2D Grey-Level Convex Hull Computation: A Discrete 3D Approach

Ingela Nyström[1], Gunilla Borgefors[2], and Gabriella Sanniti di Baja[3]

[1] Centre for Image Analysis, Uppsala University
Uppsala, Sweden
ingela@cb.uu.se

[2] Centre for Image Analysis, Swedish University for Agricultural Sciences
Uppsala, Sweden
gunilla@cb.uu.se

[3] Istituto di Cibernetica, National Research Council of Italy (CNR)
Pozzuoli (Napoli), Italy
gsdb@imagm.cib.na.cnr.it

Abstract. We compute discrete convex hulls in 2D grey-level images, where we interpret grey-level values as heights in 3D landscapes. For these 3D objects, using a 3D binary method, we compute approximations of their convex hulls. Differently from other grey-level convex hull algorithms, producing results convex only in the geometric sense, our convex hull is convex also in the grey-level sense.

Keywords: shape representation, grey-level morphology, concavity analysis, convex deficiency

1 Introduction

The notion of convexity is important in image analysis tasks, especially in shape analysis. One tool that has been considered valuable for a long time is the *convex hull*, defined as the smallest convex set including a given set [6]. The convex hull can be used in a number of different applications. Two recent examples are: in robotics for collision detection [7]; and in laminated manufacturing to enable extraction of the object [5]. More general examples are found in [4, Ch. 11].

In most of the image analysis literature on convex hull computation, the original sets considered are sets of isolated points or objects in binary images. Most digital images are originally grey-level (or colour) images, and when creating a binary image much information is lost. In many cases, that information loss is serious and the end results not acceptable. Therefore, working on the original grey-level image is preferable. A way to take grey-level information into account while still using the methods developed for binary images, is to transform the 2D grey-level image into a 3D binary image, where for each image element the

J. Bigun and T. Gustavsson (Eds.): SCIA 2003, LNCS 2749, pp. 763–770, 2003.

grey-level value becomes the third co-ordinate. Binary image processing is usually simpler than grey-level image processing. This transformation is also used in, e.g., grey-level morphology, where common operators are the top hat and the rolling ball transformations [10],[4, Ch. 9.6].

Convex hulls found in literature are usually defined using continuous notions and the result often a continuous set, rather than a discrete set, even when working with digital imagery. We think it is desirable to keep the result in the image domain.

In this paper, we present an algorithm for computing the 2D discrete grey-level convex hull, by using a method that computes a sufficiently good approximation of convex hulls in 3D binary images [3]. This is a purely discrete approach using small local operations, as opposed to, e.g., the Quickhull algorithm [1]. The resulting convex hull and convex deficiency (i.e., the set difference between the convex hull and the object) are both 2D grey-level images. These images contain geometric and grey-level information, as both geometric (planar) concavities and grey-level concavities (areas with lower grey-level than their surroundings) are filled. In addition to shape analysis without thresholding, differentiating objects with highly varying grey-levels from objects with smooth grey-level variation is possible, e.g., by analysis of their grey-level convex deficiency. We apply the algorithm when analyzing the structure of a paper sheet through confocal microscope images within pulp and paper research.

The idea of grey-level convex hulls is not new. A previous algorithm is found in [8, 9]. This algorithm produces convex hulls where every plane parallel to the original image plane contains only convex objects, but where a cut perpendicular to this plane can contain concavities. It is thus a true convex hull only in the geometric sense, not in the grey-level sense.

2 Background

Consider a binary image in $\mathbb{Z}^n$ containing object and background elements.

Definition 1. *A* digital convex set *is a set of object elements in $\mathbb{Z}^n$ bounded by a finite number of discrete half-spaces.*

Definition 2. *The* convex hull *of a set of object elements in $\mathbb{Z}^n$ is the smallest convex set containing the original set.*

Definition 3. *The* convex deficiency*, or* concavity regions*, of a set of object elements $\mathbb{Z}^n$ is the set of elements in the convex hull that are not members of the original set.*

In image analysis, quantitative description of the convex hull and the convex deficiency of an object provides valuable information about geometric properties. In practice, one is usually content by computing an approximation of the convex hull of an object, e.g., a covering polygon/polyhedron. If the approximation is reasonably good, so are the resulting measurements. Also, local operations can be

used, which are fast and possible to parallelise easily. For example, in [4, Ch. 9.6] the morphological algorithm given to compute 2D convex hulls computes the smallest *octagon* containing the object.

We have previously presented a method for computing covering polyhedra for digital volume objects, inspired by earlier work in 2D, see [2],[3]. The resulting covering polyhedron is convex and is a good approximation of the convex hull of the object. This method will be part of the present algorithm, see Section 3.1.

A 2D grey-level image is converted to a 3D binary image as follows. For each pixel at position (x, y) with grey-level g, all voxels at positions (x, y, z) with $1 \leq z \leq g$ are set to object and all voxels at positions (x, y, z) with $g + 1 \leq z \leq 255$ and $z = 0$ are set to background. Because grey-level is not equivalent to height, the correspondence between grey-level g and z value is not given, but depends on the application. Linear scaling may be necessary. See Figures 1 and 2 for original grey-level (a) and corresponding 3D representations (b). The inverse conversion is straight-forward.

3 Computing the grey-level convex hull

3.1 Binary 3D convex hull computation

This Section is a brief version of [3]. We start with a binary 3D image with one or more objects. Our covering polyhedra are built by "filling" *local concavities*, i.e., by changing appropriate background voxels to object voxels. Local concavities are defined by the number and the configuration of neighbouring object voxels. Consider a border voxel, i.e., a background voxel having at least one face-neighbour in the object. If it is located in a local concavity it is changed to object. By iteratively filling local concavities the concavity regions are filled globally:

1. **Label the border voxels.** For each border voxel the number of its object face- and edge-neighbours are counted, separately in the x-, y-, and z-planes. The three resulting sums, denoted Σ_x, Σ_y, and Σ_z, are stored as a vector label for the border voxel.
2. **Include the border voxels satisfying the concavity criteria in the object.** Voxels with at least one $\Sigma_\xi > 4$, $\xi \in \{x, y, z\}$; and voxels with one $\Sigma_\xi = 4$ having, in the same plane ξ, at least one neighbour with $\Sigma_\xi > 2$, are defined as being located in local concavities.

The larger the neighbourhood used to identify local concavities is, the more half-spaces can be used for delimiting the covering polyhedron, and the better the approximation of the convex hull will be. Here, simple $3 \times 3 \times 3$ operators are used, but curvature information is derived from a $5 \times 5 \times 5$ neighbourhood. The resulting covering polyhedron is convex and includes the convex hull. It can have up to 90 faces. (If only information from a $3 \times 3 \times 3$ neighbourhood were used, the polyhedron could have at most 26 faces.) The difference between our covering polyhedron and the convex hull is reasonably small, and the covering polyhedron is good enough for most practical purposes.

3.2 Grey-level concavity analysis

The grey-level concavity computation consists of the following steps. Convert the 2D grey-level image f to a 3D "landscape". Compute the convex hull of this landscape. Project the convex landscape back to a 2D grey-level image g, i.e., the grey-level convex hull. The grey-level convex deficiency is the difference image $h = g - f$. The image h can of course also be represented as a landscape during the process of analysis. All computed images are analysed according to the specific application. More formally:

1. Convert 2D grey-level image f to 3D binary image.
2. Compute 3D convex hulls of objects in binary image according to method in Section 3.1.
3. Convert 3D convex hull image to 2D grey-level image g.
4. Compute 2D grey-level convex deficiency image h.
5. Convert h to 3D binary image.
6. (a) Analyse 3D convex hull.
   (b) Analyse 3D convex deficiency.
7. (a) Analyse 2D grey-level convex hull.
   (b) Analyse 2D grey-level convex deficiency.

4 Examples and comparison

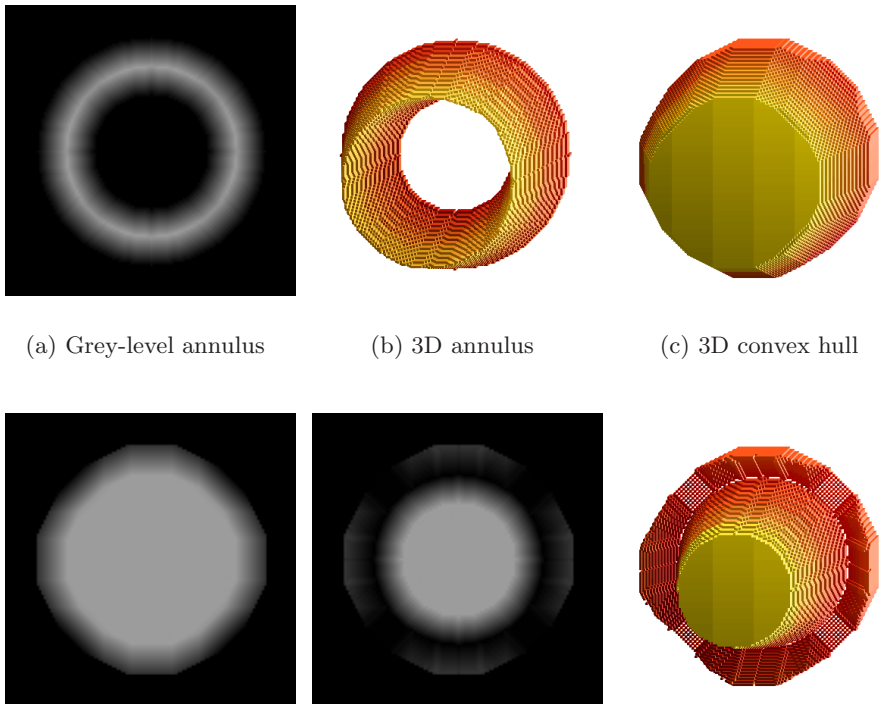
The first set of images is a synthetic example, showing how our method performs in the case of a geometric concavity. We start from a binary annulus with inner radius 25 pixels and outer radius 50 pixels. The grey-level values are the distance values to the background (according to the 3-4 distance transform). The maximum grey-level, and thus the maximum height, is 39. See Figure 1. Note that the 3D representations have flat "bottoms".

The next set of images is a real example, showing how our method performs in the case of grey-level concavities. We start from a photograph of a face. The illumination is uneven and there are also spots of dirt on the face that will become filled in during the grey-level convex hull computation. See Figure 2.

The purpose of the last image example, taken from [8, 9], is to illustrate the difference between our grey-level convex hull and the one computed by Soille, see Figure 3, where both Soille's and our convex hulls are shown. The difference is clearest in the three profiles of the images (cuts perpendicular to the image plane). Soille's convex hull is not convex in this direction, while our is.

5 Discussion

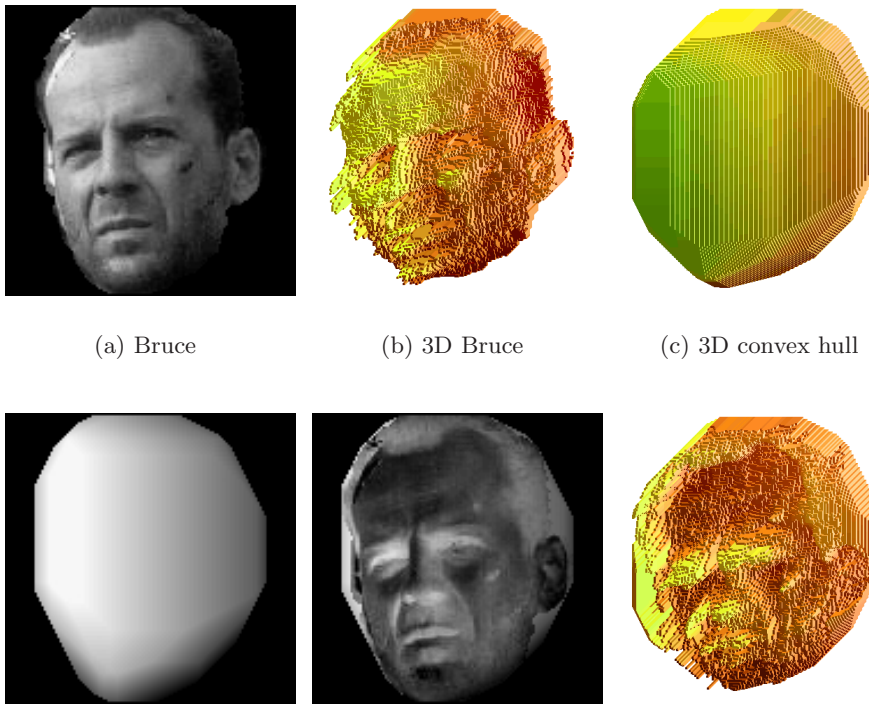
This paper describes an algorithm using a purely discrete approach, based on a method for 3D binary images, to compute 2D grey-level convex hulls. For good results, some pre-processing by, e.g., grey-level closing, is generally needed in most applications, due to the inherent noise in grey-level images. The resulting



(a) Grey-level annulus (b) 3D annulus (c) 3D convex hull

(d) Grey-level convex hull (e) Grey-level concavities (f) 3D concavities

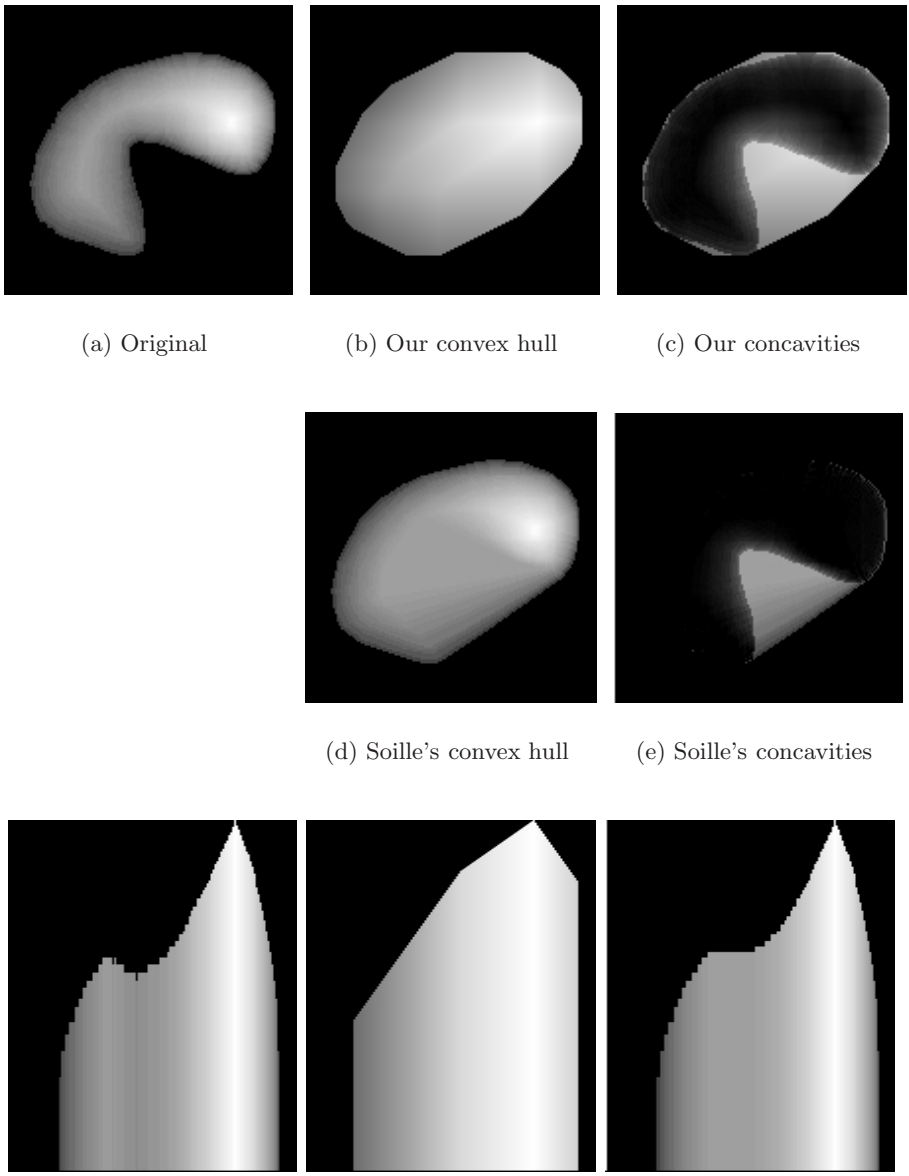
Fig. 1. Grey-level convex hull computation and analysis. (a) An annulus using distances to the background as grey-levels. (b) The 3D representation of the annulus. (c) The convex hull of (b). (d) The grey-level representation of (c). (e) The grey-level concavity regions, i.e., (d)−(a). (f) The 3D representation of (e).



(a) Bruce (b) 3D Bruce (c) 3D convex hull

(d) Grey-level convex hull (e) Grey-level concavities (f) 3D concavities

Fig. 2. Grey-level convex hull computation and analysis. (a) A photograph of a face. (b) The 3D representation of the face. (c) The convex hull of (b). (d) The grey-level representation of (c). (e) The grey-level concavity regions, i.e., (d)−(a). (f) The 3D representation of (e): the uneven illumination has been eliminated and the spots of dirt shows quite well.



(a) Original (b) Our convex hull (c) Our concavities

(d) Soille's convex hull (e) Soille's concavities

(f) 1D profiles from (a), (b), and (d)

Fig. 3. Comparison of two methods for grey-level convex hull computation.

convex hull approximations fulfill convexity in terms of geometric as well as grey-level information, while other grey-level convex hull algorithms have focused on geometric information. What is desired depends on the application.

An application we have in mind, is within pulp and paper research when analyzing the structure of a paper sheet through confocal microscope images. We obtain 2D grey-level images, where the grey-levels correspond to the depth at which the first paper fibre is visible, i.e., how deep the pores penetrate into the paper. It is of interest to study the structure of the surface under different pressures to measure how the pores change. The pressure may be unevenly distributed, a problem overcome by computing our grey-level convex hull.

Our current implementation of the algorithm first converts the 2D grey-level images to 3D binary images. It is possible, for efficiency reasons, to implement it directly for the 2D grey-level images.

The algorithm should extend quite easily to 3D grey-level images. Intuitively, we then see the images as 4D binary images. Our idea is to use information from the $3 \times 3 \times 3 \times 3$ neighbourhood for the 4D convex hull computations, but performed as 3D computations.

Acknowledgements

Dr. Pierre Soille, EC Joint Research Centre, Ispra, Italy, is acknowledged for providing the image in Figure 3.

References

1. C. B. Barber, D. P. Dobkin, and H. Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software*, 22(4):469–483, 1996.
2. G. Borgefors, I. Nyström, and G. Sanniti di Baja. Computing covering polyhedra of non-convex objects. In *Proc. of* 5^{th} *British Machine Vision Conference, York, UK*, pages 275–284, 1994.
3. G. Borgefors and G. Sanniti di Baja. Analyzing nonconvex 2D and 3D patterns. *Computer Vision and Image Understanding*, 63(1):145–157, 1996.
4. R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Prentice Hall, Inc, Upper Saddle River, New Jersey, 2nd ed., 2002.
5. K. P. Karunakaran, *et al.* Efficient stock cutting for laminated manufacturing. *Computer-Aided Design*, 34(4):281–298, 2002.
6. F. P. Preparata and M. I. Shamos. *Computational Geometry an Introduction*, Ch. 3. Springer-Verlag, New York, 1985.
7. H. D. Sherali, J. C. Smith, and S. Z. Selim. Convex hull representations of models for computing collisions between multiple bodies. *European Journal of Operational Research*, 135(3):514–526, 2001.
8. P. Soille. Grey scale convex hulls: Definition, implementation, and application. In H. Heijmans and J. Roerdink, eds, *Proc. of ISMM'98. Computational Imaging and Vision*, Vol. 12, pages 83–90. Kluwer Academic Publishers, 1998.
9. P. Soille. From binary to grey scale convex hulls. *Fundamenta Informaticae*, 41(1–2):131–146, 2000.
10. S. Sternberg. Grayscale morphology. *Computer Graphics and Image Processing*, 35:333–355, 1986.