29 | 3,47 | 0,47 | Masood | 52 | 31,49 | 8,88 | 0,28 | 3,54 | **0,40** |
| Proposed | 38 | **22,89** | 7,74 | **0,34** | **2,96** | **0,38** | Proposed | 53 | **30,53** | 8,74 | **0,29** | **3,49** | **0,40** |
| *Maple leaf*; $n = 424$ | | | | | | | *Plane3*; $n = 431$ | | | | | | |
| Carmona | **53** | 91,78 | 8,00 | 0,09 | 11,47 | 1,43 | Carmona | **39** | 92,99 | 11,05 | 0,12 | 8,41 | 0,76 |
| RDP | 58 | 53,11 | 7,31 | 0,14 | 7,27 | 0,99 | RDP | 45 | 56,46 | 9,58 | 0,17 | 5,89 | 0,62 |
| Masood | 105 | **15,29** | 4,04 | 0,26 | 3,79 | 0,94 | Masood | 54 | 32,06 | 7,98 | 0,25 | 4,02 | 0,50 |
| Proposed | 105 | 15,36 | 4,04 | **0,27** | **3,78** | **0,93** | Proposed | 52 | **29,8** | 8,29 | **0,28** | **3,60** | **0,43** |
| *Rabbit*; $n = 293$ | | | | | | | *Screw driver*; $n = 253$ | | | | | | |
| Carmona | **38** | 66,64 | 7,71 | 0,12 | 8,64 | 1,12 | Carmona | **14** | 138,59 | 18,07 | 0,13 | 7,67 | 0,42 |
| RDP | 40 | 35,98 | 7,33 | 0,20 | 4,91 | 0,67 | RDP | 16 | 34,52 | 15,81 | 0,46 | 2,18 | 0,14 |
| Masood | 45 | 21,67 | 6,51 | 0,30 | 3,33 | 0,51 | Masood | 16 | 24,68 | 15,81 | 0,64 | 1,56 | 0,10 |
| Proposed | 43 | **21,2** | 6,81 | **0,32** | **3,11** | **0,46** | Proposed | 15 | **23,18** | 19,93 | **0,73** | **1,37** | **0,08** |
| *Dog*; $n = 343$ | | | | | | | *Tin opener*; $n = 278$ | | | | | | |
| Carmona | 54 | 54,26 | 6,35 | 0,12 | 8,54 | 1,34 | Carmona | **39** | 73,08 | 7,13 | 0,10 | 10,25 | 1,44 |
| RDP | **52** | 44,81 | 6,60 | 0,15 | 6,79 | 1,03 | RDP | 42 | 33,99 | 6,62 | 0,19 | 5,14 | 0,78 |
| Masood | 54 | 32,51 | 6,35 | **0,20** | 5,12 | 0,81 | Masood | 45 | 20,69 | 6,18 | 0,30 | 3,35 | 0,54 |
| Proposed | 55 | **32,10** | 6,33 | 0,19 | **5,07** | **0,80** | Proposed | 43 | **20,59** | 6,76 | **0,42** | **2,40** | **0,36** |
| *Hammer*; $n = 388$ | | | | | | | *Plane4*; $n = 450$ | | | | | | |
| Carmona | **14** | 57,96 | 27,71 | 0,48 | 2,09 | 0,08 | Carmona | **41** | 127,93 | 6,78 | 0,05 | 18,87 | 2,78 |
| RDP | 16 | 55,17 | 24,25 | 0,44 | 2,28 | 0,09 | RDP | 46 | 68,62 | 6,04 | 0,09 | 11,35 | 1,88 |
| Masood | 15 | 38,35 | 25,87 | 0,67 | 1,48 | **0,06** | Masood | 68 | 16,92 | 4,09 | 0,24 | 4,14 | 1,01 |
| Proposed | 15 | **37,75** | 25,87 | **0,69** | **1,45** | **0,06** | Proposed | 67 | **16,24** | 6,76 | **0,42** | **2,40** | **0,36** |

3. From *List*1, let $\mathrm{ISE_T} = \sum_{k=1}^{m} \mathrm{ISE_{k,k+1}}$, and update as follows,

   $\mathrm{ISE_T} \rightarrow \mathrm{ISE_T} - (\mathrm{ISE_{k,k+1}} + \mathrm{ISE_{k+1,k+2}})$,

   and then,

   $\mathrm{ISE_T} \rightarrow \mathrm{ISE_T} + \mathrm{ISE_{Min}}$

4. If ($\mathrm{ISE_T} \leq \mathrm{T}$) then remove $\mathrm{P}_{k+1}$, remove $\mathrm{ISE}_{k,k+1}$ and $\mathrm{ISE}_{k+1,k+2}$ from *List*1 and add $\mathrm{ISE_{Min}}$. Go to step 1. Else, stop.

As an example of the break points elimination, Fig. 3 illustrates part of the algorithm to follow. The ISE values in *List*1 of initial break points are placed, while in *List*2 the errors are ordered alternately by pairs of odd and even break points. Part of this step corresponds to the case of Fig. 3(a) and (b). Considering T = 4, Fig. 3(a) shows the polygon obtained from the break points, which were found with the language $L$ of Eq. (8) as well as the condition given in Eq. (9), whereas in Fig. 3(b) *List*2 of Step 1 is obtained.

Next, the lower ISE value, $\mathrm{ISE_{Min}}$, and its corresponding break points pair are searched. See Fig. 3(c). On the other hand, Fig. 3(d) and (e) show that Step 3 is executed and the value of the errors between the three break points is subtracted, whereas the error due to the two contiguous break points are added in *List*1. Once the total error is less than the tolerable one, the intermediate break point is removed. This is shown in Fig. 3(f), which corresponds to Step 4. The algorithm is repeated until the condition of Step 4 is not satisfied. The next iteration is given by Steps 1, 2, 3 and 4, which correspond to Fig. 3(g), (h), (i) and (j), respectively. And so on. The penultimate iteration that corresponds to Step 1, is shown in Fig. 3(k) and (l), whereas Step 2

**Table 5**
Comparison of error criteria due to each part of the proposed methods.

| | DP | ISE | CR | FOM | WE | $WE_2$ | | DP | ISE | CR | FOM | WE | $WE_2$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *Chromosome*; T = 3,88 | | | | | | | *Cup*; T = 159,56 | | | | | | |
| BPs detection | 37 | 0 | 1,62 | ∞ | 0 | 0 | BPs detection | 127 | 9,65 | 3,18 | 0,33 | 3,03 | 0,95 |
| BPs detection + BPs elimination | 18 | 3,33 | 3,33 | 1 | 1 | 0,3 | BPs detection + BPs elimination | 13 | 123,39 | 31,08 | 0,25 | 3,96 | 0,13 |
| BPs detection + BPs elimination + BPs Rearrangement | 16 | 3,5 | 3,75 | 1,07 | 0,93 | 0,25 | BPs detection + BPs elimination + BPs Rearrangement | 12 | 99,39 | 33,67 | 0,34 | 2,95 | 0,09 |
| *Africa*; T = 25,89 | | | | | | | *Device6-9*; T = 330,00 | | | | | | |
| BPs detection | 113 | 3,74 | 2,6 | 0,7 | 1,45 | 0,56 | BPs detection | 360 | 84.82 | 4,42 | 0,01 | 19,2 | 4,35 |
| BPs detection + BPs elimination | 43 | 23,97 | 6,84 | 0,29 | 3,54 | 0,52 | BPs detection + BPs elimination | 35 | 324,8 | 45,43 | 0,02 | 7,15 | 0,16 |
| BPs detection + BPs elimination + BPs Rearrangement | 38 | 22,89 | 7,74 | 0,34 | 2,99 | 0,39 | BPs detection + BPs Elimination + BPs Rearrangement | 32 | 287,88 | 49,69 | 0,04 | 5,8 | 0,12 |

**Table 6**
Context-free grammars that recognize DSS without loss of pixels and some examples of DSS.

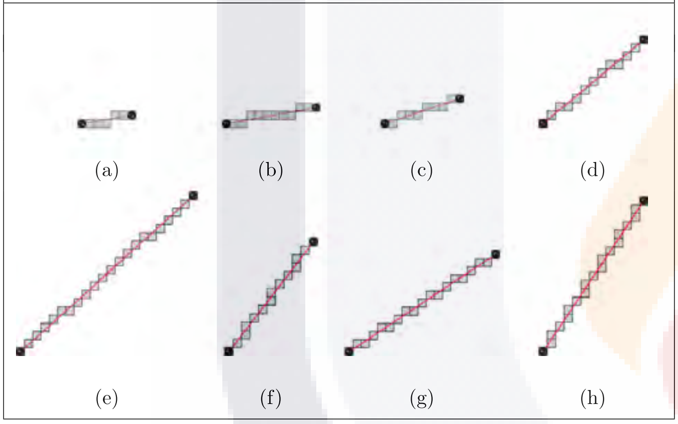
| (1) $xa^p$, where $0 \le p$ |
|---|
|  $p = 5$ $p = 4$ |
| (2) $xa^p(bha^q)^r bha^t$, where $0 \le t$, $\lvert p - t \rvert \le 1$, $2p < q \le 2(p+1)$ and $0 \le r$ |
|  (a) (b) (c) (d) (e) (f) (g) (h) |

**Table 7**
Parameters of the context-free grammars used in examples given in Table 6.

| | *p* | *q* | *r* | *t* |
|---|---|---|---|---|
| (a) | 2 | 3 | 0 | 1 |
| (b) | 1 | 4 | 1 | 1 |
| (c) | 0 | 1 | 2 | 0 |
| (d) | 1 | 4 | 1 | 2 |
| (e) | 4 | 9 | 1 | 4 |
| (f) | 1 | 2 | 2 | 2 |
| (g) | 0 | 1 | 5 | 0 |
| (h) | 0 | 1 | 5 | 0 |

is shown in Fig. 3(m) and (n), and Steps 3 and 4 are illustrated in Fig. 3(o) and (p), respectively. The last iteration of Step 1 is illustrated in Fig. 3(p), whereas Fig. 3(q) and (r) illustrates Step 2. Finally Fig. 3(s) and (t), correspond to Steps 3 and 4, respectively. Fig. 3(t) shows that the final break points are 18.

The previous algorithm allows us to eliminate break points until the ISE is greater than the tolerable. However, although the above algorithm reduces break points by a certain amount, the following procedure allows us to reduce them even further.

### 3.3. Break points rearrangement

In order to obtain a polygon with the smallest possible perimeter, a closed path must be established, which begins and ends at the same starting point. Of course, the search for the shortest path results in the rearrangement of the break points obtained from the elimination described in the previous subsection.

Take the option of moving from each break point to one of the neighboring points of the next break point. Our experiments suggest that they may be within a radius of 2 of each break point. Then, for each break point, $P^k$, there are five candidate points to be the next break point, thus forming five possible graphs, whose initial nodes are $P^1_{n-1}$, $P^1_n$, $P^1_1$, $P^1_2$ and $P^1_3$, where $P^1_{n-1}$, $P^1_n$, $P^1_2$ and $P^1_3$ are neighboring points of $P^1_1$, in a neighborhood of radius 2, respectively.

The problem of achieving a minimum closed path, from the starting point to itself, produced by the approximate polygon, is equivalent to finding the minimum distance, with the minimum ISE value due to the *m* break points on the above-mentioned graphs. Fig. 4 shows the graphs produced, in which $v = \{n-2, n-1, 1, 2, 3\}$ in $P^1_v$.

**Algorithm 1:** General pseudocode

```
Obtain $p_{max}, q_{max}, r_{max}$;
do
    Find break points using context-free grammar with $p_{max}, q_{max}, r_{max}$;
    Calculate ISE for all break points and record it in $List1$;
    $r_{max} \rightarrow r_{max}/2$
while there is a $List1[i] > T/sd$;
$L' \rightarrow \{\}$;
do
    if $L'$ is not empty then
        $List1 \rightarrow L'$;
    end
    Break points elimination();
    Break points rearrangement() ;
    Calculate ISE for all break points and record in $L'$;
while $L' \neq List1$;
```

Applying the Dijkstra algorithm [57] we can obtain the set of *m* break points that conform to the polygon with the smallest possible perimeter. For this, we find the minimum distance due to each initial point $P^1_v$ to itself: $min\{d(P^1_v, P^1_v)\}$, and, finally, we search the shortest path (not the trivial) of the five cases: $\underset{v}{min}\{min\{d(P^1_v, P^1_v)\}\}$.

Algorithm 1 summarizes the steps followed to obtain the DP explained in this section.

## 4. Experiments and results

We applied our proposed method to different sets of shapes used in the literature. The results from the first set, given by the common

(a) (b) (c) (d)

**Fig. 5.** Comparison methods: (a) Algo1, (b) APS, (c) FDP and (d) Proposed.

shapes, such as *Chromosome, Leaf, Semicircle* and *Infinity*, compared to other authors are presented in Table 1.

Of course, CR is inversely proportional to DP. Therefore, we consider one of the two parameters as significant, for example DP.

As we can observe, our method is the one that most times appears with the best parameters, mainly in ISE, FOM, WE and $WE_2$.

On the other hand, we tested our method for objects with greater resolution. The comparison with Algo 1, APS (applying automatic simplification process) and FDP (fixing the desired number of DP) reported recently by Nasser et al. [60] is given in Table 2, while Fig. 5 shows the best approximate polygons due to each of the methods. From the results of this test it follows that, in the case of the DP criterion, APS is better for two cases: *Cup* and *Heart*, while FDP is better for the other two: *Shark* and *Stingray*. However, our method has better error criteria in ISE, FOM, WE and $WE_2$ for all objects except *Heart*, in which APS is best for $WE_2$.

Comparing our method with another set of shapes, Table 3 shows the results with some MPEG-7 dataset shapes and the comparison with Fernández- García [14] and Kalaivani and Ray [15]. As a result of this experiment, it can be seen that our method achieves 100% of the best error criteria, concerning 8% of [15]. Of course, it should be noted that in the case of *Bell-7* and *Ray-17*, [15] and proposed methods are tied with the same DP number.

A final test set is the one that has been used to compare methods such as Masood [8], Ramer, Douglas and Peucker (RDP) [21,22] and Carmona-Poyato [10]. Choosing the results of these methods with the best ISE and FOM values, Table 4 presents the comparisons with our proposed method.

From these results, we can observe that, although most of Carmona's DP values are the best, the rest of the error criteria due to our method have the best performance except the ISE of *Maple leaf* and FOM of *Dog*, where the Masood method obtained better results. In Fig. 6 the comparison of two methods is shown: ours (red vertices) and Masood (green vertices), while the vertices that are in yellow, represent those that were found by both methods.

### 4.1. Effectiveness of each part of the proposed method

To verify the effectiveness of each part of our proposed method, the results of Brake points detection, Brake points detection+Break points elimination, Brake points detection+Break points elimination+Break points rearrangement are analyzed. To do this, we have chosen the above representative samples, from the sets of objects used to test our method. The results are shown in Table 5, in which can be observed that Brake points detection obtains a very small value of ISE regarding the tolerable one, T. In this step, the number of DP is the highest of the whole process. However, in Brake points detection+Break points

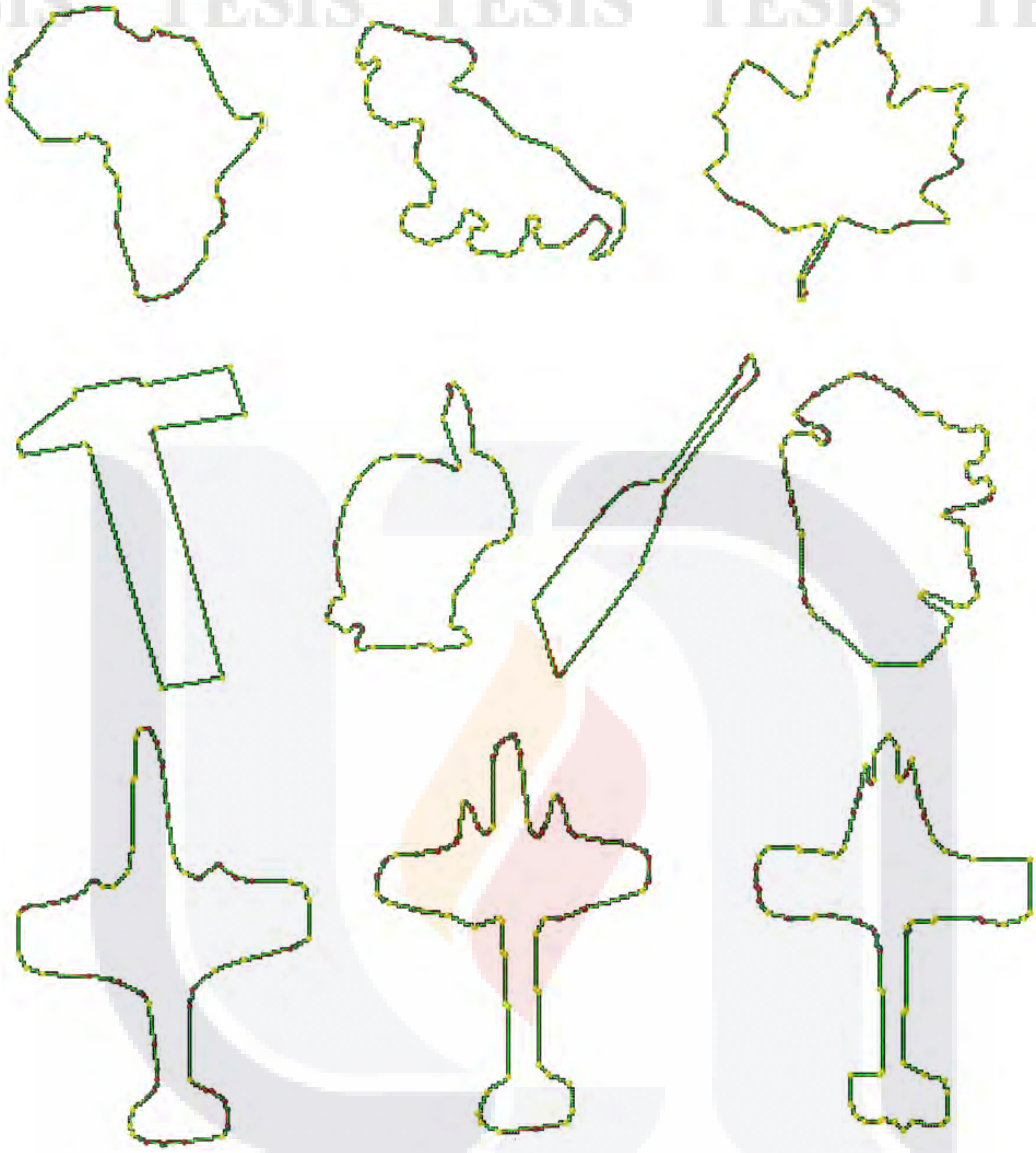


**Fig. 6.** Methods of comparison of the fourth set of shapes. The red points are obtained by our method, while the green ones are from Masood and the yellow points are due to both methods.

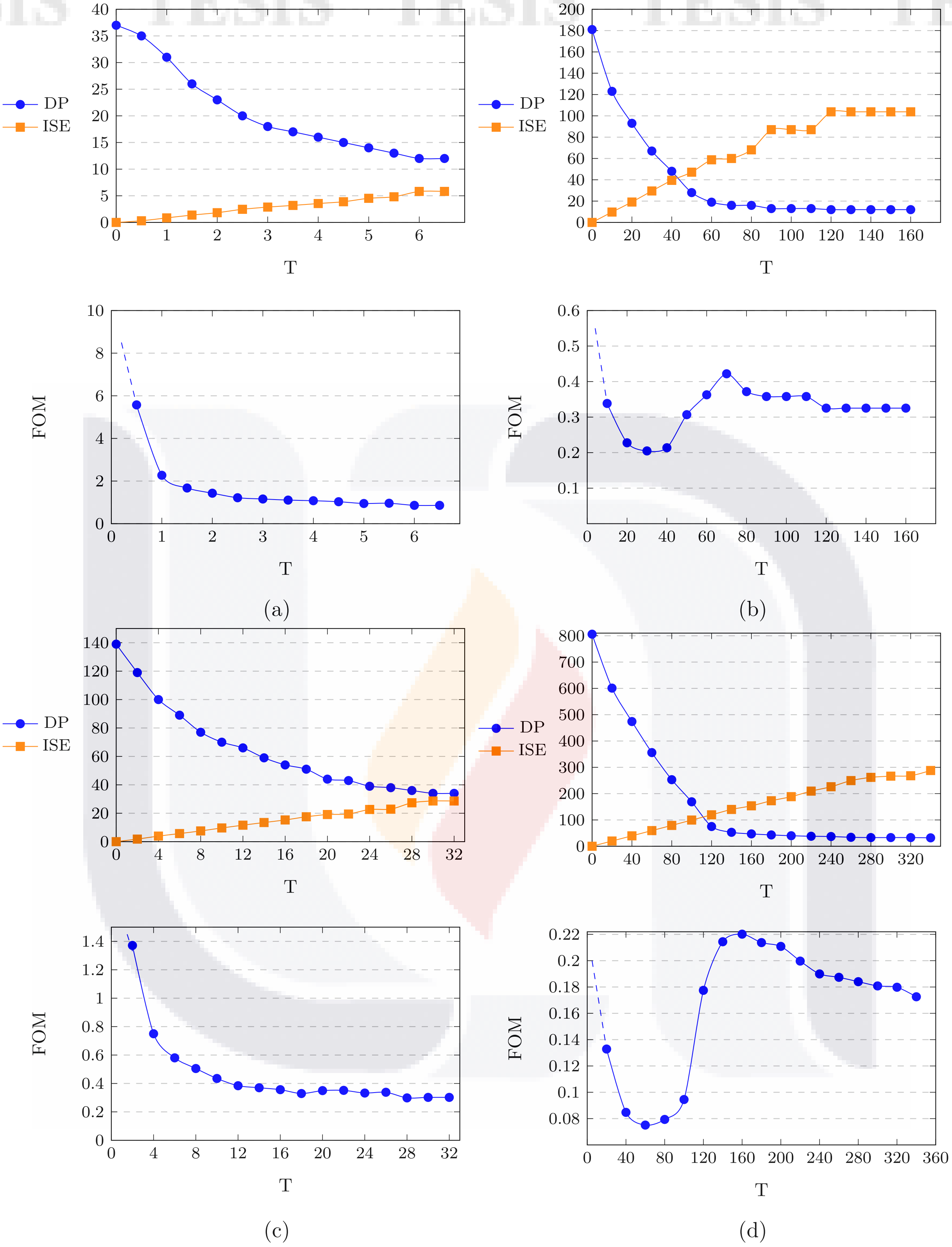
elimination the number of break points is reduced by more than half, considering, also, that ISE does not exceed T. This is achieved in all cases. Finally, we enter a cycle of elimination and rearrangement of break points, to reduce the number of them and maintain an ISE lower than T. It is worth mentioning that ISE can fluctuate increasing or decreasing in different iterations of the algorithm, but never exceeding T. The other error criteria, CR, FOM, WE, $WE_2$, depend on DP and ISE following their own definitions and the behavior given by the Eqs. (3), (6) and (7), respectively.

### 4.2. Error criteria tradeoff

Now, a question arises: How much can the number of DP be sacrificed with respect to the integral square error to have a good error criterion? To handle this tradeoff, Sarkar [3] introduced the Eq. (6). The calculation of the error criteria, ISE, DP and FOM shows interesting results in Fig. 7. On the one hand, it is observed that the FOM value is maximum (infinite) when the tolerable error is zero. This means that there is no loss of information when approximating the polygon to the original shape.

Although the main challenge is to reduce the ISE and the DP, over time there have been methods that, in order to decrease ISE, the number of DP has been sacrificed, thereby existing a tradeoff between them that can be reflected in other the error criteria. In Fig. 8 it can be appreciated that different authors have reduced ISE despite the increase in DP for the case of *Chromosome*.

Clearly, the fact of obtaining a set of DP that represents the shape of the object implies finding the least number of them without sacrificing too much the ISE. However, if we want to lose fewer pixels in the reconstruction of the shape, then we would have to spend more points, with the subsequent decrease of the ISE. It is possible to recover the original shape, without loss of pixels, even though the ISE is different from zero.



(a)

(b)

(c)

(d)

**Fig. 7.** Tradeoff between DP, ISE and FOM for (a) *Chromosome*, (b) *Cup*, (c) *Africa* and (d) *Device6-9* shapes, respectively.

In this work, we also consider the number of pixels that are lost (LP) when decoding is carried out to recover the shape. The reasons are as follows. Once the DP is found, if decoding is done, the lost pixels can be counted. The approximate polygon is obtained by considering the pixels that contain part of the continuous straight segments given by DP pairs. Starting with the first DP to decode, the 1-pixel in its 8-neighborhood for decoding is chosen when it contains the longest continuous straight line segment length. If the neighboring cell with the largest segment coincides with the 1-pixel of the original contour, then the pixel is not lost, otherwise it is a lost pixel.

Following this approach, a set of grammars that provide DP with the *central condition*, *i.e.* LP $= 0$, are proposed. We start with the simplest case, where a string of $p$ $a$'s represents a DSS of pixels with no change of direction. So, a type of DSS is recognized by the grammar $xa^p$, where $x \in \{a, b, c, d, e, f, g, h\}$. Other set of DSS is represented by the string $xbh$ (or $xhb$), concatenated by $p$ $a$'s to the left and $t$ $a$'s to the right. To maintain the central condition, the difference of $a$'s on both

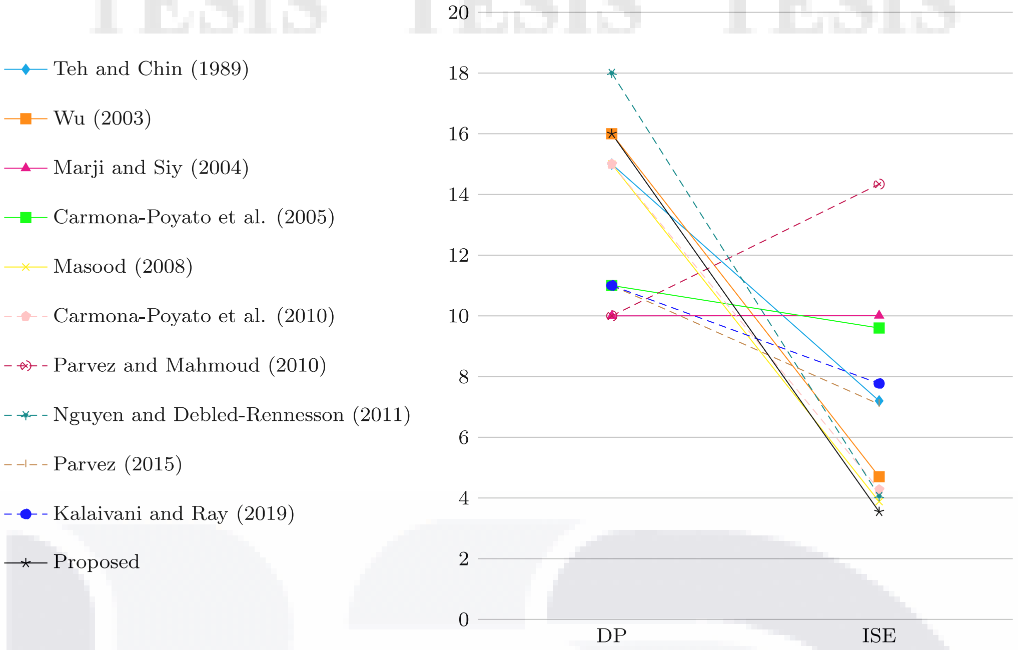


**Fig. 8.** Relationship between the DP and the ISE proposed by different authors.

**Fig. 9.** An approximate polygon for *Shark*. There are 73 DSS, such that LP $= 0$.

sides do not exceed more than one, *i.e.*, $|p - t| \leq 1$, and the string $xa^p bha^t$ is achieved. If we concatenate another substring $bha^q$ (or $hba^q$) to the right of $a^p$ it results in the substring $xa^p bha^q bha^t$. The amount of $a$'s between the two $bh$'s depends on $p$ and satisfies the inequality $2p \leq q \leq 2(p+1)$. Thus, we can obtain a second grammar that recognizes a DSS: $xa^p(bha^q)^r bha^t$, where $t \geq 0$, $2p \leq q < 2(p+1)$ and $|p - t| \leq 1$. As the reader can verify, this grammar has the characteristic that, although any amount ($r$ times) of $bha^q$ is concatenated, the central condition is satisfied.

Table 6 summarizes the context-free grammars found in the previous analysis, and shows some DSS examples, whereas Table 7 shows the parameter values used in the DSS of the examples given in Table 6. On the other hand, Table 8 shows the results of applying the grammars mentioned above to a set of shapes, where they are compared between the cases with loss and without loss. In the case of *Shark*, we found that there are 3.65 times more DP to obtain LP $= 0$ than the DP obtained when LP $\neq 0$. However, it represents 24.91% of its contour. *Hammer* the one that less number of DP uses, with 24.87%, and *Stingray* the one that more, with 45% of its contour.

As an example of the use of the grammars (1) and (2) of Table 6, in the case of *Shark*, each of the DSS that make up its contour is shown in Fig. 9 and displayed in detail in Table 9.

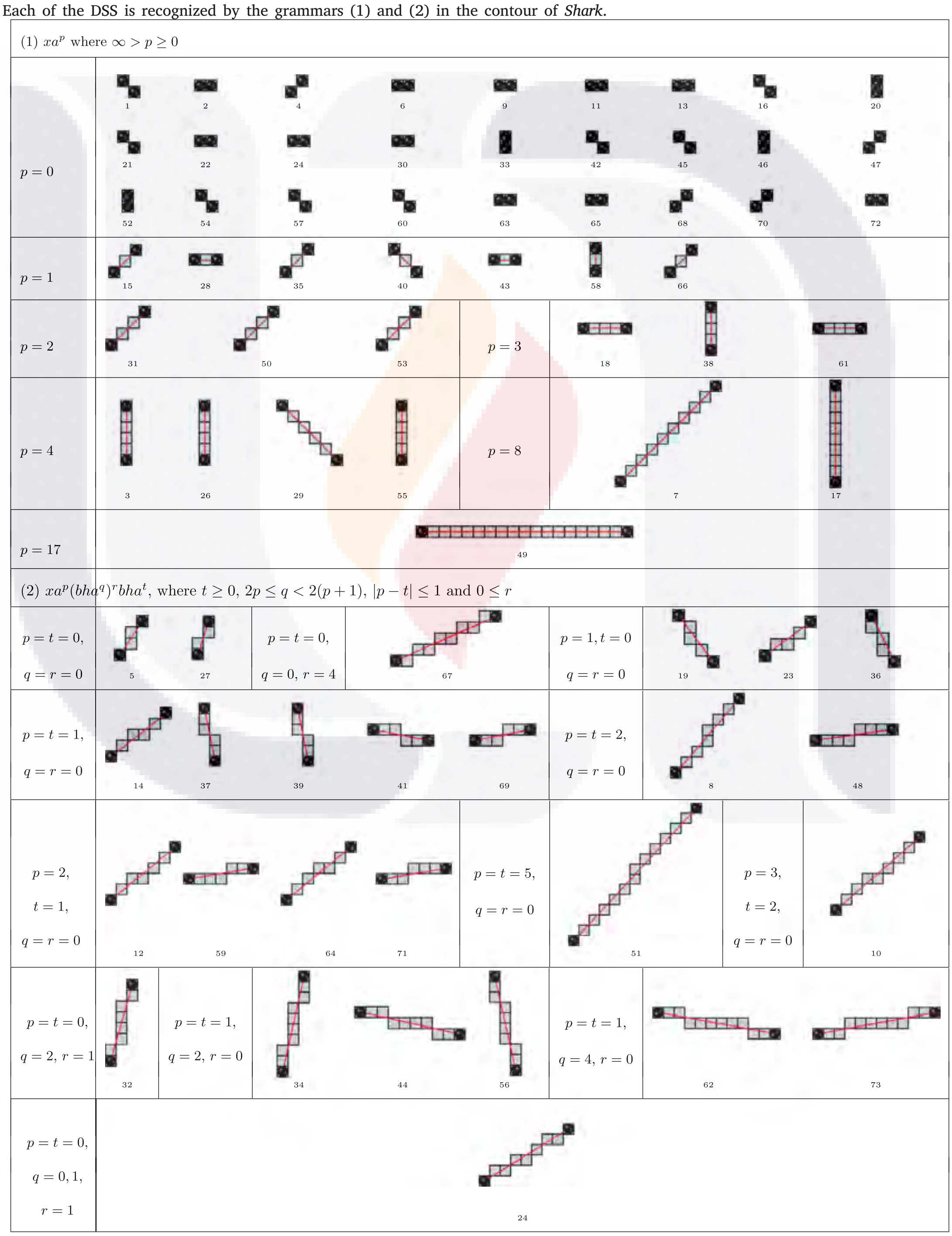
### 4.3. Complexity

The complexity of the proposed method can be obtained as follows: to obtain the break points it is necessary to visit the contour, which requires an order $O(n)$, while to calculate ISE the order is $O(m^2)$, where $m$ is the number of break points obtained. The previous two operations are performed iteratively by reducing $r$ by a half, which gives us a

**Table 8**
Comparison of error criteria with loss and without loss of pixels.

| | n | DP | ISE | FOM | | n | DP | ISE | FOM |
|---|---|---|---|---|---|---|---|---|---|
| *Shark* | | | | | *Africa* | | | | |
| LP $= 0$ | 293 | 73 | 11,871 | 0,338 | LP $= 0$ | 291 | 77 | 12,515 | 0,305 |
| LP $\neq 0$ | 293 | 20 | 66,910 | 0,219 | LP $\neq 0$ | 291 | 39 | 23,036 | 0,327 |
| *Cup* | | | | | *Hammer* | | | | |
| LP $= 0$ | 405 | 135 | 8,358 | 0,260 | LP $= 0$ | 388 | 97 | 24,486 | 0,164 |
| LP $\neq 0$ | 105 | 12 | 99,390 | 0,340 | LP $\neq 0$ | 388 | 15 | 37,657 | 0,690 |
| *Heart* | | | | | *Stingray* | | | | |
| LP $= 0$ | 303 | 134 | 4,625 | 0,489 | LP $= 0$ | 328 | 149 | 11,780 | 0,187 |
| LP $\neq 0$ | 303 | 20 | 63,364 | 0,239 | LP $\neq 0$ | 328 | 22 | 90,606 | 0,165 |

**Table 9**
Each of the DSS is recognized by the grammars (1) and (2) in the contour of *Shark*.



(1) $xa^p$ where $\infty > p \geq 0$

| | |
|---|---|
| $p = 0$ | 1, 2, 4, 6, 9, 11, 13, 16, 20<br>21, 22, 24, 30, 33, 42, 45, 46, 47<br>52, 54, 57, 60, 63, 65, 68, 70, 72 |
| $p = 1$ | 15, 28, 35, 40, 43, 58, 66 |
| $p = 2$ | 31, 50, 53 |
| $p = 3$ | 18, 38, 61 |
| $p = 4$ | 3, 26, 29, 55 |
| $p = 8$ | 7, 17 |
| $p = 17$ | 49 |

(2) $xa^p(bha^q)^r bha^t$, where $t \geq 0$, $2p \leq q < 2(p+1)$, $|p-t| \leq 1$ and $0 \leq r$

| | |
|---|---|
| $p = t = 0$, $q = r = 0$ | 5, 27 |
| $p = t = 0$, $q = 0$, $r = 4$ | 67 |
| $p = 1, t = 0$, $q = r = 0$ | 19, 23, 36 |
| $p = t = 1$, $q = r = 0$ | 14, 37, 39, 41, 69 |
| $p = t = 2$, $q = r = 0$ | 8, 48 |
| $p = 2$, $t = 1$, $q = r = 0$ | 12, 59, 64, 71 |
| $p = t = 5$, $q = r = 0$ | 51 |
| $p = 3$, $t = 2$, $q = r = 0$ | 10 |
| $p = t = 0$, $q = 2$, $r = 1$ | 32 |
| $p = t = 1$, $q = 2$, $r = 0$ | 34, 44, 56 |
| $p = t = 1$, $q = 4$, $r = 0$ | 62, 73 |
| $p = t = 0$, $q = 0, 1$, $r = 1$ | 24 |

$O(log_2(r))$ order. So, to obtain the break points the method requires $O(log_2(r))(O(n)+O(m^2))$.

On the other hand, the elimination process is made up of two independent threads. The first searches for the $\mathrm{ISE}_{min}$ (it takes $O(m)$) and in the second a break point is removed if $\mathrm{ISE}_{min}$ plus the rest of ISEs are equal to or less than T (what takes $O(m)$), so a time of $O(2m)$ for elimination process is required.

Finally, as we know, when using Dijkstra's algorithm to calculate the minimum distance, in the rearranging process, the complexity for the worst case is $O(|V|^2)$, where $V$ is the number of vertices that contains the graph that represents the problem to be treated. In our problem, the number of vertices is given by considering that we have $5(m-1)$, plus two extra vertices: the beginning and the end of a graph. Since there are five graphs, we have a total of $5(5(m-1)+2)$ vertices. Let us recall that we are using a radius of 2 (five break points) of the vicinity of each break point. So, $O([5(5(m-1)+2)]^2)$. Again, computing ISE on break points is $O(m^2)$. This process is achieved for each break point, so, the total complexity is $O(m)[O(2m)+O[5(5(m-1)+2)]^2]+O(m^2)$. Therefore, the total complexity of our method is at most $O(m^3)$.

We ran our method on an Intel (R) Core (TM), i7-8706G computer, at 3.10 GHz, 3.10 GHz CPU and 16 GB RAM. On average, the time used in our tests was 217.61 ms.

## 5. Conclusions and further work

We have presented a polygonal approximation method that improves the error criteria commonly used in the literature. Our method is based primarily on the search for a set of break points, which are obtained from a context-free grammar, to subsequently reduce the quadratic integral error through the imposition of a tolerable error. Then, we perform an elimination and rearrangement of break points optimally, until the final dominant points are obtained. The complexity of our method does not exceed that of optimal algorithms like those of Masood. On the other hand, we have addressed the challenge of obtaining polygonal approximation without loss of information, by searching different context-free grammars, such that they recognize digital straight segments without loss of pixels. For further work, a theoretical development could be carried out to find the minimum number of digital straight segments, so that there are no lost pixels.

The problem of DP detection and error criteria tradeoff optimization using deep learning and graph-based methods should be investigated. A future work to compare our method with deep learning optimization methods is recommended, such as CNN or graph-based theory with deep learning, which warrants further extensive research.

## CRediT authorship contribution statement

**Osvaldo A. Tapia-Dueñas:** Conceptualization, Design of study, Acquisition of data, Analysis, Interpretation of data, Writing - original draft, Writing - review & editing. **Hermilo Sánchez-Cruz:** Conceptualization, Design of study, Acquisition of data, Analysis, Interpretation of data, Writing - original draft, Writing - review & editing.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgment

H. Sánchez-Cruz was supported by Universidad Autónoma de Aguascalientes, México, under grant PIINF20-6.

All authors approved the version of the manuscript to be published.

## References

[1] P. Sankar, C. Sharma, A parallel procedure for the detection of dominant points on a digital curve, Comput. Graph. Image Process. 7 (3) (1978) 403–412, http://dx.doi.org/10.1016/S0146-664X(78)80006-9.

[2] C. Teh, R.T. Chin, On the detection of dominant points on digital curves, IEEE Trans. Pattern Anal. Mach. Intell. 11 (8) (1989) 859–872.

[3] D. Sarkar, A simple algorithm for detection of significant vertices for polygonal approximation of chain-coded curves, Pattern Recognit. Lett. 14 (12) (1993) 959–964, http://dx.doi.org/10.1016/0167-8655(93)90004-W.

[4] T.M. Cronin, A boundary concavity code to support dominant point detection, Pattern Recognit. Lett. 20 (6) (1999) 617–634, http://dx.doi.org/10.1016/S0167-8655(99)00025-2.

[5] B. Sarkar, L.K. Singh, D. Sarkar, Approximation of digital curves with line segments and circular arcs using genetic algorithms, Pattern Recognit. Lett. 24 (15) (2003) 2585–2595, http://dx.doi.org/10.1016/S0167-8655(03)00103-X.

[6] F. Arrebola, F. Sandoval, Corner detection and curve segmentation by multiresolution chain-code linking, Pattern Recognit. 38 (10) (2005) 1596–1614, http://dx.doi.org/10.1016/j.patcog.2005.03.006.

[7] A. Masood, S.A. Haq, A novel approach to polygonal approximation of digital curves, J. Vis. Commun. Image Represent. 18 (3) (2007) 264–274, http://dx.doi.org/10.1016/j.jvcir.2006.12.002.

[8] A. Masood, Optimized polygonal approximation by dominant point deletion, Pattern Recognit. 41 (1) (2008) 227–239, http://dx.doi.org/10.1016/j.patcog.2007.05.021.

[9] A. Masood, Dominant point detection by reverse polygonization of digital curves, Image Vis. Comput. 26 (5) (2008) 702–715, http://dx.doi.org/10.1016/j.imavis.2007.08.006.

[10] A. Carmona-Poyato, F. Madrid-Cuevas, R. Medina-Carnicer, R.M. noz Salinas, Polygonal approximation of digital planar curves through break point suppression, Pattern Recognit. 43 (1) (2010) 14–25, http://dx.doi.org/10.1016/j.patcog.2009.06.010.

[11] F. Madrid-Cuevas, E. Aguilera-Aguilera, A. Carmona-Poyato, R. Muñoz-Salinas, R. Medina-Carnicer, N. Fernández-García, An efficient unsupervised method for obtaining polygonal approximations of closed digital planar curves, J. Vis. Commun. Image Represent. 39 (2016) 152–163, http://dx.doi.org/10.1016/j.jvcir.2016.05.021.

[12] H. Sánchez-Cruz, A proposal method for corner detection with an orthogonal three-direction chain code, in: J. Blanc-Talon, W. Philips, D. Popescu, P. Scheunders (Eds.), Advanced Concepts for Intelligent Vision Systems. ACIVS 2006. Lecture Notes in Computer Science, Vol. 4179, Springer-Verlag, Berlin, Heidelberg, 2006, pp. 161–172, http://dx.doi.org/10.1007/11864349_15.

[13] T.P. Nguyen, I. Debled-Rennesson, A discrete geometry approach for dominant point detection, Pattern Recognit. 44 (1) (2011) 32–44, http://dx.doi.org/10.1016/j.patcog.2010.06.022.

[14] N. Fernández-García, L.D.-M. Martínez, A. Carmona-Poyato, F. Madrid-Cuevas, R. Medina-Carnicer, A new thresholding approach for automatic generation of polygonal approximations, J. Vis. Commun. Image Represent. 35 (2016) 155–168, http://dx.doi.org/10.1016/j.jvcir.2015.12.013.

[15] S. Kalaivani, B.K. Ray, A heuristic method for initial dominant point detection for polygonal approximations, Soft Comput. 23 (18) (2019) 8435–8452, http://dx.doi.org/10.1007/s00500-019-03936-1.

[16] A. Fred, Informational aspects of visual perception, Psychol. Rev. 61 (1954) 183–193, http://dx.doi.org/10.1037/h0054663.

[17] M. Marji, P. Siy, A new algorithm for dominant points detection and polygonization of digital curves, Pattern Recognit. 36 (10) (2003) 2239–2251, http://dx.doi.org/10.1016/S0031-3203(03)00119-5.

[18] A. Garrido, N.P. de la blanca, M. Garcia-Silvente, Boundary simplification using a multiscale dominant-point detection algorithm, Pattern Recognit. 31 (6) (1998) 791–804, http://dx.doi.org/10.1016/S0031-3203(97)00104-0.

[19] M.T. Parvez, Optimized polygonal approximations through vertex relocations in contour neighborhoods, Image Vis. Comput. 34 (2015) 1–10, http://dx.doi.org/10.1016/j.imavis.2014.10.012.

[20] D.K. Prasad, M.K. Leung, C. Quek, S.-Y. Cho, A novel framework for making dominant point detection methods non-parametric, Image Vis. Comput. 30 (11) (2012) 843–859, http://dx.doi.org/10.1016/j.imavis.2012.06.010.

[21] U. Ramer, An iterative procedure for the polygonal approximation of plane curves, Comput. Graph. Image Process. 1 (3) (1972) 244–256, http://dx.doi.org/10.1016/S0146-664X(72)80017-0.

[22] D.H. Douglas, T.K. Peucker, Algorithms for the reduction of the number of points required to represent a digitized line or its caricature, 1973.

[23] N.L. Fernández García, L.D.-M. Martínez, Ángel Carmona Poyato, F.J. Madrid Cuevas, R.M. Carnicer, Unsupervised generation of polygonal approximations based on the convex hull, Pattern Recognit. Lett. 135 (2020) 138–145, http://dx.doi.org/10.1016/j.patrec.2020.04.014.

[24] E. Aguilera-Aguilera, A. Carmona-Poyato, F. Madrid-Cuevas, M. Marín-Jiménez, Fast computation of optimal polygonal approximations of digital planar closed curves, Graph. Models 84 (2016) 15–27, http://dx.doi.org/10.1016/j.gmod.2016.01.004.

[25] M. Marji, P. Siy, Polygonal representation of digital planar curves through dominant point detection—a nonparametric algorithm, Pattern Recognit. 37 (11) (2004) 2113–2130, http://dx.doi.org/10.1016/j.patcog.2004.03.004.

[26] A. Carmona-Poyato, R. Medina-Carnicer, F. Madrid-Cuevas, R. Muñoz-Salinas, N. Fernández-García, A new measurement for assessing polygonal approximation of curves, Pattern Recognit. 44 (1) (2011) 45–54, http://dx.doi.org/10.1016/j.patcog.2010.07.029.

[27] S. Saha, S. Bhunia, L. Nayak, R. Bhattacharyya, P.R. Sinha Mahapatra, A multi-level polygonal approximation-based shape encoding framework for automated shape retrieval, 2020, pp. 203–215, http://dx.doi.org/10.1007/978-981-13-7403-6_20.

[28] P. Ngo, A discrete approach for polygonal approximation of irregular noise contours, in: CAIP 2019 - Computer Analysis of Images and Patterns, in: LNCS, vol. 11678, Salerno, Italy, 2019, pp. 433–446, http://dx.doi.org/10.1007/978-3-030-29888-3_35.

[29] K. Zhao, S. Li, Z. Kang, Smooth trajectory generation for linear paths with optimal polygonal approximation, ISBN: 978-981-32-9681-7, 2020, pp. 457–465, http://dx.doi.org/10.1007/978-981-32-9682-4_48.

[30] Z. Wu, C. Zhao, B. Liu, Polygonal approximation based on coarse-grained parallel genetic algorithm, J. Vis. Commun. Image Represent. 71 (2020) 102717, http://dx.doi.org/10.1016/j.jvcir.2019.102717.

[31] P.-Y. Yin, Ant colony search algorithms for optimal polygonal approximation of plane curves, Pattern Recognit. 36 (8) (2003) 1783–1797, http://dx.doi.org/10.1016/S0031-3203(02)00321-7.

[32] S. Huang, C. Wang, Genetic algorithms for approximation of digital curves with line segments and circular arcs, J. Chin. Inst. Eng. 32 (4) (2009) 437–444, http://dx.doi.org/10.1080/02533839.2009.9671526, arXiv:10.1080/02533839.2009.9671526.

[33] D. Chetverikov, Z. Szabó, A simple and efficient algorithm for detection of high curvature points in planar curves, in: N. Petkov, M.A. Westenberg (Eds.), Computer Analysis of Images and Patterns, Springer Berlin Heidelberg, Berlin, Heidelberg, 2003, pp. 746–753.

[34] N. Nain, V. Laxmi, B. Bhadviya, A. Gopal, Corner detection using difference chain code as curvature, in: 2007 Third International IEEE Conference on Signal-Image Technologies and Internet-Based System, 2007, pp. 821–825, http://dx.doi.org/10.1109/SITIS.2007.118.

[35] L.J. Latecki, R. Lakämper, Convexity rule for shape decomposition based on discrete contour evolution, Comput. Vis. Image Underst. 73 (3) (1999) 441–454, http://dx.doi.org/10.1006/cviu.1998.0738.

[36] W.-Z. Shao, Z.-H. Wei, Edge-and-corner preserving regularization for image interpolation and reconstruction, Image Vis. Comput. 26 (12) (2008) 1591–1606, http://dx.doi.org/10.1016/j.imavis.2008.03.002.

[37] F. Mokhtarian, R. Suomela, Robust image corner detection through curvature scale space, IEEE Trans. Pattern Anal. Mach. Intell. 20 (12) (1998) 1376–1381, http://dx.doi.org/10.1109/34.735812.

[38] D.K. Prasad, C. Quek, M.K.H. Leung, S. Cho, A parameter independent line fitting method, in: The First Asian Conference on Pattern Recognition, 2011, pp. 441–445, http://dx.doi.org/10.1109/ACPR.2011.6166585.

[39] Y. Li, S. Wang, Q. Tian, X. Ding, A survey of recent advances in visual feature detection, Neurocomputing 149 (2015) 736–751, http://dx.doi.org/10.1016/j.neucom.2014.08.003.

[40] F. Chabat, G. Yang, D. Hansell, A corner orientation detector, Image Vis. Comput. 17 (10) (1999) 761–769, http://dx.doi.org/10.1016/S0262-8856(98)00150-4.

[41] Z. Liu, J. Watson, A. Allen, A polygonal approximation of shape boundaries of marine plankton based-on genetic algorithms, J. Vis. Commun. Image Represent. 41 (2016) 305–313, http://dx.doi.org/10.1016/j.jvcir.2016.10.010.

[42] S. Singh, A. Sharma, I. Chhabra, A dominant points-based feature extraction approach to recognize online handwritten strokes, Int. J. Doc. Anal. Recognit. (IJDAR) 20 (1) (2017) 37–58, http://dx.doi.org/10.1007/s10032-016-0279-x.

[43] F. Meng, H. Li, Q. Wu, B. Luo, C. Huang, K.N. Ngan, Globally measuring the similarity of superpixels by binary edge maps for superpixel clustering, IEEE Trans. Circuits Syst. Video Technol. 28 (4) (2018) 906–919.

[44] F. Meng, H. Li, S. Zhu, B. Luo, C. Huang, B. Zeng, M. Gabbouj, Constrained directed graph clustering and segmentation propagation for multiple foregrounds cosegmentation, IEEE Trans. Circuits Syst. Video Technol. 25 (11) (2015) 1735–1748.

[45] A. Gordo, J. Almazán, J. Revaud, D. Larlus, Deep image retrieval: Learning global representations for image search, in: B. Leibe, J. Matas, N. Sebe, M. Welling (Eds.), Computer Vision – ECCV 2016, Springer International Publishing, Cham, 2016, pp. 241–257.

[46] A.S. Razavian, J. Sullivan, S. Carlsson, A. Maki, [Paper] visual instance retrieval with deep convolutional networks, ITE Trans. Media Technol. Appl. 4 (3) (2016) 251–258, http://dx.doi.org/10.3169/mta.4.251.

[47] W. Xu, H. Wang, F. Qi, C. Lu, Explicit shape encoding for real-time instance segmentation, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 5167–5176.

[48] X. Xu, F. Meng, H. Li, Q. Wu, Y. Yang, S. Chen, Bounding box based annotation generation for semantic segmentation by boundary detection, in: 2019 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), 2019, pp. 1–2.

[49] E. Xie, P. Sun, X. Song, W. Wang, D. Liang, C. Shen, P. Luo, Polarmask: Single shot instance segmentation with polar representation, 2020, arXiv URL https://arxiv.org/abs/1909.13226v4.

[50] F. Meng, H. Li, Q. Wu, K.N. Ngan, J. Cai, Seeds-based part segmentation by seeds propagation and region convexity decomposition, IEEE Trans. Multimed. 20 (2) (2018) 310–322, http://dx.doi.org/10.1109/TMM.2017.2739919.

[51] H. Ling, J. Gao, A. Kar, W. Chen, S. Fidler, Fast interactive object annotation with curve-GCN, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 5252–5261, http://dx.doi.org/10.1109/CVPR.2019.00540.

[52] M. Gori, G. Monfardini, F. Scarselli, A new model for learning in graph domains, in: Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005., Vol. 2, 2005, pp. 729–734, http://dx.doi.org/10.1109/IJCNN.2005.1555942.

[53] M. Henaff, J. Bruna, Y. LeCun, Deep convolutional networks on graph-structured data, 2015, CoRR abs/1506.05163. arXiv:1506.05163.

[54] M. Defferrard, X. Bresson, P. Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, in: D.D. Lee, M. Sugiyama, U.V. Luxburg, I. Guyon, R. Garnett (Eds.), Advances in Neural Information Processing Systems 29, Curran Associates, Inc., 2016, pp. 3844–3852, URL http://papers.nips.cc/paper/6081-convolutional-neural-networks-on-graphs-with-fast-localized-spectral-filtering.pdf.

[55] D. Bacciu, F. Errica, A. Micheli, M. Podda, A gentle introduction to deep learning for graphs, Neural Netw. 129 (2020) 203–221, http://dx.doi.org/10.1016/j.neunet.2020.06.006.

[56] H. Sánchez-Cruz, O.A. Tapia-Dueñas, F. Cuevas, Polygonal approximation using a multiresolution method and a context-free grammar, in: J.A. Carrasco-Ochoa, J.F. Martínez-Trinidad, J.A. Olvera-López, J. Salas (Eds.), Pattern Recognition, Springer International Publishing, Cham, 2019, pp. 261–270, http://dx.doi.org/10.1007/978-3-030-21077-9_24.

[57] E.W. Dijkstra, A note on two problems in connexion with graphs, Numer. Math. 1 (1) (1959) 269–271, http://dx.doi.org/10.1007/BF01386390.

[58] W.-Y. Wu, An adaptive method for detecting dominant points, Pattern Recognit. 36 (10) (2003) 2231–2237, http://dx.doi.org/10.1016/S0031-3203(03)00087-6.

[59] Y.K. Liu, B. Žalik, An efficient chain code with huffman coding, Pattern Recognit. 38 (4) (2005) 553–557, http://dx.doi.org/10.1016/j.patcog.2004.08.017.

[60] H. Nasser, P. Ngo, I. Debled-Rennesson, Dominant point detection based on discrete curve structure and applications, J. Comput. System Sci. 95 (2018) 177–192, http://dx.doi.org/10.1016/j.jcss.2017.07.007.

# 6 Simplificación de objetos 3D usando nubes de puntos basadas en código de cadena

A continuación se presenta el artículo publicado, cuyo título es: **3D object simplification usin chain code-based point clouds**

Multimedia Tools and Applications (2023) 82:9491–9515
https://doi.org/10.1007/s11042-022-13588-3

# 3D object simplification using chain code-based point clouds

**Osvaldo A. Tapia-Dueñas[1] · Hermilo Sánchez-Cruz[1] · Hiram H. López[2]**

Received: 1 December 2021 / Revised: 25 February 2022 / Accepted: 18 July 2022 /

## Abstract

This work aims to obtain a sequence of 3D point clouds associated with a 3D object that reduces the volume data and preserves the shape of the original object. The sequence contains point clouds that give different simplifications of the object, from a very fine-tuned representation to a simple and sparse one. Such a sequence is important because it satisfies different needs, from a faithful representation with a low reduction of points to a significant data reduction that only preserves the main properties of the object. We construct the sequence in the following way. We first obtain a voxelization of the original 3D object. Then, we organize the voxels by slices to get a single chain code that represents the original 3D object. The point clouds depend on the key points of the chain code. The Hausdorff distance and the average geometric error prove that the point clouds are invariant under rigid rotations and maintain the shape of the object. Our results indicate that the proposed method has an average efficiency of 60% regarding the state-of-the-art simplification methods.

**Keywords** Chain code · Voxelization · 3D shape · Data reduction · Key points

## 1 Introduction

During the last years, numerous works in the literature have studied the representation of a three-dimensional (3D) object. One of the main reasons is the growing demand for applications related to these objects, such as digitization, storage, and recognition. We focus on the representations given in terms of point clouds or voxels.

---

✉ Hermilo Sánchez-Cruz
hermilo.sanchez@edu.uaa.mx

Osvaldo A. Tapia-Dueñas
osvaldo.tapia@edu.uaa.mx

Hiram H. López
h.lopezvaldez@csuohio.edu

[1] Departamento de Ciencias de la Computación, Universidad Autónoma de Aguascalientes, Aguascalientes, 20100, Aguascalientes, México

[2] Department of Mathematics and Statistics, Cleveland State University, Cleveland, 44115, OH, USA

The *point clouds representation* consists of a sample of points from the 3D object. An appropriate selection of points represents the shape of the object as close as desired. The use of key points to capture the essence of the object is used in [14, 21, 27, 38, 39, 56]. Numerous applications that demand the best representations of a 3D object or a scene, such as animation [11, 26, 50], video [8, 22, 51], or CAD systems [20, 33, 54], employ point clouds. The massive amount of data that all these applications require has led to significant research in the area of point cloud compression. In [1] is presented an excellent survey. One of the main goals of the compression is to store and transmit efficiently the data that is needed to represent the 3D object [1, 10, 13, 40]. In [18] is introduced a simplification algorithm based on detailed feature points. In [40] is given an algorithm based on the geometric characteristics of the point clouds data set. In [10] is offered a compression method for the geometric information of voxel point clouds. In [16] is proposed a compression method based on deep learning. In [13] is introduced an efficient compression scheme for the attributes of voxel 3D point clouds. In [17] is given an edge-preserved point cloud simplification algorithm based on a normal vector.

A *voxel representation* is another common way to depict a 3D object [23, 45, 47]. Some of the main applications are morphometric studies in MRI [12], place recognition [41], 3D building detection for airborne LIDAR point clouds [52], and crankshaft mass balancing [57].

A *chain code* is a common and compact way to represent the contour shape of a two-dimensional (2D) object. The Freeman chain code, proposed by Freeman in 1961 [9], is a sequence of symbols that belong to the set $\{0, 1, 2, 3, 4, 5, 6, 7\}$. The angle Freeman chain code, introduced by Kui and Žalik in 2005 [30], is a sequence of symbols that belong to the set $\{a, b, c, d, e, f, g, h\}$ and codifies the angles of the contour shape of the 2D object. In [37] is shown that certain string of symbols in the angle Freeman chain code represent digital straight segments of the contour shape of the 2D object. Chain coding takes advantage of symbolic representation and obtains patterns in digital straight segments, which give us knowledge for key point detection [55].

This work constructs a sequence of point clouds that reliably describes and simplifies a 3D object. The sequence contains point clouds that give different object descriptions, from a straightforward and sparse one to a very fine-tuned representation. The main steps are the following.

1. Given a 3D object, we first apply a voxelization method.
2. We obtain a chain code that represents the whole 3D object.
3. Fixing non-negative integers $p$, $q$, and $r$, and a positive integer $\delta$, we use the chain code to obtain a point cloud that depends on the digital straight segments of the 3D object.

It is important to remark the following about a voxelization of a 3D object. More voxels will give a better representation, and fewer will result in a worse model. Thus, as we explain below, the lower the parameters $p, q, r$, and $\delta$, the better the resolution of the associated point cloud. The greater the values, the poorer the representation. Consequently, if we want a dense point cloud representation of the 3D object, we need an excellent voxelization and integers $p, q, r$, and $\delta$ close to 1. We require a simpler voxelization and large integers $p, q, r$, and $\delta$ to save data and processing time. This decision would depend on the needs of the user.

### 1.1 Previous work on point cloud sequence

In a dynamic point cloud sequence [29, 49] and a point cloud video [7, 28], the point clouds can change in position and characteristics along the time. These point clouds can represent a 3D object with modified shapes in different frames. They also arise in robotic motion, where point clouds change their distribution in terms of time [3, 24]. In our work, we use a *sequence of point clouds* (or *point cloud sequence*) as a collection of different point clouds representing the same 3D object. Our point clouds have no modification in time (no dynamic) but in density.

Regarding the area of point cloud sequence, in [19] is proposed a balance between preserving sharp characteristics and maintaining a uniform density on the point clouds. In [5] is presented a resampling framework to select the point clouds. An essential application of the last method is to extract application-dependent features and reduce post-computation on large-scale point clouds. Leal et al. [25] presented a new approach for point cloud simplification based on dictionary learning and sparse coding. Pauly et al. [36] presented two algorithms: uniform incremental clustering and hierarchical clustering for surface simplification of geometric models represented by unstructured point clouds. In [44] is proposed a point cloud simplification method aiming to preserve edges in the data. The method detected points on the edges and applied the progressive simplification of points in smooth regions. Shoaib et al. [32] proposed a fractal bubble method to simplify 3D data points. The process selects important data points caused by expanding a recursive generation of self-similar 2D bubbles until contacting any point. Ning et al. [34] proposed a method to describe the semantic form of different objects in point clouds based on an optimized shape semantic graph. Song et al. [42] proposed a simplification algorithm for point cloud data from mechanical models, which identifies and preserves the edges by progressively removing non-edge in these regions. Song et al. [43] proposed a simplification approach by dividing the original data set into groups by representative points. El Sayed et al. [6] proposed a simplification method for 3D point clouds using weighted graphs representation that optimized the point clouds and preserved the features of the initial data. Chen et al. [4] employed a resampling framework based on a graph that selects representative points while preserving application-dependent features. Huang et al. [15] proposed a correction over the original LOP algorithm, incorporating locally adaptive density weights into LOP. Wei et al. [53] proposed a point clouds fast thinning algorithm using the method of spatial index judgment.

The contents of this work are the following. In Section 2, we develop a chain code associated with a 3D object. The chain code depends on the digital straight segments of the slices of the object. In Section 3, we take advantage of the properties of the chain code to define the sequence of cloud points in terms of the key points, which are the endpoints of the digital straight segments of the object. In Section 5, we use a sample of 3D objects to study the behavior of the sequence. We prove that the sequence is invariant under rotation using the Hausdorff distance and the average geometric error. In addition, we show that the point clouds visually describe the object. In Section 6, we present conclusions and further work.

## 2 A chain code

Let $\mathcal{D}$ be a 3D connected object. The first step is to use the Binvox software [31, 35] to obtain a voxelization of $\mathcal{D}$. From now on, we assume that $\mathcal{D}$ is a voxelized 3D connected object unless stated otherwise. In this section, we define a chain code that represents the surface of $\mathcal{D}$. This chain code is an essential tool to develop the sequence of point clouds that we associate to $\mathcal{D}$.

Let $\mathcal{B}$ be a 2D connected object. A *chain code* is a common and compact way to represent the object $\mathcal{B}$. The *Freeman chain* ($F8$) *code* of $\mathcal{B}$, proposed by Freeman in 1961 [9] and denoted by $F8\left(\mathcal{B}\right)$, is a sequence of symbols that belong to the set $\{0, 1, 2, 3, 4, 5, 6, 7\}$ and codifies two adjacent pixels of the contour shape of $\mathcal{B}$.

The *angle Freeman chain* ($AF8$) *code* of $\mathcal{B}$ is based on the $F8$ code. The $AF8$ code, proposed by Kui and Žalik in 2005 [30] and denoted by $AF8\left(\mathcal{B}\right)$, is a sequence of symbols that belong to the set $\{a, b, c, d, e, f, g, h\}$. Every symbol different from $e$ codifies the angle between three adjacent pixels. The symbol $e$ appears when a pixel in the contour shape of $\mathcal{B}$ is adjacent to only one pixel of the contour shape.

Note that $\mathcal{D}$ is a union of $n \geq 1$ $z$-slices: $\mathcal{D} = \bigcup_{i=1}^{n} \mathcal{D}_i$. Each $z$-slice $\mathcal{D}_i$ is a 2D object, which may not be connected. Consider that every 2D object $\mathcal{D}_i$ is the union of $m_i \in \mathbb{Z}_{\geq 0}$ connected components: $\mathcal{D}_i = \bigcup_{j=1}^{m_i} \mathcal{D}_{ij}$. We now assume that the connected components $\mathcal{D}_{i1}, \mathcal{D}_{i2}, \ldots, \mathcal{D}_{im_i}$ in every slice $\mathcal{D}_i$ have been ordered from the closest to the furthest respect to the origin of the space. Every connected component $\mathcal{D}_{ij}$ is also a 2D object. See Figs. 1 and 2(a). As a consequence, it is possible to compute $AF8\left(\mathcal{D}_{ij}\right)$. See Fig. 2(b).

We now define the chain code of $\mathcal{D}$, which is a required tool to determine the sequence of point clouds associated with $\mathcal{D}$. The chain code of $\mathcal{D}$ is obtained by concatenating the chain codes $AF8\left(\mathcal{D}_{ij}\right)$ and a few more symbols.

**Definition 1** Let $\mathcal{D}$ be a 3D object. The *chain code* of $\mathcal{D}$ is denoted and defined by

$$CC\left(\mathcal{D}\right) = y_1 AF8\left(\mathcal{D}_{11}\right) \ldots AF8\left(\mathcal{D}_{1m_1}\right) \ldots y_n AF8\left(\mathcal{D}_{n1}\right) \ldots AF8\left(\mathcal{D}_{nm_n}\right),$$

where $y_i$ is a symbol that helps identify the chain code associated with the $i$-th $z$-slice.

*Example 2* Figure 1 shows a 3D object $\mathcal{D}$. Observe that $\mathcal{D}$ has the following nine connected components: $\mathcal{D}_{11}, \mathcal{D}_{21}, \mathcal{D}_{22}, \mathcal{D}_{31}, \mathcal{D}_{32}, \mathcal{D}_{41}, \mathcal{D}_{42}, \mathcal{D}_{51}$ and $\mathcal{D}_{61}$. We first find $AF8\left(\mathcal{D}_{11}\right), \ldots, AF8\left(\mathcal{D}_{61}\right)$. Then, we compute the chain code $CC(\mathcal{D})$:

$$CC\left(\mathcal{D}\right) = y_1 AF8\left(\mathcal{D}_{11}\right) y_2 AF8\left(\mathcal{D}_{21}\right) AF8\left(\mathcal{D}_{22}\right) \cdots y_5 AF8\left(\mathcal{D}_{51}\right) y_6 AF8\left(\mathcal{D}_{61}\right).$$

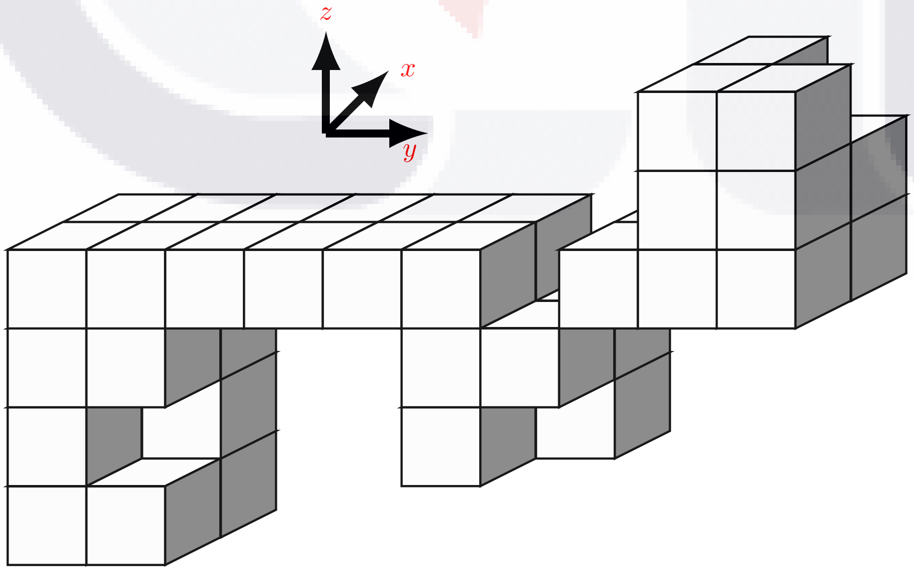


**Fig. 1** A 3D object $\mathcal{D}$ with nine connected components. We need to find first $AF8\left(\mathcal{D}_{11}\right), \ldots, AF8\left(\mathcal{D}_{61}\right)$ to compute the chain code $CC\left(\mathcal{D}\right)$

## 3 Point clouds

One of the main outputs of this work is to associate a sequence of point clouds with the 3D connected object $\mathcal{D}$ that preserves the original shape of the object. In this section, we describe the process of finding these point clouds.

Recall that $\mathcal{D}$ denotes a voxel 3D connected object that is the union of $n \geq 1$ $z$-slices: $\mathcal{D} = \bigcup_{i=1}^{n} \mathcal{D}_i$. Each $z$-slice $\mathcal{D}_i$ is a 2D object, which may be not connected, and $\mathcal{D}_i$ is the union of $m_i \in \mathbb{Z}_{\geq 0}$ connected components: $\mathcal{D}_i = \bigcup_{j=1}^{m_i} \mathcal{D}_{ij}$. The point clouds that we associate to $\mathcal{D}$ are obtained from the chain code $CC(\mathcal{D})$. The process is the following.

Let $\mathcal{B}$ be a two-dimensional object. The chain code $AF8\,(\mathcal{B})$ contains the information of the digital straight segments of the contour shape [37, 48]. For this reason, we have the following definition.

**Definition 3** A *digital straight segment* of the contour shape of the 2D object $\mathcal{B}$ is given by a substring in $AF8\,(\mathcal{B})$ of the following form:

$$Xa^{p}\left(Ya^{q}\right)^{r},$$

where $X \in \{a, b, c, d, e, f, g, h\}$, $Y \in \{bh, hb\}$, and $p$, $q$, and $r$ are non-negative integers representing the number of times a symbol is concatenated.

Every symbol in the chain code $AF8\,(\mathcal{B})$ represents the center of a pixel in the contour shape of $\mathcal{B}$ [30].

**Definition 4** Let $Xa^{p}\,(Ya^{q})^{r}$ be a digital straight segment in the 2D object $\mathcal{B}$. The center of the pixel represented by $X$ is called a *key point* of $\mathcal{B}$. We can see the chain code $AF8\,(\mathcal{B})$ as a concatenation of digital straight segments [37]:

$$AF8\,(\mathcal{B}) = X_1a^{p_1}\,(Y_1a^{q_1})^{r_1}\,X_2a^{p_2}\,(Y_2a^{q_2})^{r_2}\,\cdots\,X_\ell a^{p_\ell}\,(Y_\ell a^{q_\ell})^{r_\ell}$$

In this case, the *key points* of $\mathcal{B}$ are the centers of the pixels defined by the symbols $X_1, \ldots, X_\ell$.

*Remark 5* The representation of the chain code $AF8\,(\mathcal{B})$ in terms of the digital straight segments given in Definition 4 is not unique. Consequently, the number of key points of $\mathcal{B}$ depends on the non-negative integers $p_i$'s, $q_i$'s, and $r_i$'s. For instance, if we take $p_i = q_i = r_i = 0$ for all $i$, we obtain that every pixel in the contour shape of $\mathcal{B}$ is a key point.

*Example 6* Fig. 2(a) shows a 2D object $\mathcal{B}$, which is the slice of a 3D object. If we start from the deep-most and leftmost voxel of the slice, we obtain that the angle Freeman chain code is given by $AF8\,(\mathcal{B}) = caaabbaaabhbbbbahbaaa = ca^3bba^3bhbbbbahba^3$. See Fig. 2 (b). By Definition 3, we can decompose $AF8\,(\mathcal{B})$ as the concatenation of the following digital straight segments:

$$ca^3, \qquad b, \qquad ba^3bh, \qquad b, \qquad b, \qquad b, \qquad bahba^3.$$

Using Definition 4, we obtain that the key points of $\mathcal{B}$ are given by the centers of the pixels defined by the symbols at the beginning of every digital straight segment. Thus, we obtain 7 key points. See Fig. 2(c), where the dark balls represent the key points.

The natural candidates to define the point clouds associated with $\mathcal{D}$ are the key points of the connected components $\mathcal{D}_{ij}$ of the $z$-slices of $\mathcal{D}$. The more points we take to define the

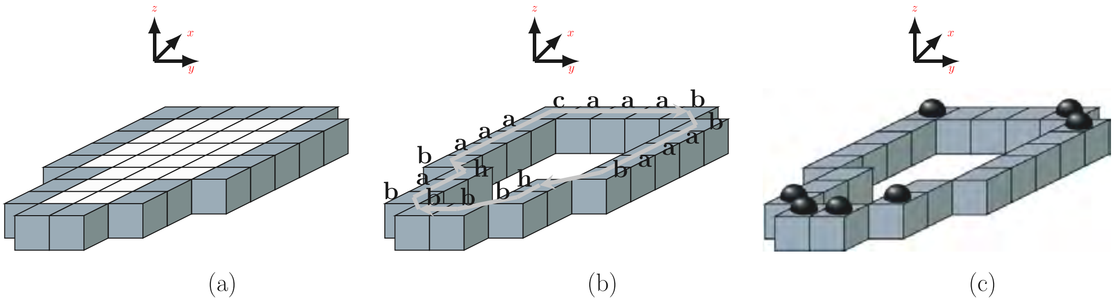


**Fig. 2** (a) A connected component of a slice of a 3D object. (b) The $AF8$ chain code of (a). (c) The key points of (a)

cloud, the better the resolution, but a higher price because of the number of key points we need to find and store. For this reason, it is vital to determine an optimal number of points in such a way that the object is well-represented.

By Remark 5, the number of key points of $\mathcal{D}_{ij}$ depends on a set of non-negative integers. We are interested in representing $AF8\left(\mathcal{D}_{ij}\right)$ using digital straight segments of a fixed size to obtain an applicable algorithm that finds a point cloud of $\mathcal{D}$. So, we have the following definition.

**Definition 7** Let $\mathcal{B}$ be a 2D object and fix positive integers $p, q$, and $r$. Let $AF8\,(\mathcal{B}) = X_1 X_2 \cdots X_\ell$ be the angle Freeman chain code of $\mathcal{B}$. Assume that the symbols next to $X_i$ form either the substring $a^p\,(bha^q)^r$ or $a^p\,(hba^q)^r$. The center of the voxel represented by the symbol $X_i$ is called a *key point* of $\mathcal{B}$ with *parameters* $p, q$, and $r$. Note that the set of key points depends only on $p, q$, and $r$.

We are ready to define the sequence of point clouds that we associate with the 3D object $\mathcal{D}$.

**Definition 8** Let $p, q$, and $r$ be non-negative integers, $\delta$ a positive integer, $\mathcal{D}$ a 3D object, and $\mathcal{D}_1, \ldots, \mathcal{D}_n$ the $z$-slices of $\mathcal{D}$. The *point cloud* associated with $\mathcal{D}$ with parameters $p, q, r$, and $\delta$, denoted by $\mathbf{P}\,(\mathcal{D}, p, q, r, \delta)$, is given by the set of key points with parameters $p, q$, and $r$ of the components of $\mathcal{D}_1, \mathcal{D}_{1+\delta}, \mathcal{D}_{1+2\delta}, \ldots, \mathcal{D}_{1+i\delta}, \ldots$. The *sequence of point clouds* associated with $\mathcal{D}$ is the collection of all point clouds associated with $\mathcal{D}$.

We come to one of the main results of this section. The following Theorem shows that for every set of integers $p, q, r$, and $\delta$, we obtain a point cloud representing the 3D object $\mathcal{D}$. This means that we are getting a sequence of point clouds representing $\mathcal{D}$.

**Theorem 9** *Let $p, q$, and $r$ be non-negative integers, $\delta$ a positive integer, and $\mathcal{D}$ a 3D object. The point cloud $\mathbf{P}\,(\mathcal{D}, p, q, r, \delta)$ can be found using the chain code $CC\,(\mathcal{D})$.*

*Proof* By Definition 1, the chain code $CC\,(\mathcal{D})$ contains the angle Freeman chain codes of the connected components of the $z$-slices $\mathcal{D}_1, \mathcal{D}_{1+\delta}, \mathcal{D}_{1+2\delta}, \ldots, \mathcal{D}_{1+i\delta}, \ldots$.

Due to Definitions 7 and 8, $\mathbf{P}\,(\mathcal{D}, p, q, r, \delta)$ depends on the substring $Xa^p\,(Ya^q)^r$ of the $z$-slices $\mathcal{D}_1, \mathcal{D}_{1+\delta}, \mathcal{D}_{1+2\delta} \ldots$. Thus, we obtain the sequence of point clouds directly from the chain code $CC\,(\mathcal{D})$. ☐

The following example shows a connected component of a 3D object, the chain code of this component, and three different point clouds that belong to the sequence of point clouds associated with this component.

*Example 10* Figure 3(a) shows a component with three slices of a 3D object. Figure 3(b) displays that the chain code of this component is

$$y1baaabhbbbbahbaaacaaaby2bbhbhcabbhbhbbhcaaabbhy3bbhbhcbabhbbababhch.$$

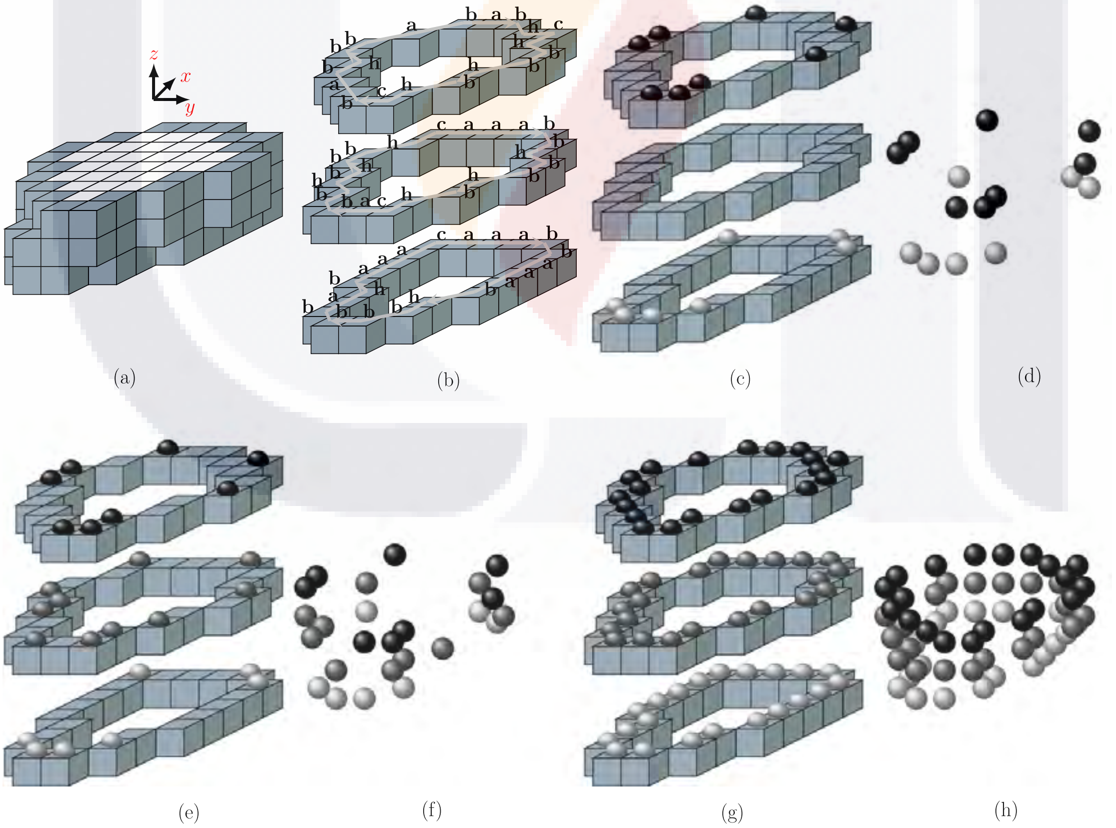
Note that the chain code contains the $AF8$ code of each slice.

Figure 3(c) shows the key points of (b) associated with $p = 3, q = 3, r = 2$, and $n = 2$. We obtained these key points by finding the substrings $Xa^3\left(Ya^3\right)^2$ in the chain code, where $X \in \{a, b, c, d, e, f, g, h\}$ and $Y \in \{bh, hb\}$. See Example 6 for a detailed explanation. Figure 3(d) contains the point cloud formed by the key points of (c).

In a similar way to what we did for (c), Figure 3(e) shows the key points of (b) associated with $p = 3, q = 3, r = 1$, and $n = 1$. Figure 3(f) contains the point cloud formed by the key points of (c).

Finally, Fig. 3(g) shows the key points of (b) associated with $p = 0, q = 0, r = 0$, and $n = 1$. Figure 3(h) contains the point cloud formed by the key points of (c).

The following example shows a point cloud of a 3D object.



(a) (b) (c) (d)
(e) (f) (g) (h)

**Fig. 3** The process to obtain 3D point clouds with different densities. (a) A connected component of a 3D object. (b) The chain code of (a) contains the $AF8$ code of each slice. (c) The key points of (b) with parameters $p = 3, q = 3, r = 2, n = 2$. (d) The point cloud associated with (c). (e) The key points of (b) with parameters $p = 3, q = 3, r = 1, n = 1$. (f) The point cloud associated with (e). (g) The key points of (b) with parameters $p = 0, q = 0, r = 0, n = 1$. (h) The point cloud associated with (g)

*Example 11* Figure 4 shows the point cloud $\mathbf{P}(\mathcal{D}, 4, 4, 4, 4)$ , where $\mathcal{D}$ is the 3D voxel Lion obtained from the voxelization of the Lion in Fig. 9 at a resolution of $512 \times 512 \times 512$. We first compute the chain code of the voxel Lion. Then we find the point cloud using Theorem 9 and the parameters $p = 4, q = 4, r = 4$, and $\delta = 4$.

## 4 Properties of the point clouds

In this section, we analyze the behavior of the point cloud $\mathbf{P}(\mathcal{D}, p, q, r, \delta)$ in terms of the integers $p, q, r$, and $\delta$. By Theorem 9, the chain code $CC(\mathcal{D})$ determines the point cloud by finding the substrings

$$Xa^p \left(Ya^q\right)^r ,$$

where $X \in \{a, b, c, d, e, f, g, h\}$ and $Y \in \{bh, hb\}$. The first consequence is that if $N$ is the number of voxels on the surface of $\mathcal{D}$, then we have the following bounds for the parameters:

$$0 \leq p \leq N, \quad 0 \leq q \leq N/3, \quad 0 \leq r \leq \log_{q+2} N, \quad \text{and} \quad 0 \leq \delta \leq N. \tag{4.1}$$

*Remark 12* The lower the parameters $p, q, r$, and $\delta$, the better the resolution of the point cloud $\mathbf{P}(\mathcal{D}, p, q, r, \delta)$. Indeed, if the parameters are small, we find more strings $Xa^p(Ya^q)^r$ in the chain code $CC(\mathcal{D})$. Thus, we get more key points, and as a consequence, the point cloud is denser and the resolution better. When the parameters approach their upper bounds of the Inequalities (4.1), there are fewer strings $Xa^p(Ya^q)^r$ in $CC(\mathcal{D})$, and the point cloud may lose some original object properties.

We now give two examples to illustrate that the lower the parameters $p, q, r$, and $\delta$, the better the resolution.

*Example 13* Let $\mathcal{D}$ be the 3D voxel Lion obtained from the voxelization of the Lion in Fig. 9 at a resolution of $512 \times 512 \times 512$. In Fig. 5, we observe that the lower the parameters

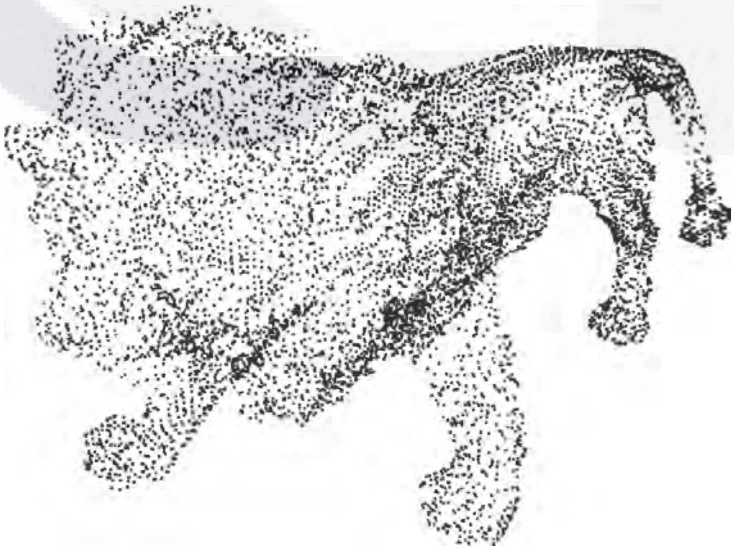


**Fig. 4** Point cloud $\mathbf{P}(\mathcal{D}, 4, 4, 4, 4)$ , where $\mathcal{D}$ is the 3D voxel Lion obtained from the voxelization of the Lion in Fig. 9 at a resolution of $512 \times 512 \times 512$

**Fig. 5** The lower the parameters $p, q, r$, and $\delta$, the better the resolution. Let $\mathcal{D}$ be the 3D voxel Lion obtained from the voxelization of the Lion in Fig. 9 at a resolution of $512 \times 512 \times 512$. (a) Shows $\mathbf{P}(\mathcal{D}, 13, 9, 7, 5)$. (b) Shows $\mathbf{P}(\mathcal{D}, 4, 4, 4, 4)$

$p, q, r$, and $\delta$, the better the resolution. Figure 5(a) shows $\mathbf{P}(\mathcal{D}, 13, 9, 7, 5)$. Figure 5(b) shows $\mathbf{P}(\mathcal{D}, 4, 4, 4, 4)$.

*Example 14* Let $\mathcal{D}$ be the 3D voxel Cow obtained from the voxelization of the Cow in Fig. 9 at a resolution of $256 \times 256 \times 256$. In Fig. 6, we observe that the lower the parameters $p, q, r$, and $\delta$, the better the resolution. Figure 6(a) shows $\mathbf{P}(\mathcal{D}, 31, 31, 6, 20)$. Figure 6(b) shows $\mathbf{P}(\mathcal{D}, 16, 16, 7, 10)$. Figure 6(c) shows $\mathbf{P}(\mathcal{D}, 14, 14, 6, 5)$. Figure 6(d) shows $\mathbf{P}(\mathcal{D}, 4, 4, 4, 4)$.

## 4.1 Rotational invariance

It is needed to analyze the rotational invariance. The reason is that the sequence of point clouds that we just defined in the last section depends on the voxelization, which is not invariant under rotations. Indeed, assume that $\mathcal{F}$ is a 3D object given in a PLY format. Suppose we obtain the voxelization of $\mathcal{F}$, and we rotate. In this case, we do not necessarily get the same as if we first rotate $\mathcal{F}$ and after we obtain the voxelization.

Let $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3)$ be a vector of real numbers such that $0 \leq \theta_i < 2\pi$. $\mathcal{D}\boldsymbol{\theta}$ denotes the rotation of the angles $\theta_1$ about the $x$-axis, $\theta_2$ about the $y$-axis, and $\theta_3$ about the $z$-axis of the object $\mathcal{D}$. Observe that $\mathbf{P}(\mathcal{D}\boldsymbol{\theta}, p, q, r, \delta)$ denotes the point cloud of the rotated object $\mathcal{D}\boldsymbol{\theta}$.

**Fig. 6** The lower the parameters $p, q, r$, and $\delta$, the better the resolution. Let $\mathcal{D}$ be the 3D voxel Cow obtained from the voxelization of the Cow in Fig. 9 at a resolution of $256 \times 256 \times 256$. (a) Shows $\mathbf{P}(\mathcal{D}, 31, 31, 6, 20)$. (b) Shows $\mathbf{P}(\mathcal{D}, 16, 16, 7, 10)$. (c) Shows $\mathbf{P}(\mathcal{D}, 14, 14, 6, 5)$. (d) Shows $\mathbf{P}(\mathcal{D}, 5, 5, 4, 4)$

**Fig. 7** Let $\mathcal{D}$ be the 3D voxel Brain obtained from the voxelization of the Brain in Fig. 9 at a resolution of $256 \times 256 \times 256$. The Black dots in the top-left picture represent $\mathbf{P}(\mathcal{D}, 13, 13, 4, 5)$. In the top-middle picture, the black dots represent $\mathbf{P}(\mathcal{D}\boldsymbol{\theta}, 10, 8, 8, 5)$, where $\boldsymbol{\theta} = \left(\frac{\pi}{12}, 0, 0\right)$. In the top-right picture, the black dots represent $\mathbf{P}(\mathcal{D}\boldsymbol{\theta}, 8, 9, 7, 5)$, with $\boldsymbol{\theta} = \left(\frac{\pi}{6}, 0, 0\right)$. In the left picture, the black dots represent $\mathbf{P}(\mathcal{D}\boldsymbol{\theta}, 10, 8, 5, 5)$, where $\boldsymbol{\theta} = \left(\frac{\pi}{4}, 0, 0\right)$

*Example 15* Let $\mathcal{D}$ be the 3D voxel Brain obtained from the voxelization of the Brain in Fig. 9 at a resolution of $256 \times 256 \times 256$. The Black dots in the top-left picture represent $\mathbf{P}(\mathcal{D}, 13, 13, 4, 5)$. In the top-middle picture, the black dots represent $\mathbf{P}(\mathcal{D}\boldsymbol{\theta}, 10, 8, 8, 5)$, where $\boldsymbol{\theta} = \left(\frac{\pi}{12}, 0, 0\right)$ means that we rotate $\mathcal{D}$ the angles $\pi/12$ about the $x$-axis, 0 about the $y$-axis, and 0 about the $z$-axis. In the top-right picture, the black dots represent $\mathbf{P}(\mathcal{D}\boldsymbol{\theta}, 8, 9, 7, 5)$, with $\boldsymbol{\theta} = \left(\frac{\pi}{6}, 0, 0\right)$. In the left picture, the black dots represent $\mathbf{P}(\mathcal{D}\boldsymbol{\theta}, 10, 8, 5, 5)$, where $\boldsymbol{\theta} = \left(\frac{\pi}{4}, 0, 0\right)$ (Fig. 7).

Note that the sets of voxels $\mathbf{P}(\mathcal{D}, p, q, r, \delta)$ and $\mathbf{P}(\mathcal{D}\boldsymbol{\theta}, p, q, r, \delta)$ are not always the same because, in general, a rigid rotation does not preserve key points [37]. A rigid rotation does not even maintain the number of voxels when every $\theta_i$ is not a multiple of $\frac{\pi}{2}$. In other words, in general, the number of voxels in $\mathcal{D}$ and $\mathcal{D}\boldsymbol{\theta}$ is different when $\frac{\pi}{2}$ does not divide each $\theta_i$. So, it makes no sense to ask if the sets of vertices $\mathbf{P}(\mathcal{D}, p, q, r, \delta)$ and $\mathbf{P}(\mathcal{D}\boldsymbol{\theta}, p, q, r, \delta)$ are equal.

The Hausdorff distance and the average geometric error are excellent tools for comparing point clouds with distinct cardinalities. These two functions, which have different properties, have been used independently in the literature to measure the similitude between two sets of points living in the same metric space [36, 46]. As we see in Section 5.2, these two functions help to prove that our point clouds are invariant under rigid rotations.

**Definition 16** Let $V_1$ and $V_2$ be two sets of voxels. The *Hausdorff distance* and the *average geometric error* between $V_1$ and $V_2$ are given, respectively, by

$$Hau(V_1, V_2) := \max\left\{\max_{v_1 \in V_1}\left\{\min_{v_2 \in V_2}\{\|v_1, v_2\|\}\right\}, \max_{v_2 \in V_2}\left\{\min_{v_1 \in V_1}\{\|v_1, v_2\|\}\right\}\right\} \text{ and}$$

$$\triangle_{\text{avg}}(V_1, V_2) := \frac{1}{|V_1|}\sum_{v_1 \in V_1}\min_{v_2 \in V_2}\{\|v_1, v_2\|\},$$

where $\|v_1, v_2\|$ denotes the Euclidean distance between the centers of the voxels $v_1$ and $v_2$, and $\mid V_1 \mid$ represents the cardinality of $V_1$.

As we see in Section 5, the sequence of point clouds associated with $\mathcal{D}$ in Theorem 9 is invariant under rotations because the Hausdorff distance and the average geometric error between sets $\mathbf{P}(\mathcal{D}, p, q, r, \delta)$ and $\mathbf{P}(\mathcal{D}\boldsymbol{\theta}, p, q, r, \delta)$ are relatively small.

*Example 17* Let $\mathcal{D}$ be the 3D voxel Cow obtained from the voxelization of the Cow in Fig. 9 at a resolution of $256 \times 256 \times 256$. Figure 8(a) shows the set $V_1 := \mathbf{P}(\mathcal{D}, 16, 16, 7, 10)$ and (b) shows the set $V_2 := \mathbf{P}(\mathcal{D}\boldsymbol{\theta}, 14, 14, 9, 10)$, where $\boldsymbol{\theta} = \left(\frac{4\pi}{9}, \frac{5\pi}{18}, 0\right)$. The Hausdorff distance and the average geometric error between $V_1$ and $V_2$ are $Hau(V_1, V_2) = 21.739$ and $\triangle_{\text{avg}}(V_1, V_2) = 7.860$, respectively. These numbers are relatively small. If we take, for instance, the Cow and the Lion in Fig. 9, which are in a PLY format, the Hausdorff distance and the average geometric error between the vertices of the PLYs are 58.22 and 22.05, respectively.

## 5 Results

In this section, we study the behavior of the point clouds associated with the objects shown in Fig. 9.

We first find the voxelization of the objects, which are initially in a PLY format. A thick voxelization is crucial because more voxels will give a chain code $CC(\mathcal{D})$ with more key points. Thus, we get a denser point cloud and a better representation of the 3D objects.

By Theorem 9, the chain code of $\mathcal{D}$ determines the point clouds. So, after the voxelization, we get the chain code for every voxel 3D object.

By Remark 12, the lower the values of $p, q, r$, and $\delta$, the better the resolution of the point clouds, and the larger the values, the worsen the description. To obtain a complete sequence analysis, we define below high, medium, and low densities of point clouds associated with $\mathcal{D}$.

We may be tempted to fix the parameters $p, q, r$, and $\delta$ to define the different densities for all the objects. For instance, we may always use $p = q = r = \delta = 5$ for the high


(a) (b)

**Fig. 8** Let $\mathcal{D}$ be the 3D voxel Cow obtained from the voxelization of the Cow in Fig. 9 at a resolution of $256 \times 256 \times 256$. (a) Shows $V_1 := \mathbf{P}(\mathcal{D}, 16, 16, 7, 10)$. (b) Shows $V_2 := \mathbf{P}(\mathcal{D}\boldsymbol{\theta}, 14, 14, 9, 10)$, where $\boldsymbol{\theta} = \left(\frac{4\pi}{9}, \frac{5\pi}{18}, 0\right)$. The Hausdorff distance and the average geometric error between $V_1$ and $V_2$ are $Hau(V_1, V_2) = 21.739$ and $\triangle_{\text{avg}}(V_1, V_2) = 7.860$, respectively

**Fig. 9** A set of three-dimensional objects in PLY format

density for every 3D object. This approach has a downside. As we do not know the behavior of the chain codes, this set of parameters may give a dense point cloud for a particular 3D object but a sparse point cloud for a different 3D object. For this reason, to make sure we are getting different densities, we first need to analyze the frequencies of the symbols of the chain codes. The process is the following.

Fix an integer $\delta \geq 1$. Find $CC(\mathcal{D}, \delta)$, which is the substring of the chain code $CC\,(\mathcal{D})$ representing the $z$-slices $\mathcal{D}_1, \mathcal{D}_{1+\delta}, \mathcal{D}_{1+2\delta} \ldots$ of $\mathcal{D}$. Define the frequency function $f_p \colon \mathbb{Z}_{\geq 1} \to \mathbb{Z}_{\geq 0}$, $i \to f_p(i)$ as the number of times that the chains $a^i\,(hba^q)^r$ or $a^i\,(bha^q)^r$ appear in $CC\,(\mathcal{D}, \delta)$, where $q, r \geq 0$. Similarly, define the frequency functions $f_q(i)$ and $f_r(i)$. In other words,

$$
\begin{aligned}
f_p(i) &:= \left|\left\{Xa^i S \in CC(\mathcal{D}, \delta) : X \notin \{a, bh, hb\}, S \neq a\right\}\right|,\\
f_q(i) &:= \left|\left\{Ya^i S \in CC(\mathcal{D}, \delta) : Y \in \{bh, hb\}, S \neq a\right\}\right|, \quad \text{and}\\
f_r(i) &:= \left|\left\{\left(Ya^q\right)^i S \in CC(\mathcal{D}, \delta) : q \geq 0, Y \in \{bh, hb\}, S \neq Ya^q\right\}\right|,
\end{aligned}
$$

where $|\cdot|$ stands for cardinality. Let $p_1$, $p_{max} \geq 1$ be the least and greatests integer, respectively, such that $f_p(p_1) = 0$ and $f_p(p_{max}) \neq 0$. Similarly, we define the integers $q_1$, $q_{max}$, $r_1$, and $r_{max}$.

*Example 18* We first obtain the voxel objects of Fig. 9 at a resolution of 256 × 256 × 256. Figure 10 shows the frequency functions $f_p(i)$ associated with these voxel objects and $\delta = 20$. We present here the main properties of $f_p(i)$.

- The top-left image shows all the values $i \geq 1$ for which $f_p(i) \neq 0$. We can observe that $p_{max} < 100$ for all the objects. We see that for small values of $i$, the decreasing rate of the functions $f_p(i)$ is high, which means that the functions $f_p(i)$ decreases immediately.
- The top-right image shows that the functions $f_p(i)$ are not strictly decreasing.

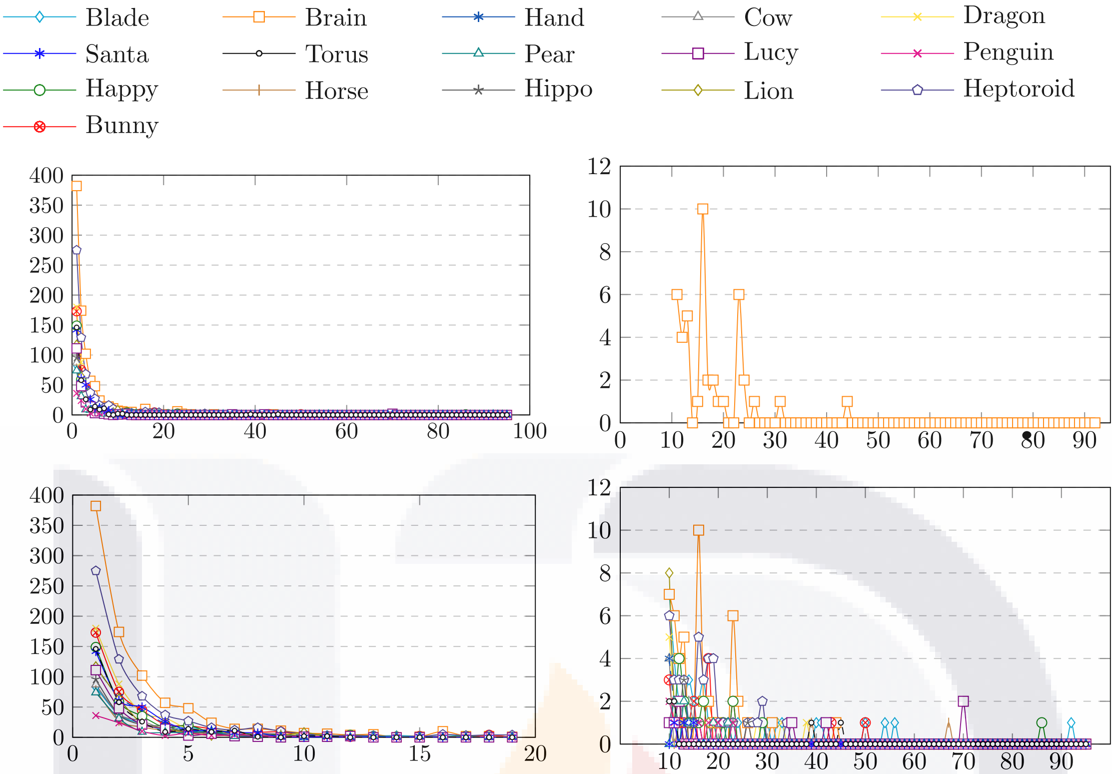


**Fig. 10** Frequency functions $f_p(i)$ associated with the voxel objects of Fig. 9 and $\delta = 20$. Among the main properties: $p_{max} < 100$ for all the objects; for small values of $i$, the decreasing rate is high; for $i \geq 10$, $f_p(i) \leq 10$; and the functions are not strictly decreasing

- The bottom-left image shows that when $i > 5$, all the values of $f_p(i)$ are upper bounded by 50. When $i \geq 10$, we have that $f_p(i) \leq 10$.
- Finally, the bottom-right image shows the frequency function $f_p(i)$ for the voxel Brain from Fig. 9.

*Remark 19* The integers $p_1$, $p_{max}$, $q_1$, $q_{max}$, $r_1$, and $r_{max}$ depend on the functions $f_p$, $f_q$, and $f_r$. These functions are built in terms of $\delta$. Thus, the integers $p_1$, $p_{max}$, $q_1$, $q_{max}$, $r_1$, and $r_{max}$ are related to $\delta$.

Example 18 and Fig. 10 conclude that $f_p\left(\left\lfloor \frac{p_1}{2} \right\rfloor\right)$ is much higher than $f_p\left(\left\lfloor \frac{p_{max}}{2} \right\rfloor\right)$, which is higher than $f_p\left(p_{max}\right)$. A similar study shows the same properties for the functions $f_q(i)$ and $f_r(i)$. Consequently,

$$\left|\, \mathbf{P}\left(\mathcal{D}, \left\lfloor \frac{p_1}{2} \right\rfloor, \left\lfloor \frac{q_1}{2} \right\rfloor, \left\lfloor \frac{p_1}{2} \right\rfloor, \delta\right) \right| > \left|\, \mathbf{P}\left(\mathcal{D}, \left\lfloor \frac{p_{max}}{2} \right\rfloor, \left\lfloor \frac{q_{max}}{2} \right\rfloor, \left\lfloor \frac{r_{max}}{2} \right\rfloor, \delta\right) \right|$$
$$> \left|\, \mathbf{P}\left(\mathcal{D}, p_{max}, q_{max}, r_{max}, \delta\right) \right|.$$

Inspired by previous inequalities, we can now define the point clouds with different densities representing $\mathcal{D}$.

**Definition 20** The *Point cloud of density H* associated with $\mathcal{D}$ is given by

$$\mathbf{P}_H\left(\mathcal{D}\right) = \mathbf{P}\left(\mathcal{D}, \left\lfloor \frac{p_1}{2} \right\rfloor, \left\lfloor \frac{q_1}{2} \right\rfloor, \left\lfloor \frac{r_1}{2} \right\rfloor, 5\right),$$

where $p_1, q_1$, and $r_1$ are related to $\delta = 5$. The *Point cloud of density M* associated with $\mathcal{D}$ is given by

$$\mathbf{P}_M(\mathcal{D}) = \mathbf{P}\left(\mathcal{D}, \left\lfloor \frac{p_{max}}{2} \right\rfloor, \left\lfloor \frac{q_{max}}{2} \right\rfloor, \left\lfloor \frac{r_{max}}{2} \right\rfloor, 10\right),$$

where $p_{max}, q_{max}$, and $r_{max}$ are related to $\delta = 10$. he *Point cloud of density L* associated with $\mathcal{D}$ is given by

$$\mathbf{P}_L(\mathcal{D}) = \mathbf{P}(\mathcal{D}, p_{max}, q_{max}, r_{max}, 20),$$

where $p_{max}, q_{max}$, and $r_{max}$ are related to $\delta = 20$.

It is important to emphasize that these three different point clouds that we just defined in Definition 20 are only a sample of point clouds from the sequence. The sequence obtained from Theorem 9 allows us to define point clouds with ultra-high or with very low density.

*Example 21* Here, we show an example with the three densities: low, medium, and high. Let $\mathcal{D}$ be the voxel Bunny obtained from the voxelization of the Bunny in Fig. 9 at a resolution of $512 \times 512 \times 512$. Figure 11(a) shows $\mathbf{P}_L(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 339, 259, 27, 20)$. Figure 11(b) shows $\mathbf{P}_M(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 169, 129, 13, 10)$. Figure 11(c) shows $\mathbf{P}_H(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 19, 25, 11, 5)$.

*Example 22* Here, we show an example with three densities: low, medium, and high. Let $\mathcal{D}$ be the voxel Cow obtained from the voxelization of the Cow in Fig. 9 at a resolution of $512 \times 512 \times 512$. Figure 12(a) shows $\mathbf{P}_L(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 168, 115, 31, 20)$. Figure 12(b) shows $\mathbf{P}_M(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 84, 57, 15, 10)$. Figure 12(c) shows $\mathbf{P}_H(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 32, 17, 11, 5)$.

*Example 23* Here, we show an example with three densities: low, medium, and high. Let $\mathcal{D}$ be the voxel Dragon obtained from the voxelization of the Dragon in Fig. 9 at a resolution of $512 \times 512 \times 512$. Figure 13(a) shows $\mathbf{P}_L(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 168, 86, 24, 20)$. Figure 13(b) shows $\mathbf{P}_M(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 84, 43, 12, 10)$. Figure 13(c) shows $\mathbf{P}_H(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 58, 20, 10, 5)$.

*Example 24* Here, we show an example with three densities: low, medium, and high. Let $\mathcal{D}$ be the voxel Lion obtained from the voxelization of the Lion in Fig. 9 at a resolution of $512 \times 512 \times 512$. Figure 14(a) shows $\mathbf{P}_L(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 131, 31, 38, 20)$. Figure 14(b) shows $\mathbf{P}_M(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 65, 15, 19, 10)$. Figure 14(c) shows $\mathbf{P}_H(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 11, 13, 11, 5)$.

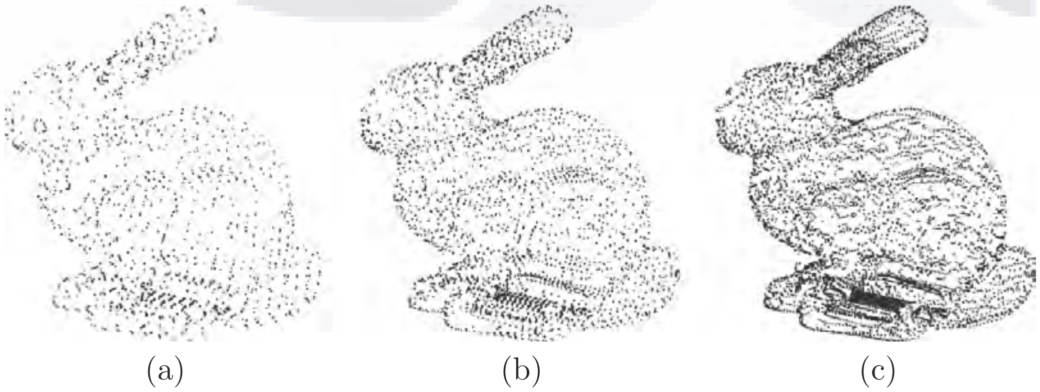

(a) (b) (c)

**Fig. 11** Let $\mathcal{D}$ be the voxel Bunny obtained from the voxelization of the Bunny in Fig. 9 at a resolution of $512 \times 512 \times 512$. (a) Shows $\mathbf{P}_L(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 339, 259, 27, 20)$. (b) Shows $\mathbf{P}_M(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 169, 129, 13, 10)$. (c) Shows $\mathbf{P}_H(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 19, 25, 11, 5)$

**Fig. 12** Let $\mathcal{D}$ be the voxel Cow obtained from the voxelization of the Cow in Fig. 9 at a resolution of $512 \times 512 \times 512$. (a) Shows $\mathbf{P}_L(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 168, 115, 31, 20)$. (b) Shows $\mathbf{P}_M(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 84, 57, 15, 10)$. (c) Shows $\mathbf{P}_H(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 32, 17, 11, 5)$

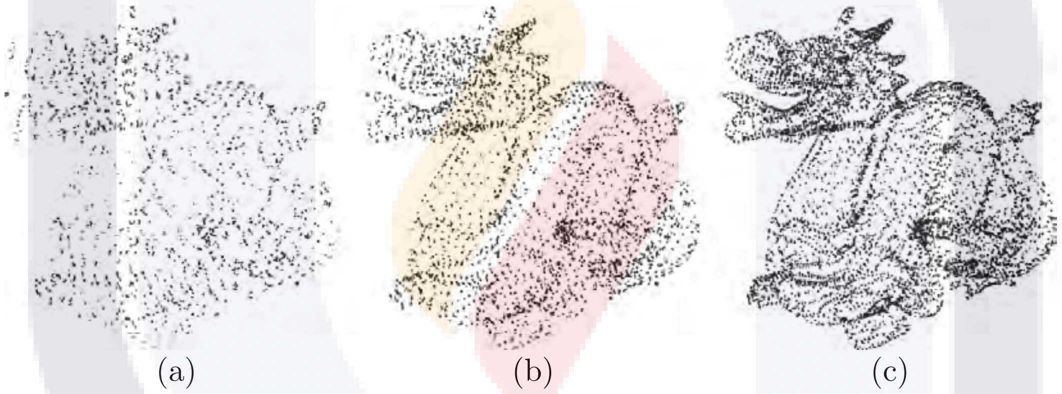


**Fig. 13** Let $\mathcal{D}$ be the voxel Dragon obtained from the voxelization of the Dragon in Fig. 9 at a resolution of $512 \times 512 \times 512$. (a) Shows $\mathbf{P}_L(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 168, 86, 24, 20)$. (b) Shows $\mathbf{P}_M(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 84, 43, 12, 10)$. (c) Shows $\mathbf{P}_H(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 58, 20, 10, 5)$

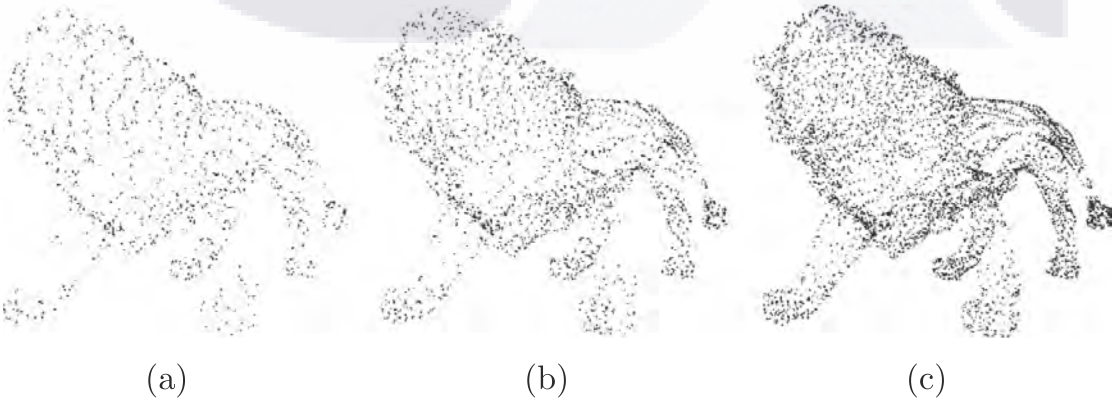


**Fig. 14** Let $\mathcal{D}$ be the voxel Lion obtained from the voxelization of the Lion in Fig. 9 at a resolution of $512 \times 512 \times 512$. (a) Shows $\mathbf{P}_L(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 131, 31, 38, 20)$. (b) Shows $\mathbf{P}_M(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 65, 15, 19, 10)$. (c) Shows $\mathbf{P}_H(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 11, 13, 11, 5)$

*Example 25* Here, we show an example with three densities: low, medium, and high. Let $\mathcal{D}$ be the voxel Santa obtained from the voxelization of the Santa in Fig. 9 at a resolution of $512 \times 512 \times 512$. Figure 15(a) shows $\mathbf{P}_L(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 186, 38, 27, 20)$. Figure 15(b) shows $\mathbf{P}_M(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 93, 19, 13, 10)$. Figure 15(c) shows $\mathbf{P}_H(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 21, 10, 10, 5)$.

### 5.1 Proposed method vs literature

We use a PC with Intel (R) Core (TM), i7-8706G, at 3.10GHz, 3.10GHz CPU, and 16 GB RAM for the computations. Let $\mathcal{D}$ be the voxel Bunny obtained from the voxelization of the Bunny in Fig. 9 at a resolution of $256 \times 256 \times 256$. Table 1 shows a comparison between the point clouds $\mathbf{P}(\mathcal{D}, 19, 25, 11, 5)$, $\mathbf{P}(\mathcal{D}, 169, 129, 13, 10)$, and $\mathbf{P}(\mathcal{D}, 339, 259, 27, 20)$ and the simplifications methods that are proposed in [2, 4, 6, 15, 18, 19, 25, 32, 34, 36, 42–44, 53]. Image in Fig. 16 shows the point clouds.

In Table 1, we can see that our three proposed point clouds with different densities have fewer points than other simplifications methods. Recall that we first obtain the voxel object from the original PLY file. The voxelization of the Bunny at a resolution of $256 \times 256 \times 256$ requires 27.51 sec. Considering this preprocessing time, plus the seconds it takes to get the three proposed point clouds, we summarize a total of 27.948 sec. Thus, we obtain each point cloud in an average of 9.316 sec. Additionally, the average size of the three proposed point clouds is 1,438. Regarding the best result in the literature, Ref [42], which has 3,594 points, our point clouds represent an average efficiency of 60% concerning the size.

Current literature focuses on the trade-off between the execution time of the method and the number of points of the obtained cloud. In our case, as we already have the voxel object from the PLY file, we can observe that the obtention of point clouds with different densities requires less than a second.

### 5.2 Rotational invariance

The main goal in this section is to show that our proposed cloud points are invariant under rotation.

As an example of the rotations we are analyzing, Fig. 17 shows the Cow in Fig. 9 and its rotations $\boldsymbol{\theta}_0 := (0, 0, 0)$, $\boldsymbol{\theta}_1 := (0, 8\pi/9, 0)$, $\boldsymbol{\theta}_2 := (4\pi/9, 5\pi/18, 0)$, and $\boldsymbol{\theta}_3 := (5\pi/18, 7\pi/18, \pi/4)$.


(a) (b) (c)

**Fig. 15** Let $\mathcal{D}$ be the voxel Santa obtained from the voxelization of the Santa in Fig. 9 at a resolution of $512 \times 512 \times 512$. (a) Shows $\mathbf{P}_L(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 186, 38, 27, 20)$. (b) Shows $\mathbf{P}_M(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 93, 19, 13, 10)$. (c) Shows $\mathbf{P}_H(\mathcal{D}) = \mathbf{P}(\mathcal{D}, 21, 10, 10, 5)$

**Table 1** Comparison between our proposed point clouds and the simplifications methods from the literature

| | Original file (# of points) | Simplified model (# of points) | Method | Time (Sec) |
|---|---|---|---|---|
| Bunny | 35,947 | 11,779 | Ref [43] | Not given |
| | 35,947 | 16,476 | Ref [6] | Not given |
| | 35,947 | 3,595 | Ref [44] | 2,093 |
| | 34,835 | 13,571 | Ref [34] | 111.238 |
| | 35,947 | 3,594 | Ref [42] | 1720 |
| | 35,336 | 3,687 | Ref [32] | 15 |
| | 35,947 | 4,517 | Ref [25] | 21.223 |
| | 35,947 | 4,566 | Ref [18] | 56.156 |
| | 35,947 | 4,644 | Ref [19] | 38.094 |
| | 35,947 | 4,638 | Ref [4] | 9.574 |
| | 35,947 | 4,572 | Ref [15] | 16.678 |
| | 35,947 | 4,562 | Ref [2] | 0.692 |
| | 35,947 | 6,557 | Ref [53] | 0.173 |
| | 35,947 | 4,644 | Ref [36] | 4.459 |
| | 35,947 | 2,530 | $\mathbf{P}(\mathcal{D}, 19, 25, 11, 5)$ | 0.251 |
| | 35,947 | 1,202 | $\mathbf{P}(\mathcal{D}, 169, 129, 13, 10)$ | 0.122 |
| | 35,947 | 582 | $\mathbf{P}(\mathcal{D}, 339, 259, 27, 20)$ | 0.065 |

The voxelization of the Bunny in Fig. 9 at a 512 × 512 × 512 of resolution requires 27.51 sec

Every object in Fig. 9 is in a PLY format, containing vertices and edges. We first obtain the 16 voxel objects from Fig. 9 at a resolution of 256 × 256 × 256. Taking $\boldsymbol{\theta}_0 = (0, 0, 0)$, $\boldsymbol{\theta}_1 = (0, 8\pi/9, 0)$, $\boldsymbol{\theta}_2 = (4\pi/9, 5\pi/18, 0)$, and $\boldsymbol{\theta}_3 = (5\pi/18, 7\pi/18, \pi/4)$, we compute the following 12 point clouds with different densities and rotations for every voxel object $\mathcal{D}$:

$$\mathbf{P}_H(\mathcal{D}\boldsymbol{\theta}_i), \quad \mathbf{P}_M(\mathcal{D}\boldsymbol{\theta}_i), \quad \mathbf{P}_L(\mathcal{D}\boldsymbol{\theta}_i), \text{ for } i = 0, 1, 2, 3.$$

We now align the $12 \times 16 = 192$ point clouds with respect to their principal axes. Table 2 shows the Hausdorff distance and the average geometric error between these 192 point clouds and the vertices of the PLYs.

We can notice that the numbers in Table 2 are relatively small. If we take, for instance, the Cow and the Lion in Fig. 9, the Hausdorff distance and the average geometric error between the vertices of the PLYs are 58.22 and 22.05, respectively.

Now take two different voxel objects $\mathcal{D}$ and $\mathcal{D}'$ and the rotations $\mathcal{D}\boldsymbol{\theta}$ and $\mathcal{D}'\boldsymbol{\theta}'$. Table 3 shows the values $H_\circ$ and $H_{\text{next}}$. These numbers have the following interpretation.

- The Hausdorff distance $Hau(\mathbf{P}(\mathcal{D}), \mathbf{P}(\mathcal{D}\boldsymbol{\theta}))$ is upper bounded by $H_\circ$.
- The Hausdorff distance $Hau(\mathbf{P}(\mathcal{D}), \mathbf{P}(\mathcal{D}'\boldsymbol{\theta}))$ is lower bounded by $H_{\text{next}}$.

As we can see, the Hausdorff distances in Table 2 are small enough that given two objects, it is easy to identify if one of them is the rotation of another one.

Table 4 shows the Hausdorff distance and the average geometric error between the point clouds with no rotations and the point clouds with rotations. In other words, Table 4 shows the Hausdorff distance and the average geometric error between the point clouds $\mathbf{P}_H(\mathcal{D})$ and $\mathbf{P}_H(\mathcal{D}\boldsymbol{\theta}_i)$; $\mathbf{P}_M(\mathcal{D})$ and $\mathbf{P}_M(\mathcal{D}\boldsymbol{\theta}_i)$; and $\mathbf{P}_L(\mathcal{D})$ and $\mathbf{P}_L(\mathcal{D}\boldsymbol{\theta}_i)$ for $i = 0, 1, 2, 3$.

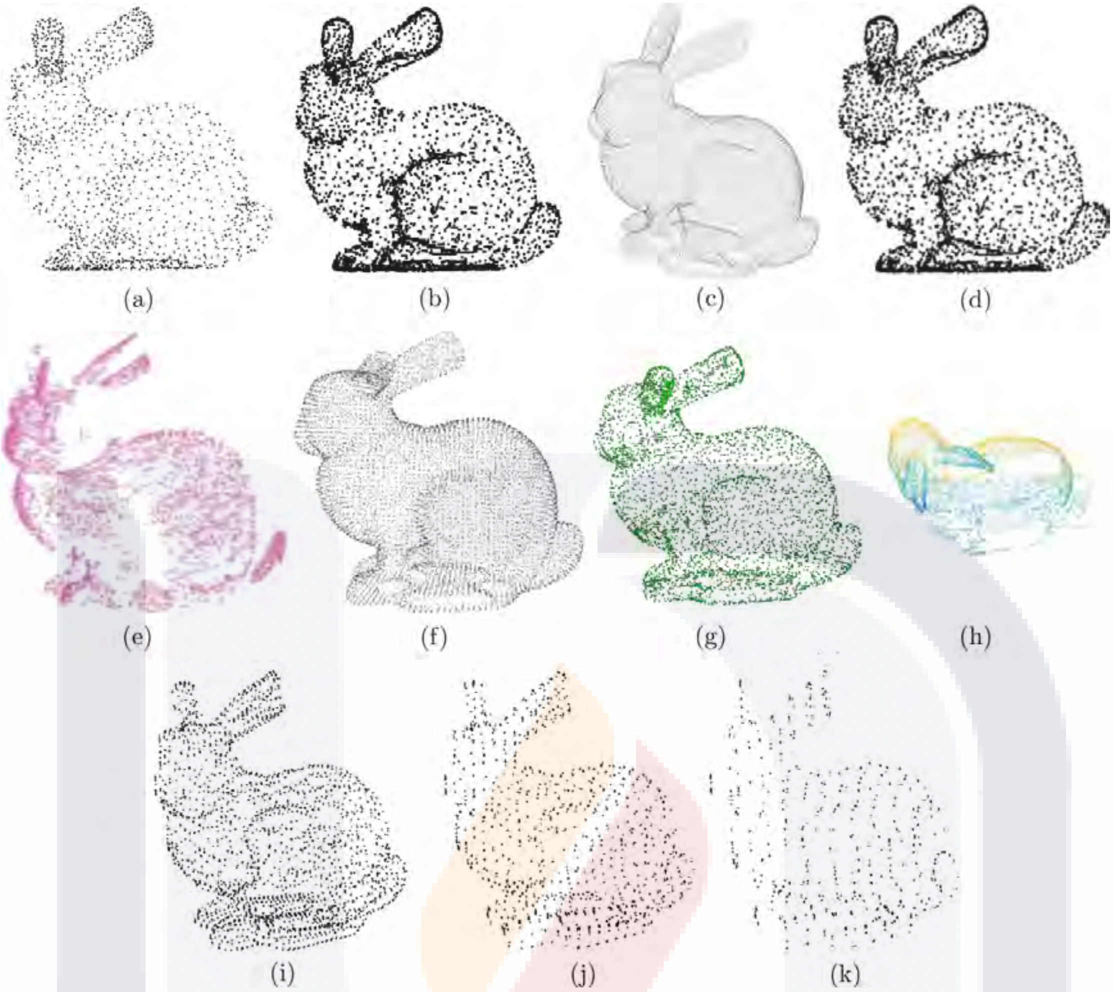


**Fig. 16** Bunny (a) Ref [43], (b) Ref [44], (c) Ref [34], (d) Ref [42], (e) Ref [32], (f) Ref [6], (g) Ref [18] (h) Ref [53],(i) $\mathbf{P}(\mathcal{D}, 19, 25, 11, 5)$, (j) $\mathbf{P}(\mathcal{D}, 169, 129, 13, 10)$, (k) $\mathbf{P}(\mathcal{D}, 339, 259, 27, 20)$, where $\mathcal{D}$ be the voxel Bunny obtained from the voxelization of the Bunny in Fig. 9 at a resolution of $256 \times 256 \times 256$

## 6 Conclusion

In this work, we introduced and developed a chain code associated with a 3D object $\mathcal{D}$. The chain code relies on the slices of $\mathcal{D}$.

Taking advantage of the properties of the chain code, we constructed a sequence of point clouds that simplify $\mathcal{D}$. The sequence contains point clouds that give different object descriptions, from a straightforward and sparse one to a very fine-tuned representation. The

**Fig. 17** Cow shape with four different orientations

**Table 2** For every object $\mathcal{D}_P$ in Fig. 9 in PLY format, this table shows the Hausdorff distance ($Hau$) and the average geometric error ($\triangle_{\text{avg}}$) between the vertices of the PLY, and the point clouds $\mathbf{P}_H\ (\mathcal{D}\boldsymbol{\theta}_i)$ , $\mathbf{P}_M\ (\mathcal{D}\boldsymbol{\theta}_i)$ , and $\mathbf{P}_L\ (\mathcal{D}\boldsymbol{\theta}_i)$ , for $i = 0, 1, 2, 3$

| Rotation | | $Rot(0, 0, 0)$ | | $Rot(0, 8\pi/9, 0)$ | | $Rot(4\pi/9, 5\pi/18, 0)$ | | $Rot(5\pi/18, 7\pi/18, \pi/4)$ | |
|---|---|---|---|---|---|---|---|---|---|
| Voxelized Shape | Density | $Hau$ | $\triangle_{\text{avg}}$ | $Hau$ | $\triangle_{\text{avg}}$ | $Hau$ | $\triangle_{\text{avg}}$ | $Hau$ | $\triangle_{\text{avg}}$ |
| Blade | $H$ | 37.358 | 8.123 | 35.532 | 7.555 | 29.555 | 7.392 | 34.151 | 8.055 |
| | $M$ | 52.140 | 12.521 | 40.694 | 10.887 | 27.453 | 8.404 | 41.681 | 10.653 |
| | $L$ | 70.382 | 20.912 | 70.833 | 18.802 | 38.612 | 11.705 | 76.703 | 17.844 |
| Brain | $H$ | 40.410 | 7.186 | 42.263 | 10.050 | 47.951 | 8.760 | 43.823 | 8.752 |
| | $M$ | 41.594 | 8.473 | 46.594 | 11.430 | 49.672 | 10.516 | 45.024 | 10.560 |
| | $L$ | 41.621 | 10.695 | 46.585 | 14.817 | 54.091 | 14.0216 | 50.867 | 13.578 |
| Bunny | $H$ | 17.176 | 4.579 | 20.789 | 4.473 | 17.242 | 4.373 | 17.034 | 4.313 |
| | $M$ | 22.777 | 6.513 | 21.507 | 6.211 | 18.892 | 5.939 | 24.569 | 6.085 |
| | $L$ | 34.895 | 9.943 | 30.517 | 9.656 | 33.526 | 9.539 | 39.262 | 9.993 |
| Cow | $H$ | 17.230 | 3.568 | 15.812 | 4.320 | 17.682 | 5.345 | 19.086 | 6.744 |
| | $M$ | 19.460 | 4.961 | 22.345 | 6.058 | 20.113 | 6.734 | 27.466 | 8.811 |
| | $L$ | 23.882 | 8.587 | 30.597 | 9.665 | 23.943 | 9.050 | 53.453 | 12.828 |
| Dragon | $H$ | 14.405 | 3.517 | 15.661 | 4.237 | 13.868 | 3.547 | 21.485 | 4.730 |
| | $M$ | 19.232 | 5.175 | 15.545 | 5.348 | 19.128 | 5.167 | 28.960 | 6.864 |
| | $L$ | 30.0477 | 8.088 | 21.858 | 7.828 | 25.029 | 8.080 | 35.703 | 10.186 |
| Hand | $H$ | 13.629 | 2.759 | 15.987 | 3.267 | 14.336 | 4.555 | 15.411 | 4.819 |
| | $M$ | 14.399 | 3.811 | 20.579 | 4.397 | 16.668 | 5.284 | 18.923 | 6.726 |
| | $L$ | 21.982 | 6.523 | 29.364 | 7.014 | 58.361 | 10.185 | 34.895 | 11.330 |
| Happy | $H$ | 13.139 | 3.494 | 13.207 | 3.721 | 14.715 | 3.796 | 14.412 | 3.755 |
| | $M$ | 18.708 | 5.309 | 20.426 | 5.544 | 20.497 | 5.119 | 27.664 | 5.632 |
| | $L$ | 24.168 | 7.694 | 25.628 | 8.709 | 28.489 | 7.669 | 33.789 | 9.559 |
| Heptoroid | $H$ | 31.388 | 3.535 | 31.265 | 3.563 | 31.758 | 4.255 | 30.818 | 3.580 |
| | $M$ | 29.995 | 4.821 | 31.708 | 4.838 | 27.328 | 5.812 | 30.032 | 4.858 |
| | $L$ | 26.477 | 7.548 | 32.097 | 7.620 | 26.759 | 8.863 | 28.603 | 7.616 |
| Hippo | $H$ | 14.486 | 3.102 | 19.385 | 3.808 | 15.423 | 3.022 | 19.873 | 3.699 |
| | $M$ | 15.907 | 4.569 | 21.549 | 5.098 | 21.559 | 4.754 | 32.583 | 5.948 |
| | $L$ | 21.838 | 8.957 | 23.762 | 7.978 | 28.700 | 7.368 | 41.731 | 11.100 |
| Horse | $H$ | 15.927 | 4.478 | 18.362 | 4.686 | 17.721 | 6.2501 | 16.384 | 4.081 |
| | $M$ | 25.280 | 6.673 | 19.866 | 6.244 | 20.118 | 7.305 | 19.762 | 5.361 |
| | $L$ | 29.976 | 9.377 | 33.656 | 9.806 | 38.937 | 9.518 | 28.504 | 8.563 |
| Lion | $H$ | 14.287 | 3.338 | 13.050 | 3.691 | 12.867 | 3.793 | 15.212 | 3.746 |
| | $M$ | 15.700 | 4.675 | 16.466 | 4.918 | 15.533 | 5.126 | 18.450 | 5.419 |
| | $L$ | 24.0402 | 7.701 | 23.843 | 7.893 | 23.141 | 8.041 | 33.770 | 9.359 |
| Lucy | $H$ | 15.463 | 3.433 | 13.793 | 3.468 | 15.507 | 3.490 | 18.139 | 4.085 |
| | $M$ | 20.940 | 5.019 | 18.121 | 5.241 | 20.461 | 4.910 | 23.207 | 5.770 |
| | $L$ | 48.866 | 9.376 | 47.158 | 10.126 | 28.202 | 7.6490 | 49.302 | 10.600 |
| Pear | $H$ | 19.844 | 5.106 | 22.768 | 6.417 | 30.166 | 8.340 | 24.098 | 7.401 |
| | $M$ | 22.517 | 6.751 | 24.929 | 7.557 | 35.561 | 9.363 | 28.177 | 8.935 |

**Table 2** (continued)

| Rotation | | $Rot\,(0,0,0)$ | | $Rot\,(0,8\pi/9,0)$ | | $Rot\,(4\pi/9,5\pi/18,0)$ | | $Rot\,(5\pi/18,7\pi/18,\pi/4)$ | |
|---|---|---|---|---|---|---|---|---|---|
| Voxelized Shape | Density | $Hau$ | $\triangle_{\text{avg}}$ | $Hau$ | $\triangle_{\text{avg}}$ | $Hau$ | $\triangle_{\text{avg}}$ | $Hau$ | $\triangle_{\text{avg}}$ |
| | $L$ | 36.494 | 9.599 | 32.807 | 10.239 | 37.450 | 11.978 | 43.269 | 11.810 |
| Penguin | $H$ | 12.457 | 3.260 | 13.849 | 3.659 | 15.749 | 4.642 | 9.655 | 2.782 |
| | $M$ | 18.138 | 5.096 | 26.235 | 5.418 | 21.815 | 6.806 | 16.144 | 4.413 |
| | $L$ | 29.349 | 9.097 | 42.796 | 11.307 | 26.877 | 9.847 | 28.685 | 7.744 |
| Santa | $H$ | 7.578 | 2.430 | 7.054 | 2.593 | 7.270 | 2.522 | 9.308 | 2.870 |
| | $M$ | 12.161 | 3.861 | 11.677 | 3.984 | 9.487 | 3.742 | 10.694 | 4.133 |
| | $L$ | 18.617 | 6.834 | 19.200 | 6.910 | 37.101 | 8.237 | 25.587 | 7.472 |
| Torus | $H$ | 19.122 | 4.391 | 15.977 | 4.679 | 21.291 | 4.644 | 23.514 | 7.612 |
| | $M$ | 21.590 | 6.058 | 25.113 | 6.357 | 25.580 | 6.322 | 30.124 | 10.163 |
| | $L$ | 37.272 | 9.554 | 33.885 | 9.874 | 40.106 | 10.173 | 35.396 | 14.364 |

point clouds depend on the digital straight segments of the slices of $\mathcal{D}$, which is information contained in the chain code.

We utilized a sample of objects to study the sequence of point clouds. We first found the voxelization at different resolutions, and then we obtained point clouds with different densities and rotations. Using the Hausdorff distance and the average geometric error, we

**Table 3** Take two different voxel objects $\mathcal{D}$ and $\mathcal{D}'$ and the rotations $\mathcal{D}\boldsymbol{\theta}$ and $\mathcal{D}'\boldsymbol{\theta}'$. The Hausdorff distance between $\mathbf{P}\,(\mathcal{D})$ and $\mathbf{P}\,(\mathcal{D}\boldsymbol{\theta})$ is upper bounded by $H_\circ$

| | $H$ density | | $M$ density | | $L$ density | |
|---|---|---|---|---|---|---|
| Voxel Shape | $H_\circ$ | $H_{\text{next}}$ | $H_\circ$ | $H_{\text{next}}$ | $H_\circ$ | $H_{\text{next}}$ |
| Blade | 25.959 | 50.226 | 34.127 | 50.402 | 51.644 | **50.513** |
| Brain | 31.075 | 57.321 | 29.673 | 57.244 | 31.556 | 55.279 |
| Bunny | 41.296 | 70.689 | 40.436 | 72.731 | 50.348 | 62.650 |
| Cow | 14.782 | 39.718 | 19.456 | 41.28 | 22.208 | 38.169 |
| Dragon | 13.283 | 51.886 | 17.214 | 50.295 | 24.109 | 51.08 |
| Hand | 17.994 | 51.988 | 19.298 | 49.774 | 20.883 | 50.354 |
| Happy | 30.244 | 39.054 | 25.914 | 38.62 | 34.757 | 34.757 |
| Heptoroid | 31.700 | 37.68 | 30.964 | 39.196 | 31.279 | 39.574 |
| Hippo | 14.828 | 39.058 | 19.634 | 38.62 | 21.607 | 34.757 |
| Horse | 36.289 | 52.816 | 43.051 | 50.925 | 55.261 | **54.010** |
| Lion | 13.398 | 47.99 | 15.372 | 46.961 | 25.583 | 45.757 |
| Lucy | 25.535 | 45.302 | 39.96 | 47.999 | 45.607 | 47.275 |
| Pear | 32.232 | 57.321 | 29.049 | 57.244 | 39.637 | 55.279 |
| Penguin | 27.810 | 50.265 | 29.220 | 63.030 | 62.613 | **45.607** |
| Santa | 32.952 | 62.154 | 43.800 | 61.683 | 62.390 | **54.860** |
| Torus | 14.434 | 38.27 | 20.728 | 38.853 | 37.451 | 50.402 |

The Hausdorff distance between $\mathbf{P}\,(\mathcal{D})$ and $\mathbf{P}\left(\mathcal{D}'\boldsymbol{\theta}\right)$ is lower bounded by $H_{\text{next}}$

**Table 4** Hausdorff distance and average geometric error between the point clouds with no rotations and the point clouds with rotations

| Rotation | | $Rot(0, 8\pi/9, 0)$ | | $Rot(4\pi/9, 5\pi/18, 0)$ | | $Rot(5\pi/18, 7\pi/18, \pi/4)$ | |
|---|---|---|---|---|---|---|---|
| Voxel Shape | Density | $Hau$ | $\triangle_{avg}$ | $Hau$ | $\triangle_{avg}$ | $Hau$ | $\triangle_{avg}$ |
| Blade | $H$ | 30.270 | 6.660 | 35.504 | 10.135 | 32.534 | 8.899 |
| | $M$ | 35.948 | 6.009 | 48.822 | 12.258 | 43.262 | 10.407 |
| | $L$ | 53.047 | 11.150 | 69.697 | 18.771 | 69.284 | 12.012 |
| Brain | $H$ | 24.521 | 6.898 | 33.149 | 5.455 | 29.727 | 5.687 |
| | $M$ | 28.641 | 8.014 | 36.097 | 6.989 | 36.040 | 6.873 |
| | $L$ | 37.287 | 10.816 | 51.381 | 10.533 | 39.625 | 9.498 |
| Bunny | $H$ | 17.642 | 3.763 | 21.795 | 5.797 | 33.653 | 9.456 |
| | $M$ | 21.222 | 6.085 | 22.127 | 7.903 | 40.085 | 12.812 |
| | $L$ | 20.872 | 6.856 | 30.965 | 8.662 | 45.266 | 12.691 |
| Cow | $H$ | 13.944 | 4.347 | 20.487 | 6.4205 | 22.000 | 7.209 |
| | $M$ | 20.681 | 5.801 | 21.739 | 7.860 | 28.890 | 9.955 |
| | $L$ | 31.356 | 9.645 | 25.186 | 9.512 | 44.971 | 14.381 |
| Dragon | $H$ | 14.120 | 4.037 | 24.347 | 7.363 | 20.091 | 6.905 |
| | $M$ | 21.728 | 5.103 | 29.440 | 9.094 | 26.141 | 9.015 |
| | $L$ | 30.720 | 8.274 | 41.604 | 12.096 | 33.875 | 10.603 |
| Hand | $H$ | 11.628 | 2.666 | 19.914 | 5.821 | 13.985 | 5.138 |
| | $M$ | 18.905 | 4.213 | 20.018 | 6.780 | 18.626 | 6.666 |
| | $L$ | 25.292 | 7.589 | 51.676 | 11.177 | 32.426 | 12.786 |
| Happy | $H$ | 12.852 | 3.717 | 13.244 | 5.403 | 15.200 | 3.931 |
| | $M$ | 19.488 | 5.396 | 17.784 | 6.548 | 18.951 | 5.398 |
| | $L$ | 32.657 | 10.672 | 21.897 | 7.585 | 24.517 | 9.315 |
| Heptoroid | $H$ | 27.798 | 8.305 | 30.279 | 9.913 | 30.474 | 10.460 |
| | $M$ | 31.562 | 10.871 | 29.882 | 10.385 | 27.303 | 5.159 |
| | $L$ | 25.468 | 9.563 | 34.696 | 13.940 | 27.384 | 8.720 |
| Hippo | $H$ | 19.604 | 5.226 | 14.098 | 5.811 | 20.156 | 8.444 |
| | $M$ | 19.277 | 6.451 | 21.610 | 8.193 | 26.774 | 9.762 |
| | $L$ | 21.745 | 9.101 | 29.358 | 11.696 | 31.515 | 14.148 |
| Horse | $H$ | 17.245 | 5.210 | 27.273 | 7.526 | 18.904 | 7.437 |
| | $M$ | 19.837 | 6.763 | 27.882 | 7.227 | 21.191 | 6.241 |
| | $L$ | 24.265 | 9.396 | 36.640 | 12.727 | 26.050 | 10.167 |
| Lion | $H$ | 12.228 | 3.643 | 16.928 | 4.476 | 17.540 | 5.906 |
| | $M$ | 16.844 | 5.281 | 20.620 | 6.103 | 22.151 | 7.946 |
| | $L$ | 24.052 | 8.464 | 27.702 | 9.290 | 36.090 | 12.421 |
| Lucy | $H$ | 13.145 | 3.166 | 18.687 | 5.962 | 21.678 | 5.829 |
| | $M$ | 17.350 | 5.597 | 27.257 | 7.765 | 29.175 | 7.673 |
| | $L$ | 29.614 | 8.785 | 37.134 | 11.132 | 37.819 | 12.103 |
| Pear | $H$ | 19.257 | 6.519 | 30.561 | 11.282 | 21.728 | 8.444 |
| | $M$ | 20.835 | 7.543 | 35.345 | 12.501 | 23.156 | 9.132 |
| | $L$ | 29.977 | 10.600 | 40.926 | 14.227 | 31.527 | 12.066 |

**Table 4** (continued)

| Rotation | | $Rot(0, 8\pi/9, 0)$ | | $Rot(4\pi/9, 5\pi/18, 0)$ | | $Rot(5\pi/18, 7\pi/18, \pi/4)$ | |
|---|---|---|---|---|---|---|---|
| Voxel Shape | Density | $Hau$ | $\triangle_{avg}$ | $Hau$ | $\triangle_{avg}$ | $Hau$ | $\triangle_{avg}$ |
| Penguin | $H$ | 13.511 | 4.788 | 14.698 | 5.073 | 13.565 | 3.122 |
| | $M$ | 16.556 | 7.337 | 18.015 | 7.284 | 16.630 | 4.645 |
| | $L$ | 38.398 | 14.824 | 30.220 | 11.015 | 27.693 | 8.830 |
| Santa | $H$ | 7.800 | 2.710 | 10.764 | 3.568 | 8.113 | 3.076 |
| | $M$ | 13.995 | 4.392 | 11.054 | 4.556 | 15.702 | 5.336 |
| | $L$ | 20.551 | 8.365 | 36.813 | 9.543 | 28.346 | 9.014 |
| Torus | $H$ | 15.634 | 4.114 | 17.929 | 3.917 | 24.679 | 7.371 |
| | $M$ | 19.430 | 5.274 | 25.932 | 4.995 | 25.737 | 9.830 |
| | $L$ | 36.923 | 7.606 | 27.836 | 5.941 | 29.514 | 14.468 |

proved that the point clouds are invariant under rotation. We showed that visually and quantitatively, using the Hausdorff distance, the shape of the original object is preserved by the point clouds.

It is paramount to emphasize that the point clouds we used in the comparisons are only a sample because the sequence of point clouds allows us to define point clouds with different densities.

Although we found a consistent method to simplify the shapes of 3D objects, the distribution of key points may still not be optimal. Future work could focus on the distribution of points so that the Hausdorff distance is maintained or even improved.

**Funding** Osvaldo A. Tapia-Dueñas was partially supported by CONACyT, CVU 781156. Hermilo Sánchez-Cruz was partially supported by Universidad Autónoma de Aguascalientes, grant PII22-5. Hiram H. López was partially supported by an AMS–Simons Travel Grant.

## Declarations

**Conflict of Interests** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References

1. Cao C, Preda M, Zaharia T (2019) 3d point cloud compression: a survey. In: The 24th international conference on 3d web technology. Web3d ’19, pp 1–9, Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3329714.3338130
2. CGAL Computational Geometry Algorithms Library. http://www.cgal.org/
3. Chen Z, Liu L (2021) Navigable space construction from sparse noisy point clouds. IEEE Robot Autom Lett 6(3):4720–4727. https://doi.org/10.1109/LRA.2021.3068939
4. Chen S, Tian D, Feng C, Vetro A, Kovačević J (2017) Fast resampling of three-dimensional point clouds via graphs. IEEE Trans Signal Process 66(3):666–681
5. Chen S, Tian D, Feng C, Vetro A, Kovačević J (2018) Fast resampling of three-dimensional point clouds via graphs. IEEE Trans Signal Process 66(3):666–681. https://doi.org/10.1109/TSP.2017.2771730

6. El Sayed AR, El Chakik A, Alabboud H, Yassine A (2019) An efficient simplification method for point cloud based on salient regions detection. RAIRO-Oper Res 53(2):487–504
7. Fan H, Yang Y, Kankanhalli M (2021) Point 4d transformer networks for spatio-temporal modeling in point cloud videos. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 14204–14213
8. Fan H, Yu X, Yang Y, Kankanhalli M (2021) Deep hierarchical representation of point cloud videos via spatio-temporal decomposition. IEEE Trans Pattern Anal Mach Intell
9. Freeman H (1961) On the encoding of arbitrary geometric configurations. IRE Trans Electron Comput EC-10(2):260–268. https://doi.org/10.1109/TEC.1961.5219197
10. Garcia DC, Fonseca TA, Ferreira RU, de Queiroz RL (2020) Geometry coding for dynamic voxelized point clouds using octrees and multiple contexts. IEEE Trans Image Process 29:313–322. https://doi.org/10.1109/TIP.2019.2931466
11. Golla T, Kneiphof T, Kuhlmann H, Weinmann M, Klein R (2020) Temporal upsampling of point cloud sequences by optimal transport for plant growth visualization. In: Computer Graphics Forum, vol 39, pp 167–179. Wiley Online Library
12. Good CD, Johnsrude IS, Ashburner J, Henson RNA, Friston KJ, Frackowiak RSJ (2001) A voxel-based morphometric study of ageing in 465 normal adult human brains. NeuroImage 14(1):21–36. https://doi.org/10.1006/nimg.2001.0786
13. Gu S, Hou J, Zeng H, Yuan H, Ma K-K (2019) 3d point cloud attribute compression using geometry-guided sparse representation. IEEE Trans Image Process PP:1–1. https://doi.org/10.1109/TIP.2019.2936738
14. Guo Y, Sohel F, Bennamoun M, Wan J, Lu M (2015) A novel local surface feature for 3d object recognition under clutter and occlusion. Inf Sci 293:196–213. https://doi.org/10.1016/j.ins.2014.09.015
15. Huang H, Li D, Zhang H, Ascher U, Cohen-Or D (2009) Consolidation of unorganized point clouds for surface reconstruction. ACM Trans Graph (TOG) 28(5):1–7
16. Huang T, Liu Y (2019) 3d point cloud geometry compression on deep learning. In: MM ’19: The 27Th ACM international conference on multimedia, pp 890–898. https://doi.org/10.1145/3343031.3351061
17. Huiyan H, Xie H, Fusheng S, Chunyan H (2015) Point cloud simplification with preserved edge based on normal vector. Optik - Int J Light Electron Opt 126(19):2157–2162. https://doi.org/10.1016/j.ijleo.2015.05.092
18. Ji C, Li Y, Fan J, Lan S (2019) A novel simplification method for 3d geometric point cloud based on the importance of point. IEEE Access 7:129029–129042
19. Junkun Q, Wei H, Zongming G (2019) Feature preserving and uniformity-controllable point cloud simplification on graph. In: 2019 IEEE International conference on multimedia and expo (ICME), pp 284–289. https://doi.org/10.1109/ICME.2019.00057
20. Kansal S, Madan J, Singh A (2013) A systematic approach for cad model generation of hole features from point cloud data. In: 2013 3rd IEEE International Advance Computing Conference (IACC), pp 1385–1393. IEEE
21. Kehl W, Milletari F, Tombari F, Ilic S, Navab N (2016) Deep learning of local rgb-d patches for 3d object detection and 6d pose estimation. In: Leibe B, Matas J, Sebe N, Welling M (eds) Computer vision – ECCV 2016, pp 205–220. Springer, Cham
22. Kim J, Im J, Rhyu S, Kim K (2020) 3d motion estimation and compensation method for video-based point cloud compression. IEEE Access 8:83538–83547
23. Klette R, Rosenfeld A (2004) Digital geometry: geometric methods for digital picture analysis. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA
24. Krüsi P, Furgale P, Bosse M, Siegwart R (2017) Driving on point clouds: motion planning, trajectory optimization, and terrain assessment in generic nonplanar environments. J Field Robot 34(5):940–984
25. Leal E, Sanchez-Torres G, Branch-Bedoya JW, Abad F, Leal N (2021) A saliency-based sparse representation method for point cloud simplification. Sensors 21(13):4279
26. Lee M-y, Lee S-h, Jung K-d, Lee S-h, Kwon S-c (2021) A novel preprocessing method for dynamic point-cloud compression. Appl Sci 11(13):5941
27. Liu H, Cong Y, Yang C, Tang Y (2019) Efficient 3d object recognition via geometric information preservation. Pattern Recogn 92:135–145. https://doi.org/10.1016/j.patcog.2019.03.025
28. Liu Z, Li Q, Chen X, Wu C, Ishihara S, Li J, Ji Y (2021) Point cloud video streaming: challenges and solutions. IEEE Netw 35(5):202–209. https://doi.org/10.1109/MNET.101.2000364
29. Liu X, Yan M, Bohg J (2019) Meteornet: deep learning on dynamic 3d point cloud sequences. In: Proceedings of the IEEE/CVF international conference on computer vision (ICCV)

30. Liu YK, Žalik B (2005) An efficient chain code with huffman coding. Pattern Recogn 38(4):553–557. https://doi.org/10.1016/j.patcog.2004.08.017
31. Min P (Unknown Month 2004) Binvox. http://www.patrickmin.com/binvox. Accessed: 18 Feb 2022 and 09 Aug 2022
32. Muhammad S, Joono C, Y K, H C (2019) Fractal bubble algorithm for simplification of 3d point cloud data. J Intell and Fuzzy Syst 37:7815–7830. https://doi.org/10.3233/jifs-182742
33. Nguyen CHP, Choi Y (2018) Comparison of point cloud data and 3d cad data for on-site dimensional inspection of industrial plant piping systems. Autom Constr 91:44–52
34. Ning X, Wang Y, Meng W, Zhang X (2016) Optimized shape semantic graph representation for object understanding and recognition in point clouds. Opt Eng 55(10):1–14. https://doi.org/10.1117/1.OE.55.10.103111
35. Nooruddin FS, Turk G (2003) Simplification and repair of polygonal models using volumetric techniques. IEEE Trans Vis Comput Graph 9(2):191–205
36. Pauly M, Gross M, Kobbelt LP (2002) Efficient simplification of point-sampled surfaces. In: IEEE Visualization, 2002. VIS 2002, pp 163–170. https://doi.org/10.1109/VISUAL.2002.1183771
37. Sánchez-Cruz H, Tapia-Dueñas OA, Cuevas F (2019) Polygonal approximation using a multiresolution method and a context-free grammar. In: Carrasco-Ochoa JA, Martínez-Trinidad JF, Olvera-López JA, Salas J (eds) Pattern Recognition, pp 261–270. Springer, Cham. https://doi.org/10.1007/978-3-030-21077-9_24
38. Shah SAA, Bennamoun M, Boussaid F (2016) A novel feature representation for automatic 3d object recognition in cluttered scenes. Neurocomputing 205:1–15. https://doi.org/10.1016/j.neucom.2015.11.019
39. Shah SAA, Bennamoun M, Boussaid F (2017) Keypoints-based surface representation for 3d modeling and 3d object recognition. Pattern Recogn 64:29–38. https://doi.org/10.1016/j.patcog.2016.10.028
40. Shiquan Q, Kun Z, Kai G (2019) Algorithm for point cloud compressing based on geometrical features. Int J Performability Eng 15:782
41. Siva S, Nahman Z, Zhang H (2020) Voxel-based representation learning for place recognition based on 3d point clouds. In: 2020 IEEE/RSJ International conference on intelligent robots and systems (IROS), pp 8351–8357. https://doi.org/10.1109/IROS45743.2020.9340992
42. Song H, Feng H-Y (2007) A point cloud simplification algorithm for mechanical part inspection. International Federation for Information Processing Digital Library; Information Technology For Balanced Manufacturing Systems; 220. https://doi.org/10.1007/978-0-387-36594-7_49
43. Song H, Feng H-Y (2007) Point-cloud simplification with bounded geometric deviations. Int J Comput Appl Technol 30(4):236–244. https://doi.org/10.1504/IJCAT.2007.017235
44. Song H, Feng H-Y (2009) A progressive point cloud simplification algorithm with preserved sharp edge data. Int J Advan Manuf Technol 45:583–592. https://doi.org/10.1007/s00170-009-1980-4
45. Szeliski R (2010) Computer vision: algorithms and applications, 1st edn. Springer, Berlin
46. Taha AA, Hanbury A (2015) An efficient algorithm for calculating the exact hausdorff distance. IEEE Trans Pattern Anal Mach Intell 37(11):2153–2163. https://doi.org/10.1109/TPAMI.2015.2408351
47. Tangelder JWH, Veltkamp RC (2004) A survey of content based 3d shape retrieval methods. In: Proceedings Shape Modeling Applications, 2004, pp 145–156. https://doi.org/10.1109/SMI.2004.1314502
48. Tapia-Dueñas OA, Sánchez-Cruz H (2021) Context-free grammars to detect straight segments and a novel polygonal approximation method. Signal Process Image Commun 116080:91. https://doi.org/10.1016/j.image.2020.116080
49. Thanou D, Chou PA, Frossard P (2016) Graph-based compression of dynamic 3d point cloud sequences. IEEE Trans Image Process 25(4):1765–1778. https://doi.org/10.1109/TIP.2016.2529506
50. Toledo L, De Gyves O, Rudomín I (2014) Hierarchical level of detail for varied animated crowds. Vis Comput 30(6):949–961
51. Vogiatzis G, Hernández C (2011) Video-based, real-time multi-view stereo. Image Vis Comput 29(7):434–441. https://doi.org/10.1016/j.imavis.2011.01.006
52. Wang L, Xu Y, Li Y (2018) A voxel-based 3d building detection algorithm for airborne lidar point clouds. J Indian Soc Remote Sens 47:349–358
53. Wei J, Xu M, Xiu H (2020) A point clouds fast thinning algorithm based on sample point spatial neighborhood. J Inform Process Syst 16(3):688–698
54. Yang Y, Li M, Ma X (2020) An advanced vehicle body part inspection scheme based on scattered point cloud data. Appl Sci 10(15):5379
55. Yang Y, Zhuang Y, Pan Y (2021) Multiple knowledge representation for big data artificial intelligence: framework, applications, and case studies. Front Inform Technol Electron Eng 22(12):1551–1558

56. Zou Y, Wang X, Zhang T, Liang B, Song J, Liu H (2018) Broph: An efficient and compact binary descriptor for 3d point clouds. Pattern Recogn 76:522–536. https://doi.org/10.1016/j.patcog.2017.11.029
57. Zuquete Guarato A, Quinsat Y, Mehdi-Souzani C, Lartigue C, Sura E (2018) Conversion of 3D scanned point cloud into a voxel-based representation for crankshaft mass balancing. International Journal of Advanced Manufacturing Technology. https://doi.org/10.1007/s00170-017-1319-5

# 7 Estrategia de código de cadena para almacenamiento y transferencia sin pérdidas de datos médicos binarios segmentados

A continuación se presenta el artículo publicado, cuyo título es: **Chain code strategy for lossless storage and transfer of segmented binary medical data**

Expert Systems With Applications 216 (2023) 119449

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

# Chain code strategy for lossless storage and transfer of segmented binary medical data

Erdoğan Aldemir [a,*], Osvaldo Arturo Tapia Dueñas [b], Ali Emre Kavur [a], Gulay Tohumoglu [c], Hermilo Sánchez-Cruz [b], Mustafa Alper Selver [c]

[a] *The Graduate School of Natural and Applied Sciences., Dokuz Eylül University, Tinaztepe Campus. No: 22, Izmir, Turkey*
[b] *Departamento de Ciencias de la Computación. Centro de Ciencias Básicas. Universidad Autónoma de Aguascalientes, Aguascalientes 20131, Ags., México*
[c] *Electrical and Electronics Engineering. Dept, Dokuz Eylül University., Tınaztepe Camp. No:22, Izmir, Turkey*

## ARTICLE INFO

*Keywords:*
DICOM
Telemedicine
Lossless compression
Chain code
Visualization

## ABSTRACT

Obtaining a 3D medical visualization is a tedious process requiring several processing steps (such as segmentation) and assigning various rendering parameters (such as color and opacity). Current systems use video/image exporting or snapshots to save results. Such vendor-dependent tools not only prevent the possibility of further interactions but also creates additional large-size data that is problematic to store in PACS over time and hard to transfer for teleradiology applications. To overcome, alternative strategies propose a representation of the visualizations, which only store segmentation masks that contains the binary form of segmented data. Unfortunately, existing compression methods are limited to effectively compress the volumetric data. In this study, lossless storage of binary segmented data is effectively performed by two newly-proposed chain code approaches. Particularly-two novel contributions are presented: 1. The dictionary of normalized angle difference is improved as a new chain symbol coding procedure, namely normalized angle difference, by adding new symbols to the dictionary aiming to generate a low-entropy symbol sequence for medical volumes. 2. A new volumetric approach that utilizes 26 symbols to encode volumetric data is developed. Each slice is visited, and the contour of the segmented object is codified such that eight different vectors for each slice (pointing to one of the four faces of each voxel, plus four towards one of its edges) are obtained. The developed methods are tested on diverse volumetric segmented data and compared to existing standards. It is shown that the proposed methods outperform well-established techniques.

## 1. Introduction

As medical imaging modalities advance to provide more detailed acquisitions, the amount of generated data is increasing drastically (Andriole et al., 2011; Gunderman and Chou, 2016; Norbash et al., 2014). This requires new tools and techniques for efficient data storage and fast and reliable distribution of data over teleradiology networks. The latter's importance is recognized, especially during the pandemic, when radiologists mostly worked remotely. Such conditions also require the remote collaboration of other clinicians with the radiologist(s) for diagnostic planning. Especially in complex cases, high-quality communication is necessary for handling multi-dimensional medical data, and to provide that, effective compression and distribution strategies are needed (Aldemir et al., 2020; Fischer et al., 2010, 2015).

3D visualization became a vital tool for analyzing tomographic medical data. Unfortunately, the generation of a clinically interpretable rendering requires a tedious procedure including several processing steps (such as segmentation) and adjustment of various parameters (i.e., optical and appearance-related parameters such as color and opacity) (Fischer et al., 2015). Moreover, reaching a consensus usually requires an iterative optimization process, during which the segmentation methods are executed several times prior to reaching desired

---

Peer review under responsibility of Submissions with the production note 'Please add the Reproducibility Badge for this item' the Badge and the following footnote to be added:The code (and data) in this article has been certified as Reproducible by the CodeOcean: https://codeocean.com. More information on the Reproducibility Badge Initiative is available at https://www.elsevier.com/physicalsciencesandengineering/computerscience/journals.

[*] Corresponding author.

*E-mail addresses:* erdogan.aldemir@deu.edu.tr (E. Aldemir), osvaldo.tapia@edu.uaa.mx (O. Arturo Tapia Dueñas), gulay.tohumoglu@deu.edu.tr (G. Tohumoglu), hermilo.sanchez@edu.uaa.mx (H. Sánchez-Cruz), alper.selver@deu.edu.tr (M. Alper Selver).

https://doi.org/10.1016/j.eswa.2022.119449
Received 21 March 2022; Received in revised form 21 November 2022; Accepted 16 December 2022

information (Fischer et al., 2010). Accordingly, remote-access multi-user visualization platforms are needed, and compression is a critical element of such tools (Aldemir et al., 2020). Unfortunately, existing general-purpose compression methods rarely consider unique properties of medical volumes, particularly inter-slice coherence and intra-slice information (Sanchez, Nasiopoulos, and Abugharbieh, 2008) about organ/tissue morphology (Bradley, 2008). Besides, volume visualization of the medical data capabilities had been raised using JPEG2000 in the study by (Krishnan et al., 2006), and additionally, in a recent study (Chen et al., 2022), both inter-and intra-slice coherence of gray level volumetric data is exploited to uncover relevant redundancy. This study achieved a considerable compression ratio compared to HEVC and JPEG-LS standards but worked on the gray level. What is more, the majority of the standards designed for medical data are intended to process the original (i.e., gray-scale) data. Thus, a significant redundancy of medical data could not be revealed by the common standards. On the other hand, 3D medical visualization focuses on the representation of organ(s)/tissue(s) of interest segmented from the volume. Accordingly, compression of the binary mask representing the segmented voxels is enough for reproduction (Aldemir, Tohumoglu, and Selver, 2019). Moreover, the allocation preserves significant storage space and enables efficient data transfer (Karimi et al., 2016) for teleradiology (Dionisio et al., 2000).

In the light of the discussions mentioned above, the main contributions of this study can be listed as follows:

1. Since the size of the volumetric medical data can be significantly large for transmission and storage even in binary form, effective lossless compression strategies are still needed. Furthermore, since the medical image data is initially in a structured grid form, the segmented objects can be represented effectively by their outer bounds (i.e., limits). These bounds correspond to a structure having a single-pixel thickness and separate the object's interior from the background. Thus, a 2D chain code strategy is developed for compressing the boundaries of the segmented data, and it is shown that a significant increase in compression ratio (CR) can be achieved compared to well-established compression strategies.
2. Volumetric processing of the segmented data instead of compressing a series of 2D planes creates another possibility of reaching high CR levels. To enable this, the DICOM committee is actively working on translating 2D Grayscale Softcopy Presentation States (Text, 1999) into multi-dimensional cases (Pianykh, 2012). Although the Surface Segmentation Information Object Definition (IOD) specifies a polygonal representation of a 3D surface (DICOM, 2006; DICOM Standards Committee, 2006), it is designed for creating object meshes for surgical planning rather than storage, compression, and transmission. Thus, a 3D compression strategy is carried out, which consists of coding the surface through a helical path. Because our binary 3D object is organized by 2D planes (slices of voxels), the helical path is achieved by visiting the contour of each of them to make a change of direction between slices then. Each movement is encoded with the Freeman chain code of 26 directions (F26, for short) as a set in the study by (Tapia-Dueñas et al., 2018). Finally, an entropy encoder based on the context-mixing algorithm is applied.
3. Extensive simulations show that the compression methods developed above have varying performances in different clinical cases (Kavur et al., 2020, 2021; Selver and Emre Kavur, 2016). This is because when the segmented objects are represented only by their shell, the utilization of inter-frame information becomes more sensitive to morphology. To determine the most appropriate strategies, we have performed further experiments on segmented objects with different outer bound characteristics, starting from very compact ones (such as a kidney) to those with high entropy (such as vascular trees). The results are analyzed in detail together with characteristic parameters such as modality and data structure.

Considering all three contributions mentioned above, two novel compression strategies based on 2D and volumetric chain rules are presented for effectively storing and transmitting 3D segmented volumes on healthcare networks. Information technologies, including cloud-based medical data networks and DICOM web standards, have become a significant part of healthcare, especially during the COVID-19 pandemic (Chung and Park, 2019; Hassanalieragh et al., 2015). High-speed transmission could provide an effective exchange of medical data among referring clinicians and enable carry-on devices to be committed in a collaborative, real-time consensus (Andriole et al., 2011; Parikh et al., 2018; Sharma, Sood, and Puthooran, 2020; Thrall, 2007). Considering telemedicine networks, managing large-scale data such as CT and MRI image series is challenging in terms of the retrieval time-bandwidth requirements. Region-of-interest (ROI) based algorithms such as (Kaur and Wasson, 2015) divide the medical image into parts -which are the diagnostically important, non-important, and background regions- for compression. The wavelet algorithms designed using ROI (Bairagi and Sapkal, 2013) and intra-band prediction (Bruylants, Munteanu, and Schelkens, 2015) provide significant compression performance for gray-level medical data. Inter-slice correlation between volumetric slices has been considered by the algorithm (Bruylants et al., 2015) for achieving more compression efficiency. An automated ROI-based hybrid compression system applies reversible and irreversible approaches for regions considered salient details and remaining regions, respectively (Devadoss and Sankaragomathi, 2019). Currently, a clustering-based compression scheme, which involves modified block truncation coding, is used for cloud-based telemedicine (Hsu, 2017). Another method raising the potential of remote healthcare services for progressive retrieval of medical data over telemedicine networks is proposed. It is reported to achieve satisfactory performance in diagnostically acceptable information loss (Parikh et al., 2018).

Despite the algorithms mentioned above that reach notable performances, their use on segmented (i.e., binary) medical volume data is limited due to the following reasons:

1) The majority of these algorithms are general-purpose gray-level methods; in other words, they are not specifically designed for binary images (Liu et al., 2017). Thus, the current solutions cannot adequately uncover redundancy in binary medical data.
2) Even though the algorithms consider an ROI and reveal a significant redundancy, these methods are not specifically designed to consider the morphology of the organs.
3) Although irreversible (lossy) algorithms are primarily employed by streaming technologies as a tool for diagnosis and referral (Liu et al., 2017), it is pointed out that the data degradation originating from irreversible algorithms must be considered seriously (Kim et al., 2011).

Consequently, up-to-date standards such as JBIG, JBIG2, ABIC, and PNG are general-purpose compression algorithms. Therefore, the amount of redundancy they reveal in medical data remains limited. To overcome these limitations, binary lossless compression strategies based on two- and three-dimensional coding of the boundary of medical data, i.e., organs or segmented tissues, have been proposed. Boundary pixels and voxels are encoded by the chain rules, considering the morphological structure and entropy of the organs. In addition, by operating the chain rule as a parameter, employing different rules is ensured, and thus, the compression has been performed in a parametric form. By the parametrization of the system, the capability of integrating the proposed system into a large telemedicine network is increased, and the system can be optimized to uncover more redundant data.

## 2. Material and methods

### 2.1. Datasets and characteristic properties

The data sets for testing the compression methods are selected from various modalities and anatomical objects with diverse spatial structures so that the performance of the techniques can be evaluated for a wide range of applications. The datasets can be roughly categorized into two groups: solid abdominal organs (from CT and MR) and vascular structures (from CT-angiography). Each of the organs of interest is segmented manually by expert physicians, and binary images are created, see Fig. 1 (Aldemir et al., 2020).

Abdominal CT data sets were acquired after the contrast agent injection at the portal phase with the spiral scanning option. This property benefits 3D visualization because it reduces the misregistration between slices and increases spatial coherence (Kavur et al., 2020). Volumetric visualization is essential for compression once the algorithm uses 3D scanning forms. Twenty data sets (CT series) are collected from the PACS of Dokuz Eylül University Radiology Department randomly, and the liver is manually segmented for each image slice individually. The image series have a 3–3.2 mm interslice distance (ISD) corresponding to a slice number around 90 (minimum 77, maximum 105 slices).

The second group of abdominal studies includes two sequences (i.e., T2-SPIR and T1-DUAL) of MRI image series obtained from 20 patients. Each image has a resolution of $256 \times 256$. The abdominal organs are extracted, including MR-T1 and MR-T2 liver (Kavur et al., 2021). For T2-SPIR, the ISD value changes between 7.7 mm and 9 mm and has an average value of 8.6 mm. Moreover, the x-y spacing in this sequence varies between 1.63 mm and 1.89 mm, with an average of 1.53 mm. The number of slices for the T2-SPIR sequence is 26 as a minimum, 36 as a maximum, and 30 as the average. On the other hand, T1-DUAL sequences include two different image series: in-phase and out-phase, indicating the fat suppression mechanism used during acquisition. Each series has the same x-y spacing, ISD, and number of slices. ISD varies between 5.5 mm and 9 mm, with an average of 7.84 mm. The x-y spacing value is between 1.44 mm and 1.89 mm, and the average value equals 1.61 mm. The average number of slices is 32.8 (min. 26, max. 50 slices). The abdominal CT/MR dataset is already available online at: https://zenodo.org/record/3431873. The third dataset consists of segmented objects in the second category: vascular trees, see Fig. 2.

The proposed compression methods are tested in 19 CT Angio (CTA) data sets from 19 different patients having Abdominal Aortic Aneurysms (AAA) using four different modalities (Fig. 1 (b)). The first data set is acquired by a 16-row detector CT scanner with 3.2 mm ISD from PACS located at the Radiology Department of DEU. The remaining data sets were selected among acquisitions, which reflect the challenges of daily clinical practice. Six data sets (i.e., data sets 2–7) were acquired with a 320-row detector with 3.0 mm ISD, and six data sets (i.e., data sets 8–13) were acquired with a 64-row detector with 5.0 mm ST obtained from PACS of Gülhane Faculty of Medicine (GATA). The performance of the proposed compression methods for 3.0 and 5.0 mm ISD values are chosen to represent the efficiency of the compression methods as these values are the most frequent ISD values used in clinical routine. Therefore, six data sets (i.e., data sets 14–19) were acquired with the 320-row detector CT scanner with 0.8 mm ISD. In total, 3649 DICOM images, which have 512x512 pixel resolution, are segmented manually to extract the aorta and the main vessels departing it (Selver and Emre Kavur, 2016).

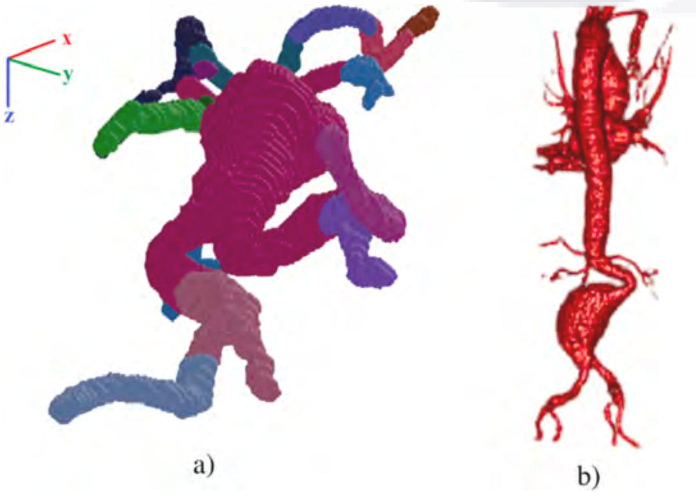


**Fig. 1.** (a) 3D regions defined when visiting the contours of each slice by helical paths, (b) A vascular tree showing abdominal aortic aneurysm (AAA).

### 2.2. Compression methods

Binary and gray-level medical images differ concerning structural characteristics such as texture and entropy. These are among the features that primarily characterize the compressibility of image data. Although there do exist specifically designed techniques that achieve noteworthy compression performance for medical (object-based 3D-RLE (Aldemir et al., 2019)) and bi-level (ABIC algorithm (Marks, 1998)) images, there still exists considerable redundant data in the medical images that could be uncovered.

#### 2.2.1. A new chain code strategy: Modified normalized angle difference (MNAD)

Although the chain code is a technique that is regularly used in pattern recognition; several chain code-based pipelines have been suggested for compression of the binary 2D images and 3D volumetric images in recent years (Mao et al., 2015; Martínez, Bribiesca, and Guzmán, 2017; Sánchez-Cruz, López-Valdez, and Cuevas, 2014; Tapia-Dueñas et al., 2018; Žalik et al., 2016, 2019). These pipelines harness traditional chain rules such as Freeman F4/F8, three orthogonal chain codes (3OT) (Sánchez-Cruz, Bribiesca, and Rodríguez-Dagnino, 2007), and normalized angle difference (NAD) chain code (Žalik and Lukač, 2014). The rule is one of the most robust and up-to-date chain rules. It is based on the angle difference between the present and subsequent movements of the boundary pixels. The alphabet of the NAD rule is formed by four elements $\{0,1,2,3\}$. The symbols contain a combination of the alphabet elements for every possible pixel movement. The angles between the movement greater than the angle of $45^\circ$ in absolute value are indicated by the $\{3\}$ prefix elements (Žalik and Lukač, 2014). Furthermore, they are generally experienced in computer graphics and basic bi-level natural images and objects. These points address the looming power of chain-code-based schemes for bi-level medical images.

Motivated by this fact, a hypothesis -which holds that the chain rule can be developed to reveal redundancy in medical data consisting of segmented organs- is suggested. In this context, an improved version of the NAD chain rule, namely MNAD, is developed by modifying the symbols created from elements in the dictionary with the shorter ones. This idea allows shape (segmented organs) boundaries to be represented by shorter and lower entropy code chains. The MNAD alphabet consists of four elements, the same as the NAD rule. Two symbols $\{3,33\}$ are used as the flags for movements distinct from the NAD. The $\{3\}$ and $\{33\}$ are employed as the prefixes for angles of $90^\circ$ in absolute and $135^\circ$, and for angles of $-135^\circ$ and $180^\circ$, respectively. The NAD rule uses 3-length symbols for the angle of $90^\circ$ in absolute and $135^\circ$ in absolute, and for the angle of $180^\circ$ (Žalik and Lukač 2014), while MNAD uses 3-length symbols only for movements in the angle of $-135^\circ$ and $180^\circ$ only, see Table 1. Considering the movement throughout the angles of $180^\circ$ (back-to-previous pixel) is the rarely seen movement in organ shapes, it is clear that MNAD will give much better results than the NAD rule, especially in low entropy organ shapes. These outcomes can be seen in Table 3 in the result section.

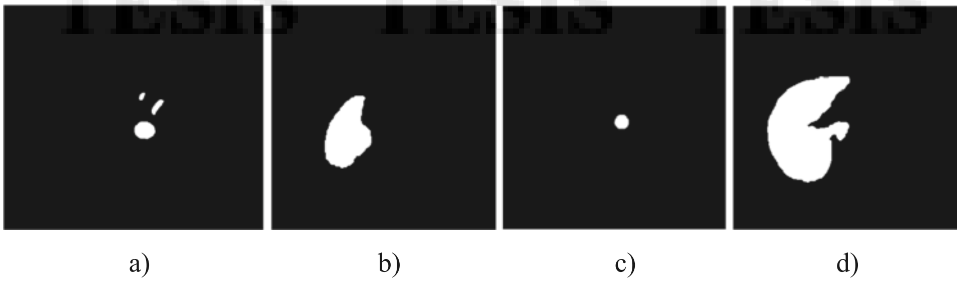


**Fig. 2.** Sample slices from volumetric tree of a-b) AAA and c-d)liver.

**Table 1**
The MNAD dictionary for angle difference.

| Angle Difference | 0° | 45° | −4 45° | 90° | −90° | 135° | −135° | 180° |
|---|---|---|---|---|---|---|---|---|
| MNAD symbols | 0 | 1 | 2 | 30 | 31 | 32 | 330 | 331 |

To implement improved chain rule MNAD, we proposed a chain-code-based compression pipeline (CS) utilizing the MNAD combined with the entropy encoders for medical images, see Fig. 3. Furthermore, the CS is designed in a parametric form that is available for different chain rules called *r* parameter (MNAD, NAD, etc..). Thus, the pipeline is implemented for various medical images having different morphological structures and levels of entropy, Fig. 3.

The chain code block is mapping the raw medical image, $I(x,y,z)$, into $s_m$ chain symbol sequences. The chain rule (r), which is the scan rule for boundary pixels of the binary medical image, and the entropy coder can also be entered parametrically. The binary compression pipeline employed the aforementioned standard chain rules (r) as well as modern chain rules such as NAD, 3OT, and MNAD. The optimal candidate of the chain rule is the one that aims to reduce the Shannon entropy (Henriques et al. 2013) of $S(m)$.

$$H(S_m) = \sum_{S_m \in I} P_m \log \frac{1}{p_m}$$

where $I$ is a 2D slice of the volumetric image, $S_m$ chain code symbols ($S_m \in \{0,1,2,30,31,32,330,331\}$) produced from slices with distributed $P(S_m = m) = p_m$. And then, chain symbols $S(m)$ produced by the chain block are processed to reveal redundancy in the reoccurred succeeding symbols by run-length encoding. The run-length block generates $r_c, v_c$ pairs indicate the number of reoccurrences and the value of symbols, respectively. The $r_c, v_c$ sequences symbols (Sm) are converted to a symbol sequence in a lower entropy form using RLE. Finally, the pipeline exploits the Huffman/Context-mixing algorithms as entropy encoders and brings out the compressed data, i.e., $C(n)$ bitstream. An illustrative scheme of the implementation of the MNAD is given in Fig. 4. The MNAD coding takes advantage of the low frequency of 180° turns on the relative direction of chain trace in the boundary pixels. The character $\{0\}$ is used as no turn, $\{1\}$ and $\{2\}$ symbols are used as the angle of 45° in absolute turns, and element of $\{3\}$ is used as an indicator for more than the angle 45° turns. The developed method takes advantage of the same properties of chain code as the NAD and has the same character dictionary.

### *2.2.2. 3-D chain code strategy*

The second approach used for binary medical image compression is the application of a context-mixing algorithm to the F26 chain code (CV). To encode our 3D medical image, we initially consider the 26 symbols of F26, which are the labels for the 26 directions that a cube has, because of its six faces, 12 edges, and eight vertices of its neighbors, in a 3 × 3 × 3 array, see Fig. 5.

If $c_1$ are the coordinates of the voxel $v_1$, $c_2$ the coordinates of the voxel $v_2$, and $b = c_2 - c_1$, then $v_1$ is in the vicinity of $v_2$ if and only if $b \in B = \{(i,j,k) | -1 \le i,j,k \le 1\} \backslash \{(0,0,0)\}$. The set of vectors B is called the *grid basis*. We give symbols to each of the elements ($b$'s) of the grid base B as shown in Table 2. So, the alphabet of F26 chain code is the set of symbols **{a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z}** used for encoding 3D object.

Part of our strategy is first to visit each slice and codify its contour, from a voxel $v_1(s)$ to a voxel $v_n(s)$ of a contour of $n$ voxels in an $s$ slice, so that we can use eight different vectors for each visited slice since the vectors point to one of the four faces of each voxel, plus four towards one of its edges. Then, eight symbols are required. On the other hand, once the contour is visited, the next step is to go to the next slice. Therefore, we need vectors that point to any of its four edges or four vertices, plus one more symbol corresponding to the upper face. Then, nine more symbols are used. Therefore, only 17 symbols from the 26 are required at most. We can define different connected regions by visiting the 3D image through a helical path.

As an example, consider an object given in Fig. 6. The object is composed of two slices. For an easy explanation, in Fig. 7(a), the left part represents the bottom slice, whereas the right part represents the top slice. On the other hand, in Fig. 7(b), the dark cells represent the contours of the connected regions in each slice.

Suppose we start to visit the bottom slice from the voxel with a dark circle, with coordinates (2,5,1), in the direction the arrow indicates, and encode the contour with F26 following symbols assignment of Table X. Once the last voxel of this contour has reached,$v_n(1)$ (black voxel), with coordinates (3,5,1), we look for the contour voxel of the next slice that is

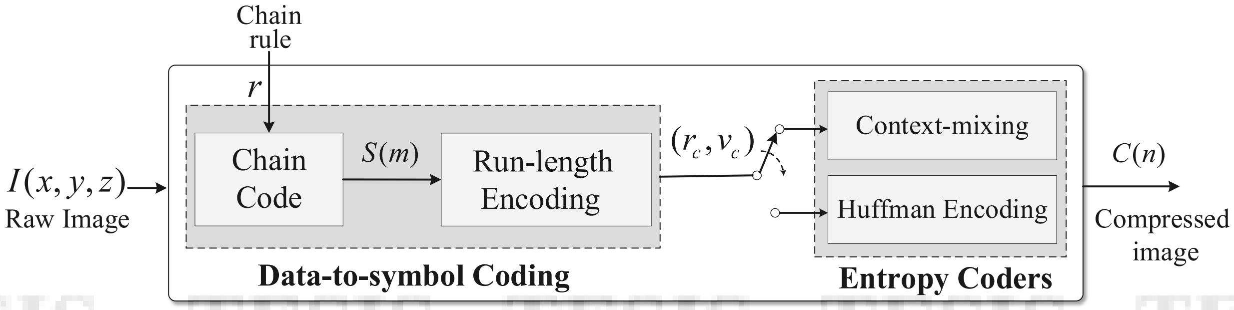


**Fig. 3.** Main block diagram for chain code-based compression system.

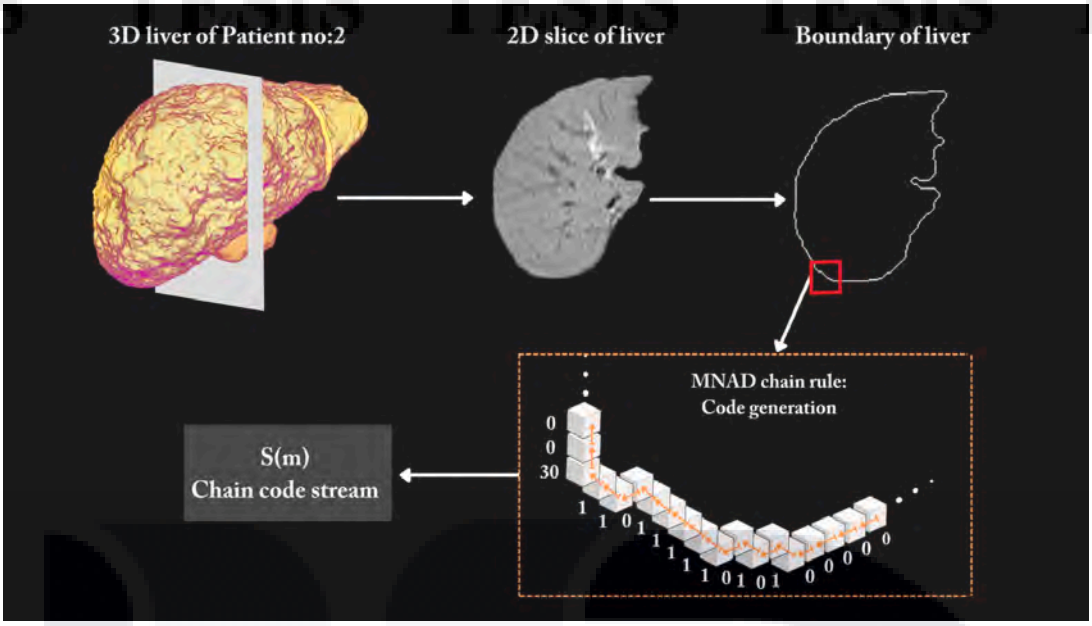


**Fig. 4.** Implementation of the MNAD chain rule.

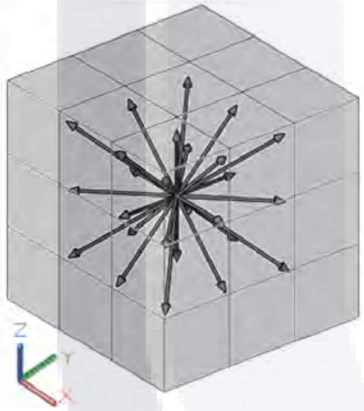


**Fig. 5.** The 26 directions of the F26 chain code.

the closest, i.e., we look for $v_1(2)$ of the second slice with coordinates (2,4,2). In this case, both voxels, $v_n(1)$ and $v_1(2)$ are neighbors. So, the corresponding vector in the path to go from the bottom to the top slice is (2, 4, 2)-(3, 5.1) = (-1,-1,1), which, according to Table 2, is labeled with the **n** symbol.

To obtain F26 chain code, F, firstly find the first non-visited contour voxel $v_1$. Store its coordinates in the chain F = (2,5,1). From $v_1(1)$, start to encode the contour shape of the first slice using F26 in the counter-clockwise direction regarding the Z-axis. Stop on the voxel $v_n(1)$, which is the voxel just before to $v_1(1)$ (the last voxel visited on the contour shape). Of course, if there are *n* voxels that make up the contour, then there are *n*-1 vectors of F26 that encode it. So, we have: F = (2,5,1) **dcbbaaaahagffedefn**. Note that the final symbol of the string is the symbol **n**; this is because there is a slice change, and an F26 symbol is required. From $v_1(2)$, start to encode the contour shape of the top slice using F26 in the clockwise direction. Stop on the voxel $v_n(2)$, which is the voxel just before $v_1(2)$, so, we have: F = (2,5,1)**dcbbaaaahagffedefndbdaaghe.** At this point, the two contours have been coded. However, in this case, there is another contour in the top slice uncoded. Code now this uncoded contour. Choose a voxel from a non-visited contour and store its coordinates. In this case, they are (7,3,2). Code in contour clockwise, in this case is **dcahhe**, and concatenates the chain, which results in F = (2,5,1)**dcbbaaaahagffedefndbdaaghe**(7,3,2) **dcahhe.**

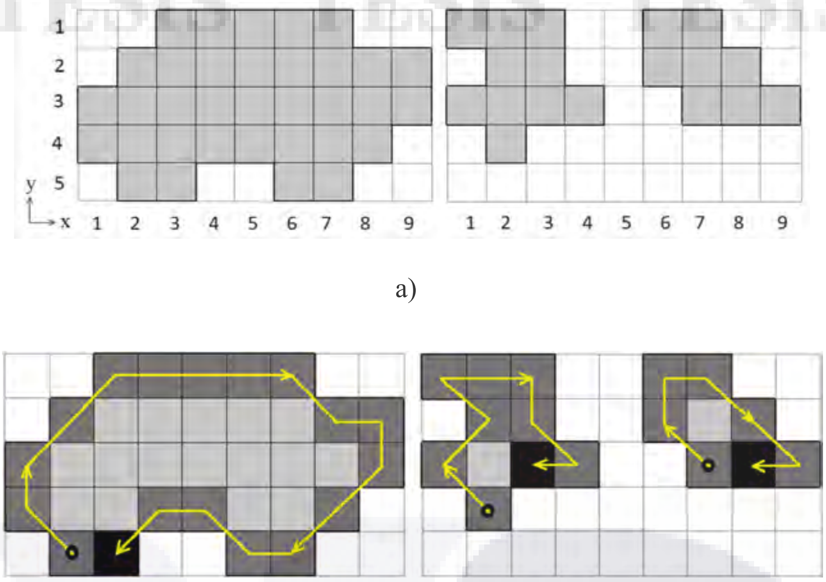
**Table 3**
Performance of different compression techniques.

| Compression Techniques | CT-AAA | | CT-LIVER | | MR-LIVER | |
|---|---|---|---|---|---|---|
| | CR | rCR | CR | rCR | CR | rCR |
| CS.NAD | **9976** | **99.43** | 302.2 | 81.5 | 185 | 81.2 |
| CS.MNAD | **9958** | **99.43** | **309.8** | **81.95** | **186.5** | **81.3** |
| CV.F26 | 9411 | 99.40 | **443.9** | **87.3** | **188.3** | **81.5** |
| JBIG2 | 148.5 | 62.3 | 98.7 | 43.3 | 50.7 | 31.3 |
| ABIC | 432.8 | 87.1 | 214.2 | 73.9 | 102.2 | 65.9 |

**Fig. 6.** A 3D object in a grid of two slices.

**Table 2**
Symbol assignment of elements of the grid bases.

| *b* | (1,0,0) | (1,1,0) | (0,1,0) | (-1,1,0) | (-1,0,0) | (-1,-1,0) | (0,-1,0) | (1,-1,0) | (1,0,1) |
|---|---|---|---|---|---|---|---|---|---|
| Symbol | **a** | **b** | **c** | **d** | **e** | **f** | **g** | **h** | **i** |
| *b* | (1,1,1) | (0,1,1) | (-1,1,1) | (-1,0,1) | (-1,-1,1) | (0,-1,1) | (1,-1,1) | (1,0,-1) | (1,1,-1) |
| Symbol | **j** | **K** | **l** | **m** | **n** | **o** | **p** | **q** | **r** |
| *b* | (0,1,-1) | (-1,1,-1) | (-1,0,-1) | (-1,-1,-1) | (0,-1,-1) | (1,-1,-1) | (0,0,1) | (0,0,-1) | |
| Symbol | **s** | **T** | **u** | **v** | **w** | **x** | **y** | **z** | |



a)

b)

**Fig. 7.** Performance of CS, CV, and state-of-the-art bi-level techniques for liver-MR T1 and T2 sequences dataset.

Fig. 8(a) shows a 3D view of Fig. 6, whereas Fig. 8(b) shows the path covered to obtain the chain code. Finally, Fig. 8(c) shows the two coded and connected 3D regions defined when following our method.

On the other hand, Fig. 1 (a) shows the case of AAA, which defines 20 regions.

The helical coding allows maintaining the starting coordinates, saving extra data, for example, the number of coordinates for its decoding. While the region in the currently visited slice is a neighbor of the previous region, as we mentioned above, there is a way to move from its last voxel to the first unvisited voxel, given by a specific F26 symbol, which allows it to recover the shape of the object without needing to know the coordinates that were used when starting to code. In the example given in Fig. 1(b), only 20 coordinates are required to decode our sample. There are different cases to move from the last visited voxel of one slice to the next, which are solved in the (Tapia-Dueñas et al. 2018). Once the chains of the 3D image slices are obtained, we apply an entropy coder based on a context-mixing algorithm (Mahoney 2005), followed by the arithmetic entropy coder, see Fig. 9.

Context-mixing is an algorithm in which the predictions of the next symbol in the chain are calculated by combining two or more statistical models to produce a prediction that becomes better than if it is made with individual predictions. In particular, PAQ file archivers are a family of lossless data compressors based on context-mixing and are distributed as free Software under the GNU general public license (https://mattmahoney.net/dc/). Among the PAQ archives, one of them is called PAQ8l, which is used for the compression, without loss of information, for our kind of data given by chain code symbols, which come from alphabets of size 26.

## 3. Results and discussion

Proposed systems compress the medical datasets: a two-dimensional chain-code-based compression strategy (CS) and volumetric chain code approach CV.F26. The performance of the proposed strategies is evaluated by the universal metric, compression ratio (CR), which is defined as the quotient of the size of the raw image and the size of the compressed data, i.e., bitstream. To reflect the achieved CR percentages according to a reference (PNG, a well-defined universal compression standard), a second metric called Relative Compression Ratio ($rCR$),

$$rCR = (1 - CR_{PNG}/CR_{method}) \times 100$$

is computed, where the $CR_{method}$ is the compression ratio of the executing technique. The CS system utilized F4, F8, 3OT, NAD, and MNAD chain rules as the data-to-symbol encoder. The run-length encoding aims to reveal the redundancy which may consist of chain code symbols. The performance of the CS, CV.F26, and state-of-the-art standards on the CT-AAA dataset are presented in Table 3.

For all dataset groups, the best two methods are indicated in bold fonts. Fig. 10 shows the performance of the CS system regarding different chain rules. Since the CS and the CV.F26 codify only the boundary of the objects (organs) in the segmented images, so it unearths considerable redundant data in the bi-level images. Fig. 10(a) shows that NAD and MNAD achieve the highest compression ratios.

As seen in Table 3, chain code-based compression systems such as MNAD and NAD achieve higher compression performances than standard algorithms such as ABIC and JBIG2. The MNAD achieves approximately the same performances as NAD on data that do not contain the relevant types of redundancy, as it tries to introduce extra redundancy by using shorter-length symbols for movements of the chain code. Therefore, the rCR rates are approximately similar for the dataset of liver and AAA.

The designed MNAD and the most commonly used F8 and NAD were tested on wide shapes having diverse entropy and morphological structure, see Table 4. The MNAD reaches the shortest bit stream, i.e., the highest compression ratio, for all shapes except the tiger and horse, and also gave much better results in other methods such as the circle and the square. Here again, the entropy and morphological structure of the data directly affect the compression performance.

The compression performances of the (Fig. 10-a) and of the CV compared to the CS and state-of-the-art standards (Fig. 10-a) are depicted in Fig. 10. The compression performances of the CS and the CV are depicted in Fig. 10-a and Fig. 10-b, respectively. The CS utilizes different chain rules, which are parameters of the main pipeline, shown in Fig. 3. The MNAD and NAD are two of the best chain rules, having the highest capability of uncovering redundancy. Moreover, the RLE block is utilized to bring out the repetitive patterns in S(m), but the results show that it cannot increase performance except only for the F8 rule. In other words, any redundancy does not exist stemming from the repetition of the chain symbols. This result also constitutes no need to use RLE to increase the performance of the transform block of the systems (Fig. 11).

The CS outperforms CV.F26, which compresses the AAA image sets through a volumetric chain code strategy (see Fig. 10 (b)). The reason

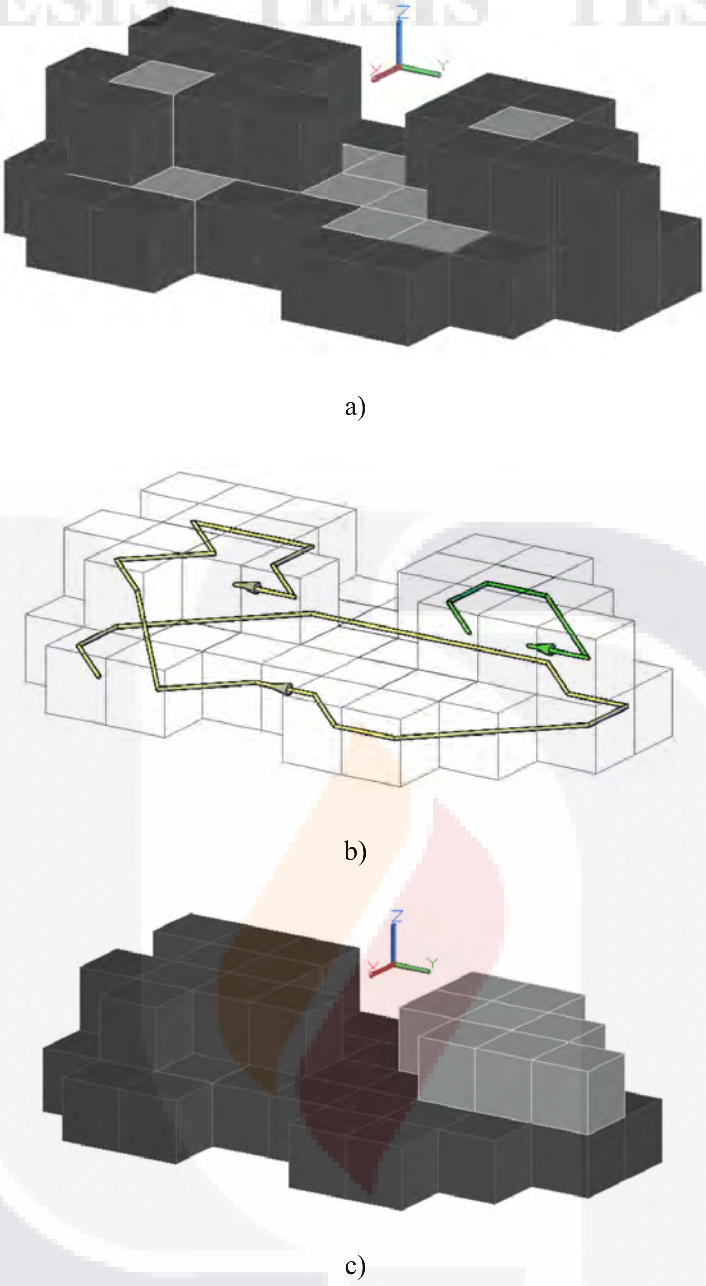


**Fig. 8.** A 3D view of the two slices in Fig. 2X: (a) the contour is in dark color, (b) two paths are covered, and (c) two 3D regions are defined.

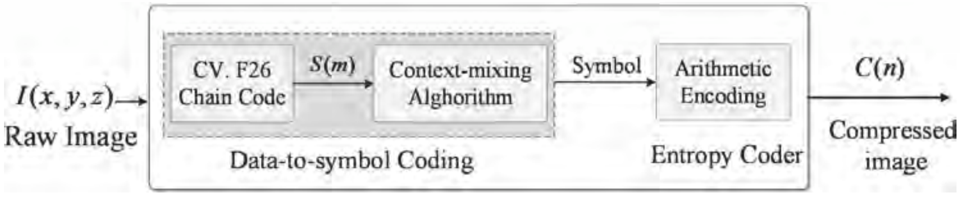
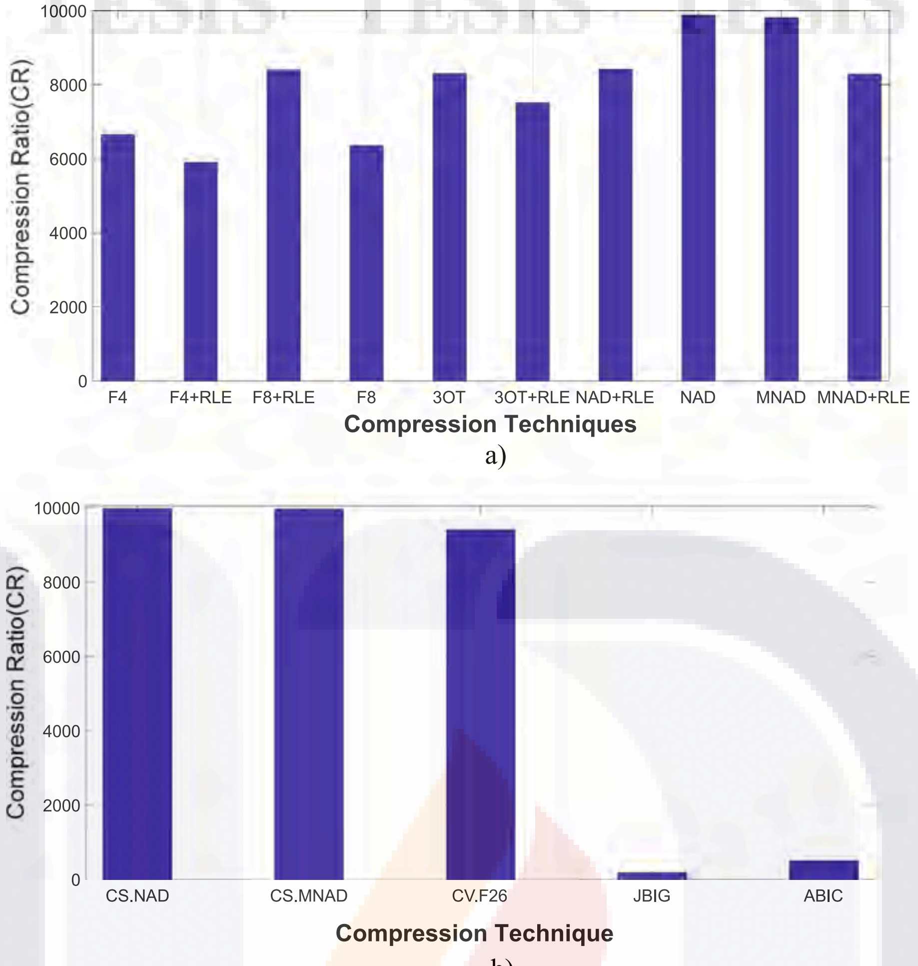


**Fig. 9.** Block diagram for chain code based context-mixing compression.

behind this result is that there is a different level of compactness between the volume and independent slices. More clearly, the independent regions in the slices are labeled as the individual pixel region that the CS compresses. Note that the slices of the abdominal aortic vessel tree are more compact than the volumetric form of the vessel in terms of morphology. Consider the slice of the segmented and volumetric form of organs given in Fig. 2 and Fig. 1(b), respectively. First, the method disassembles three independent regions and labels (append an alphanumeric character to the code). Then, the chain rule and succeeding operations are applied to these regions. These operations are reversibly employed in the sector of the decoder. Finally, the labels are employed for the reconstruction of the entire slice. It should be noted that the independent regions are more compact than the volume of the aortic tree, and also, in terms of energy, defined by the Frobenius norm, remain in a limited range. As a result, the CS system generates fewer symbols and achieves higher CR compared to the CV.F26 system.

The compression efficiency of the methods for the CT-liver dataset is evaluated. When we compare the chain-based systems, contrary to the



a)

b)

**Fig. 10.** The performance for (a) chain code based proposed system (CS) using different chain rules, and (b) chain code based volumetric approach (CV) with the-state-of-the-art and CS system.

**Table 4**
Representing different shapes using chain rules.

| Chain Rules | Bird | Butterfly | Circle | Fire | Horse | Plane | Shuttle | Spider | Square | Star | Thunder | Tiger |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F8 | 1733 | 1379 | 756 | 2137 | **3953** | 1492 | 969 | 4109 | 1092 | 1024 | 1440 | **3835** |
| NAD | 1781 | 1403 | 760 | 2147 | 4605 | 1512 | 989 | 4137 | 1098 | 1038 | 1442 | 4367 |
| MNAD | **1662** | **1272** | **199** | **1955** | **3985** | **1152** | **830** | **3880** | **99** | **283** | **1025** | 3875 |

AAA dataset, the CV.F26 gives a better compression ratio than the CS system. Similar to the previous case's interpretation, the liver shape's compactness is the primary reason for this result. In this instance, the compactness of the individual slices is less than the volume of the liver. In other words, the liver is more compact in three-dimensional space. These lead to more compression performance compared to the CS that compresses the slice of the liver individually. The compression ratios of CV.F26 reach over the ratio of 400:1 while CS remains under the 300:1 ratio. However, not surprisingly, both JBIG2 (CR = 98.7) and ABIC (CR = 214.2) cannot reach the performance of the chain-code-based systems ($MNAD_{CR} = 309.8$ and$NAD_{CR} = 302.2$). The relative compression ratios of the CV.F26 and JBIG2 are 87.3 % and 43 %, which are the best and the worst ratios, respectively. Another noteworthy observation from Table I is that energy and entropy of both slice and volumes do not considerably affect the performance. The entropy of the volume does not increase since the liver is also compact in both 2D- and 3D-spaces.

Besides improving the dictionary, the MNAD also looks for the longest repeated pattern and codifies it as an element of the dictionary. This makes sense in the case of existing smooth regions in the organs. In light of this information, the finding in Table 3 case of $MNAD_{CR} > NAD_{CR}$, which are the result of the livers, reinforces the assertion that the CS.MAND can represent repeated patterns with fewer symbols. The liver has a smoother region than the aortic vessel tree. And thus, the MNAD strategy outperforms the NAD in the case of liver compression. Three symbols of those rules do not contain the redundancy of the chain symbol. However, the F8 rules, which have relatively more elements than the rules mentioned above, include extra redundancy in chain symbols. For the result of the MR dataset, given in Fig. 7, the average compression ratios for MR-T1 and MR-T2 are $NAD_{CR} = 185$ and $(NAD + RLE)_{CR} = 160.5$ $F4_{CR} = 132$ and $(F4 + RLE)_{CR} = 115.8$, $MNAD_{CR} = 186.5$ and $(NAD + RLE)_{CR} = 159.3$. The results also validate the hypothesis since the RLE increases compression performance only for the F8 rule. The compression system achieved similar compression performance for CT and AAA when the RLE.

The form of medical image data is a structured grid and thus the segmented objects can be represented effectively by their outer bounds.. In this context, chain code-based compression approaches were proposed in the study for encoding the boundaries of objects (organs or tissues). The CS and CV.F26 compress medical images in a lossless form. To do this, border pixels of all independent regions (i.e., pixel stacks) in

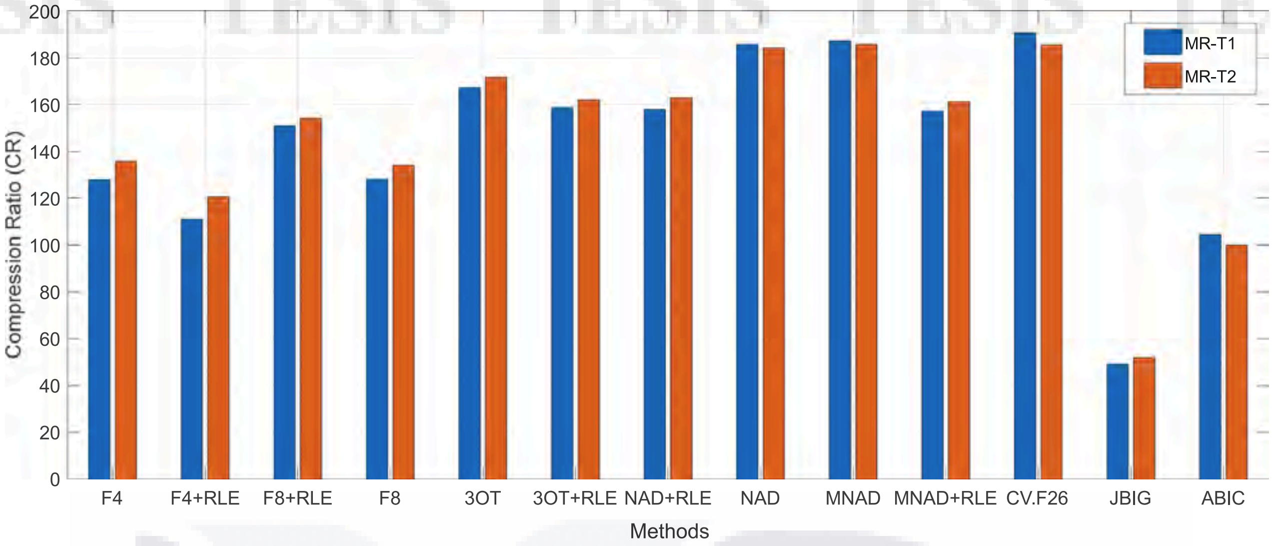


**Fig. 11.** Two views from the positive z-axis of the bottom slice (left) and the top slice (right) of the 3D object in Fig. 6.

the segmented data are detected and labeled. The boundaries are then converted into symbols, and thus, while no loss of information occurs in the segmented data, a significant compression ratio is achieved because only the border regions are encoded.

Consequently, the chain code-based bi-level lossless proposed systems outperform the state-of-the-art standards by far. Furthermore, the morphological structure of the organ is a significant parameter for chain-code-based systems. Entropy and compactness must be considered to achieve optimal compression efficiency. The proposed chain-code-based compression system achieves over 99% relative compression ratio. The results conclude that the proposed chain-rules-based reversible compression system can remarkably uncover redundancy included in medical image data.

## 4. Conclusion

Compression has increasingly become vital for efficiently transmitting healthcare data through teleradiology networks. Unfortunately, the most current compression standards cannot entirely uncover the redundancy in volumetric medical data since they do not consider spatial coherence and morphology of anatomical objects. Moreover, most of these methods are designed to process gray-level data, which means they are not geared for revealing redundancy originating from the binary data structures. However, compression of binary segmented data would be much more effective, especially for visualization studies. In this study, two compression strategies have been proposed to ensure that the segmentation masks can be effectively stored and transmitted over radiology networks in a DICOM-compliant manner.

These proposed systems are CV.F26, which encodes boundary voxels of a segmented organ(s)/tissue(s), and CV, which parametrically encodes boundary information of images in two-dimensional space using various chain rules. The proposed systems have been tested on a large data set consisting of various objects having different characteristics. The results show that both approaches reveal a significant amount of redundancy that existing well-established compression algorithms cannot achieve. Therefore, the proposed methods can make significant contributions to storage and transmission systems. Another important point figured out by the results is that the algorithms' performance primarily depends on the morphological structures (e.g., compactness) of the organs contained by medical images. If the organs with higher compactness in two-dimensional space, the CS system provides a better compression ratio, whereas, for the organs with high compactness value in volumetric space, the CV system is found to be more successful. The proposed chain code-based algorithms undeniably outperform state-of-the-art compression standards, although the compression performance of all methods is directly related to the entropy of the data. Both CV.F26 and CV pipelines are compression strategies developed within the scope of the study. These pipelines are designed by utilizing different chain rules (such as MNAD and NAD) in a parametric form. The MNAD, an improved NAD version, was developed from a new dictionary to shorten the total code length. Thus, the systems achieve considerable compression ratios over %90, which is an outstanding achievement compared to the state-of-the-art standards.

Consequently, both pipelines and the MNAD chain rule are unique contributions introduced by the manuscript. The results conclude that specifically designed compression strategies by considering morphologies of the organs can compress the medical data more efficiently. It should also be emphasized that the proposed algorithms can be designed to include medical metadata information so that it can be integrated into teleradiology networks under DICOM standards.

## CRediT authorship contribution statement

**Erdoğan Aldemir:** Conceptualization, Methodology, Validation, Investigation, Writing – original draft, Writing – review & editing, Funding acquisition, Software, Formal analysis, Visualization, Data curation, Supervision, Project administration, Resources. **Osvaldo Arturo Tapia Dueñas:** Conceptualization, Methodology, Data curation, Writing – original draft. **Ali Emre Kavur:** Data curation, Investigation. **Gulay Tohumoglu:** Conceptualization, Methodology, Supervision. **Hermilo Sánchez-Cruz:** Software, Validation, Supervision, Writing – review & editing. **Mustafa Alper Selver:** Conceptualization, Methodology, Writing – review & editing, Supervision.

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Data availability

Data will be made available on request.

## Acknowledgment

This work was supported by The Scientific and Technological Research Council of Turkey (TÜBİTAK) -Scientific and Technological

Research Projects Funding Program- under grant number 116E133. We also would like to thank Barış Turunç for his valuable contributions.

## References

Aldemir, E., Gezer, N. S., Tohumoglu, G., Mustafa Barış, A., Kavur, E., Dicle, O., & Alper Selver, M. (2020). Reversible 3D compression of segmented medical volumes: Usability Analysis for teleradiology and storage. *Medical Physics, 47*(4), 1727–1737. https://doi.org/10.1002/mp.14053

Aldemir, E., Tohumoglu, G., & Alper Selver, M. (2019). Binary medical image compression using the volumetric run-length approach. *Imaging Science Journal, 67* (3), 123–135. https://doi.org/10.1080/13682199.2019.1565695

Andriole, K. P., Wolfe, J. M., Ramin Khorasani, S., Treves, T., Getty, D. J., Jacobson, F. L., … Seltzer, S. E. (2011). Optimizing analysis, visualization, and navigation of large image data sets: One 5000-section Ct scan can ruin your whole day. *Radiology, 259* (2), 346–362. https://doi.org/10.1148/radiol.11091276

Bairagi, V. K., & Sapkal, A. M. (2013). ROI-based DICOM image compression for telemedicine. *Sadhana - Academy Proceedings in Engineering Sciences, 38*(1), 123–131. https://doi.org/10.1007/s12046-013-0126-4

Bradley, W. G. (2008). Off-site teleradiology: The pros. *Radiology, 248*(2), 337–341. https://doi.org/10.1148/radiol.2482080569

Bruylants, T., Munteanu, A., & Schelkens, P. (2015). Wavelet based volumetric medical image compression. *Signal Processing: Image Communication, 31*, 112–133. https://doi.org/10.1016/j.image.2014.12.007

Chen, Z., Shuhang, G.u., Guo, L.u., & Dong, X.u. (2022). Exploiting intra-slice and inter-slice redundancy for learning-based lossless volumetric image compression. *IEEE Transactions on Image Processing, 31*, 1697–1707. https://doi.org/10.1109/TIP.2022.3140608

Chung, K., & Park, R. C. (2019). Cloud based U-healthcare network with QoS guarantee for mobile health service. *Cluster Computing, 22*, 2001–2015. https://doi.org/10.1007/s10586-017-1120-0

Devadoss, C. P., & Sankaragomathi, B. (2019). Near lossless medical image compression using block BWT–MTF and hybrid fractal compression techniques. *Cluster Computing, 22*, 12929–12937. https://doi.org/10.1007/s10586-018-1801-3

Dicom "Digital Imaging and Communications in Medicine (DICOM) Supplement 111: Segmentation Storage SOP Class Prepared By :" 1–40 medical/dicom/final/sup132_ft.pdf.) 2006 Retrieved October 1, 2020 (ftp://medical.nema.org/.

DICOM Standards Committee, Working Group 17 (3D). 2006. "Supplement 111: Segmentation Storage SOP Class." *Digital Imaging and Communications in Medicine (DICOM)* 17:1–34. Retrieved October 1, 2020 (ftp://medical.nema.org/medical/dicom/final/sup111_ft. pdf.).

Dionisio, John David N., Ricky K. Taira, Usha Sinha, David B. Johnson, Benjamin Y. Dai, Gregory H. Tashima, Stephen Blythe, Richard Johnson, and Hooshang Kangarloo. 2000. "Teleraetiology as a Foundation for an Enterprise-Wide Health Care Delivery System." *Radiographics* 20(4):1137–50. doi: 10.1148/radiographics.20.4.g00jl321137.

Fischer, F., Alper Selver, M., Gezer, S., Dicle, O., & Hillen, W. (2015). Systematic parameterization, storage, and representation of volumetric DICOM Data. *Journal of Medical and Biological Engineering, 35*(6), 709–723. https://doi.org/10.1007/s40846-015-0097-5

Fischer, F., Alper Selver, M., Hillen, W., & Güzeliş, C. (2010). Integrating segmentation methods from different tools into a visualization program using an object-based plug-in interface. *IEEE Transactions on Information Technology in Biomedicine, 14*(4), 923–934. https://doi.org/10.1109/TITB.2010.2044243

Gunderman, R. B., & Chou, H. Y. (2016). The future of radiology consultation. *Radiology, 281*(1), 6–9. https://doi.org/10.1148/radiol.2016152781

Hassanalieragh, Moeen, Alex Page, Tolga Soyata, Gaurav Sharma, Mehmet Aktas, Gonzalo Mateos, Burak Kantarci, and Silvana Andreescu. 2015. "Health Monitoring and Management Using Internet-of-Things (IoT) Sensing with Cloud-Based Processing: Opportunities and Challenges." Pp. 285–92 in *Proceedings - 2015 IEEE International Conference on Services Computing, SCC 2015*.

Henriques, T., Gonçalves, H., Antunes, L., Matias, M., Bernardes, J., & Costa-Santos, C. (2013). Entropy and compression: Two measures of complexity. *Journal of Evaluation in Clinical Practice, 19*(6), 1101–1106. https://doi.org/10.1111/jep.12068

Hsu, W. Y. (2017). Clustering-based compression connected to cloud databases in telemedicine and long-term care applications. *Telematics and Informatics, 34*(1), 299–310. https://doi.org/10.1016/j.tele.2016.05.010

Karimi, N., Shadrokh Samavi, S. M., Soroushmehr, Rez., Shirani, Shahram, & Najarian, Kayvan (2016). Toward practical guideline for design of image compression algorithms for biomedical applications. *Expert Systems with Applications, 56*, 360–437. https://doi.org/10.1016/j.eswa.2016.02.047

Kaur, M., & Wasson, V. (2015). ROI based medical image compression for telemedicine application. *Procedia Computer Science, 70*, 579–585. https://doi.org/10.1016/j.procs.2015.10.037

Kavur, A., Emre, N. S., Gezer, M. B., Aslan, S., Conze, P. H., Groza, V., … Alper Selver, M. (2021). CHAOS challenge - Combined (CT-MR) healthy abdominal organ segmentation. *Medical Image Analysis, 69*. https://doi.org/10.1016/j.media.2020.101950

Kavur, A. E., Gezer, N. S., Barış, M., Şahin, Y., Özkan, S., Baydar, B., … Alper Selver, M. (2020). Comparison of semi-automatic and deep learning-based automatic methods for liver segmentation in living liver transplant donors. *Diagnostic and Interventional Radiology, 26*(1), 11–21. https://doi.org/10.5152/dir.2019.19025

Kim, K. J., Lee, K. H., Kim, B., Richter, T., Yun, I. D., Lee, S. U., … Shim, H. (2011). JPEG2000 2D and 3D reversible compressions of thin-section chest CT images: Improving compressibility by increasing data redundancy outside the body region1. *Radiology, 259*(1), 271–327. https://doi.org/10.1148/radiol.10100722

Krishnan, K., Marcellin, M. W., Bilgin, A., & Nadar, M. S. (2006). Efficient transmission of compressed data for remote volume visualization. *IEEE Transactions on Medical Imaging, 25*(9), 1189–1199. https://doi.org/10.1109/TMI.2006.879956

Liu, F., Hernandez-Cabronero, M., Sanchez, V., Marcellin, M. W., & Bilgin, A. (2017). The Current role of image compression standards in medical imaging. *Information (Switzerland), 8*(4). https://doi.org/10.3390/info8040131

Mahoney, M. V. (2005). *Adaptive Weighing of Context Models for Lossless Data Compression.* USA CS-2005-16:1–6: Florida Institute of Technology Melbourne.

Mao, Q., Zhou, B., Zou, Q., & Li, Q. (2015). Efficient and lossless compression of raster maps. *Signal, Image and Video Processing, 9*(1), 133–145. https://doi.org/10.1007/s11760-013-0428-3

Marks, K. M. (1998). A JBIG-ABIC compression engine for digital document processing. *IBM Journal of Research and Development, 42*(6), 753–778. https://doi.org/10.1147/rd.426.0753

Martínez, L. A., Bribiesca, E., & Guzmán, A. (2017). Chain coding representation of voxel-based objects with enclosing, edging and intersecting trees. *Pattern Analysis and Applications, 20*(3), 825–844. https://doi.org/10.1007/s10044-016-0540-4

Norbash, A., Bluth, E., Lee, C. I., Francavilla, M., Donner, M., Dutton, S. C., … McGinty, G. (2014). Radiologist manpower considerations and imaging 3.0: Effort planning for value-based imaging. *Journal of the American College of Radiology, 11* (10), 953–998. https://doi.org/10.1016/j.jacr.2014.05.022

Parikh, S. S., Ruiz, D., Kalva, H., Fernandez-Escribano, G., & Adzic, V. (2018). High bit-depth medical image compression with HEVC. *IEEE Journal of Biomedical and Health Informatics, 22*(2), 552–560. https://doi.org/10.1109/JBHI.2017.2660482

Pianykh, Oleg S. 2012. "Digital Imaging and Communications in Medicine (DICOM)." *Digital Imaging and Communications in Medicine (DICOM)* 1–99. Retrieved November 12, 2020 (ftp://medical.nema.org/medical/dicom/supps/PC/sup156_pc.pdf).

Sánchez-Cruz, H., Bribiesca, E., & Rodríguez-Dagnino, R. M. (2007). Efficiency of chain codes to represent binary objects. *Pattern Recognition, 40*(6), 1660–1674. https://doi.org/10.1016/j.patcog.2006.10.013

Sánchez-Cruz, H., López-Valdez, H. H., & Cuevas, F. J. (2014). A new relative chain code in 3D. *Pattern Recognition, 47*(2), 769–788. https://doi.org/10.1016/j.patcog.2013.08.010

Sanchez, V., Nasiopoulos, P., & Abugharbieh, R. (2008). Efficient 4D Motion Compensated Lossless Compression of Dynamic Volumetric Medical Image Data. In *In 2008 IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 549–552).

Selver, M. A., & Emre Kavur, A. (2016). Implementation and use of 3D pairwise geodesic distance fields for seeding abdominal aortic vessels. *International Journal of Computer Assisted Radiology and Surgery, 11*(5), 803–816. https://doi.org/10.1007/s11548-015-1321-z

Sharma, U., Sood, M., & Puthooran, E. (2020). Volumetric medical image compression using inter-slice correlation switched prediction approach. *International Journal of Imaging Systems and Technology, 30*(4), 1194–1208. https://doi.org/10.1002/ima.22425

Tapia-Dueñas, O. A., Sánchez-Cruz, H., López, H. H., & Sossa, H. (2018). Coding 3D Connected Regions with F26 Chain Code. In *In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Vol. 11289 LNAI* (pp. 3–14).

Text, Final. 1999. *Digital Imaging and Communications in Medicine (DICOM) Supplement 33 : Grayscale Softcopy Presentation State Storage.*

Thrall, J. H. (2007). Teleradiology. Part II. Limitations, risks, and opportunities. *Radiology, 244*(2), 325–338. https://doi.org/10.1148/radiol.2442070676

Žalik, B., & Lukač, N. (2014). Chain code lossless compression using move-to-front transform and adaptive run-length encoding. *Signal Processing: Image Communication, 29*(1), 96–106. https://doi.org/10.1016/j.image.2013.09.002

Žalik, B., Mongus, D., Žalik, K. R., & Lukač, N. (2016). Chain code compression using string transformation techniques. *Digital Signal Processing: A Review Journal, 53*, 1–10. https://doi.org/10.1016/j.dsp.2016.03.002

Žalik, B., Žalik, K. R., Zupančič, E., Lukač, N., Žalik, M., & Mongus, D. (2019). Chain code compression with modified interpolative coding. *Computers and Electrical Engineering, 77*, 27–36. https://doi.org/10.1016/j.compeleceng.2019.05.001

# 8 Conclusiones

En esta tesis se han utilizado diversos métodos y técnicas para el procesamiento de imágenes y objetos. Se ha demostrado que la aplicación de nuestra gramática libre de contexto mejora significativamente la precisión y eficiencia en la simplificación de objetos tanto en 2D como en 3D. Por otro lado, la combinación de diferentes códigos de cadena con otras técnicas ha resultado en una mejora en la compresión de datos médicos binarios segmentados. Estos hallazgos destacan la importancia de integrar múltiples herramientas y enfoques para abordar problemas complejos en el procesamiento de imágenes y objetos.

Contribuciones a las ciencias de la computación:

- El desarrollo y aplicación de una gramática libre de contexto para la simplificación de objetos en 2D y 3D, que mejora significativamente la precisión y eficiencia en el procesamiento de imágenes y objetos.
- La combinación de diferentes códigos de cadena con otras técnicas para mejorar la compresión de datos médicos binarios segmentados.

Contribuciones a la visión por computadora:

- La aplicación de la gramática libre de contexto para la simplificación de objetos en 2D y 3D, que permitirá una mejor comprensión y análisis de la información visual.
- La mejora en la compresión de datos médicos binarios segmentados, lo que permitirá un mejor almacenamiento y transmisión de información visual médica.

En resumen, estas contribuciones tienen un impacto en ambos campos, pero están enfocadas en el procesamiento y análisis de imágenes y objetos.

## 8.1 Trabajos futuros

Uno de mis principales objetivos para trabajos futuros es la reconstrucción de la malla triangular a partir de la nube de puntos simplificada. Esta técnica me permitirá representar objetos en 3D con menor cantidad de información, pero sin perder su esencia. De esta manera, se facilitará la visualización de los objetos.

Otro aspecto fundamental será simplificar los datos en mapas. Este proceso involucra detectar los puntos de inicio de las líneas rectas discretas, lo que se traduce en un ahorro considerable de espacio de almacenamiento y tiempo de procesamiento.

Por otro lado, me interesa investigar la equivalencia de diferentes códigos de cadena, especialmente en la transición de píxeles cuadrados a triangulares y/o hexagonales. Esto permitiría desarrollar algoritmos más eficientes y precisos para el procesamiento de imágenes.

Por último, otro de mis objetivos a futuro es la combinación de códigos de cadena y aprendizaje automático para mejorar la clasificación de imágenes. Esto implicaría el desarrollo de modelos de aprendizaje automático que utilicen información de los códigos de cadena para identificar patrones en las imágenes. Con esto se podría obtener resultados más precisos y eficientes en la clasificación y análisis de imágenes.

# Referencias

[1] Osvaldo A Tapia-Dueñas and Hermilo Sánchez-Cruz. "Context-free grammars to detect straight segments and a novel polygonal approximation method". In: *Signal Processing: Image Communication* 91 (2021), p. 116080.

[2] Hermilo Sánchez-Cruz, Osvaldo A Tapia-Dueñas, and Francisco Cuevas. "Polygonal approximation using a multiresolution method and a context-free grammar". In: *Pattern Recognition: 11th Mexican Conference, MCPR 2019, Querétaro, Mexico, June 26–29, 2019, Proceedings 11*. Springer. 2019, pp. 261–270.

[3] Osvaldo A Tapia-Dueñas, Hermilo Sánchez-Cruz, and Hiram H López. "3D object simplification using chain code-based point clouds". In: *Multimedia Tools and Applications* (2022), pp. 1–25.

[4] Erdoğan Aldemir et al. "Chain code strategy for lossless storage and transfer of segmented binary medical data". In: *Expert Systems with Applications* 216 (2023), p. 119449.

[5] Osvaldo A Tapia-Dueñas et al. "Coding 3D connected regions with F26 chain code". In: *Advances in Computational Intelligence: 17th Mexican International Conference on Artificial Intelligence, MICAI 2018, Guadalajara, Mexico, October 22–27, 2018, Proceedings, Part II 17*. Springer. 2018, pp. 3–14.