Published in Image Processing On Line on 2019–12–09.
Submitted on 2019–07–09, accepted on 2019–11–22.
ISSN 2105–1232 © 2019 IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
https://doi.org/10.5201/ipol.2019.270

The Gradient Product Transform: An Image Filter for Symmetry Detection

Christoph Dalitz, Jens Wilberg, Manuel Jeltsch

Institute for Pattern Recognition (iPattern), Niederrhein University of Applied Sciences, Krefeld, Germany
(christoph.dalitz@hs-niederrhein.de)

Abstract

The Gradient Product Transform (GPT) is an image filter that converts a grayscale image into a float image, such that points representing a point reflection symmetry center obtain a high score. Beside the symmetry score, it also yields an estimator for the size of the symmetry region around each point. Apart from describing the GPT, the article also explains its application for two use cases: detection of objects with a point reflection or C_{2m} rotational symmetry, and the extraction of blood vessel skeletons from medical images. For the detection of symmetric objects, a score normalization procedure is suggested that allows to choose a fixed threshold for score values representing actual symmetries.

Source Code

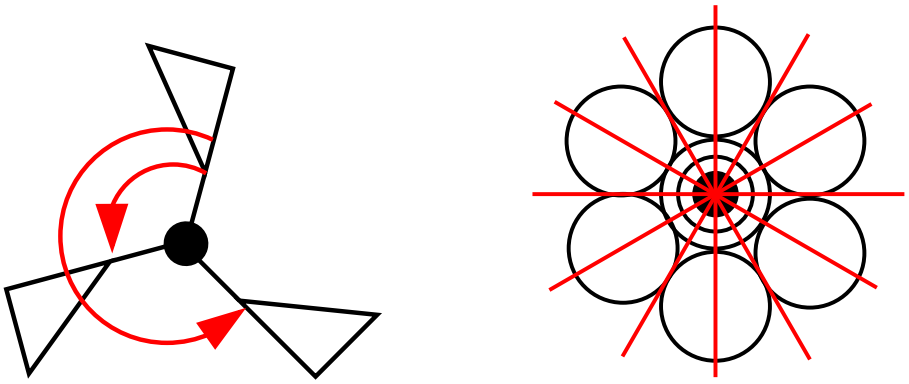
The reviewed C++ source code for this algorithm is available from the web page of this article[1]. Compilation and usage instructions are included in the `README.txt` file of the archive.

Keywords: symmetry detection; blood vessel extraction; image filtering

1 Introduction

Symmetry is an important feature in human perception [18]. Consequently, computer algorithms for automatic symmetry detection have been an area of research for some time, and there are conference workshops entirely devoted to this problem [9, 5]. Formally, a *symmetry* is a geometric transformation under which an image region is invariant. The class of the transformation defines the symmetry type: *reflection*, *rotation*, or *translation*. These symmetry types are not mutually exclusive, but often occur in combination. The object in Figure 1(b), e.g., is invariant under reflections at the six red axes, and also under rotations with angles $\alpha = k \cdot 2\pi/6$ ($k \in \mathbb{Z}$) around the intersection point of the axes. Rotations by multiples of $2\pi/m$ form the *symmetry group* C_m, with the special case C_∞ denoting rotation invariance under arbitrary angles.

[1] https://doi.org/10.5201/ipol.2019.270



(a) pure rotational C_3 symmetry (b) rotational C_6 and axial symmetries

Figure 1: Symmetric object with a pure C_3 rotation symmetry (left), and with a combination of reflection symmetries along each of the red axes and a C_6 rotation symmetry (right).

Algorithms for symmetry detection can either use an entire image as input, or a list of characteristic points that have been picked from the image by some pre-processing algorithm like an edge detector or SIFT [11, 12]. Surveys on symmetry detection algorithms can be found in [10, 13]. The present article deals with an algorithm that takes an entire grayscale image as input and assigns each image point a "symmetry score" that measures how suitable the point is as a symmetry center of a C_{2m} (point reflection) symmetry. As the score value is based on scalar products of gray value gradients at corresponding positions, the method is called *Gradient Product Transform* (GPT) [4]. The image of symmetry score values thus represents a "symmetry transform", and symmetry points correspond to maxima in the symmetry transform image.

The algorithm was originally defined in [3] for square shaped symmetry regions and a simple criterion for discriminating between rotational and axial symmetries was suggested. In [4], this was extended to rectangular regions, a more robust method for determining the size of the symmetry region was introduced, a quadratic discriminant analysis with more features was suggested to discriminate between axial and rotational symmetries, and an optional modification was proposed to make the GPT applicable to blood vessel extraction. Although the GPT was originally devised for 2D symmetry detection, is has also been applied to 3D symmetry detection in voxel data [15].

The present article provides a comprehensive description of the GPT computation and its application for object recognition and vessel detection. For object recognition, we propose a new threshold on a normalized score for identifying actual symmetry points. This problem was circumvented in [3, 4] by simply looking for the highest symmetry score in an image, which always yielded exactly *one* symmetry point, irrespective of the actual number of symmetric objects. We now give a threshold recommendation for detecting more than one (or, which might also happen: none) symmetry. Moreover, we provide a reference implementation of both the GPT and its applications in standard C++. Apart from *libpng*[2] for reading and writing images, the implementation does not depend on any third party libraries.

This article is organized as follows: Section 2 describes the computation of the GPT, Section 3 describes the two applications of detecting C_{2m}-symmetric objects (Section 3.1) and of blood vessel extraction (Section 3.2), Section 4 discusses some run time considerations, Section 5 presents some examples and a typical case of failure, and Section 6 describes the online demo provided on the web page of this article[3].

[2] http://www.libpng.org/pub/png/libpng.html
[3] https://doi.org/10.5201/ipol.2019.270

2 Gradient Product Transform (GPT)

The gradient product transform computes three float images from a grayscale image: an image S representing the symmetry score at each point $\vec{x}$, and two images R_x, R_y representing half the diagonal of the rectangular symmetry region corresponding to the symmetry score value in S. The computation of these three images requires three steps for each image point $\vec{x}$:

1. For a range of radii $\vec{r} = (r_x, r_y)$ with $0 < r_x, r_y \leq r_{max}$, a raw symmetry score $s(\vec{x}, \vec{r})$ is computed as described in Equation (3) below.

2. From these values, an "optimal" radius $\vec{R}(\vec{x}) = (R_x, R_y)$ is determined as

$$\vec{R}(\vec{x}) = \arg_{\vec{r}} \max \left\{ \frac{s(\vec{x}, \vec{r})}{(r_x + r_y)^\alpha} \middle| \max(r_x, r_y) \leq r_{max} \right\} \tag{1}$$

3. The symmetry score is then set to

$$S(\vec{x}) = \frac{s(\vec{x}, \vec{R}(\vec{x}))}{(R_x(\vec{x}) + R_y(\vec{x}))^\alpha} \tag{2}$$

The parameter $\alpha \in [0, 1]$ is typically set to $\alpha = 0.5$. The following subsections describe the details of the computation of $s(\vec{x}, \vec{r})$ and the reasoning for the choice of the parameter α.

2.1 Computation of the Symmetry Score

The raw symmetry score $s(\vec{x}, \vec{r})$ is based on the two gradient images $\vec{G} = (G_x, G_y)$, which are computed from the grayscale image via convolution with Sobel-kernels [6]. Starting with the gradient has the advantage that plain homogeneous regions do not obtain a high symmetry score, even though they are formally perfectly symmetric. The GPT utilizes the two observations that, under point reflection symmetry, the mirrored gradient $\vec{G}'$ is anti-parallel to the gradient $\vec{G}$ (see Figure 2), and that the scalar product $\langle \vec{G}, \vec{G}' \rangle$ is minimal for $\vec{G}' = -\vec{G}$. The raw symmetry score for the rectangle with circumradius $\vec{r} = (r_x, r_y)$ around the center point $\vec{x} = (x, y)$, i.e., the rectangle with upper left $\vec{x} - \vec{r}$ and lower right $\vec{x} + \vec{r}$, is therefore defined as

$$s(\vec{x}, \vec{r}) = -\sum_{d_x=1}^{r_x} \left\langle \vec{G}(d_x, 0), \vec{G}(-d_x, 0) \right\rangle - \sum_{d_y=1}^{r_y} \sum_{d_x=-r_x}^{r_x} \left\langle \vec{G}(d_x, d_y), \vec{G}(-d_x, -d_y) \right\rangle, \tag{3}$$

where $\vec{G}(d_x, d_y)$ denotes the gradient at position $(x + d_x, y + d_y)$. At first sight, it seems unintuitive to use a rectangular symmetry region for a rotational symmetry instead of a circular region or, for raster images, a square region. In real world images, however, an actually circular object usually appears ellipsoidal due to perspective distortion (see Figure 3) and the C_m symmetry only holds approximately. Testing for regions with rectangular bounding boxes thus makes the symmetry score more robust with respect to skew, which was experimentally confirmed in [4].

As the size of the symmetry region is not known in advance, the score (3) must be computed for all radii $1 \leq r_x, r_y \leq r_{max}$. This is less computationally intensive than it might seem at first sight, because Equation (3) only consists of sums. This allows to compute all scores $s(\vec{x}, \vec{r})$ in a single run over the symmetry region with the following recursion formula for $r_x, r_y > 1$

$$\begin{aligned} s\Big(\vec{x}, (r_x, r_y)\Big) \;=\; & s\Big(\vec{x}, (r_x - 1, r_y)\Big) + s\Big(\vec{x}, (r_x, r_y - 1)\Big) - s\Big(\vec{x}, (r_x - 1, r_y - 1)\Big) \\ & - \langle \vec{G}(r_x, r_y), \vec{G}(-r_x, -r_y) \rangle - \langle \vec{G}(r_x, -r_y), \vec{G}(-r_x, r_y) \rangle. \end{aligned} \tag{4}$$

The computation of the GPT is summarized in Algorithm 1.

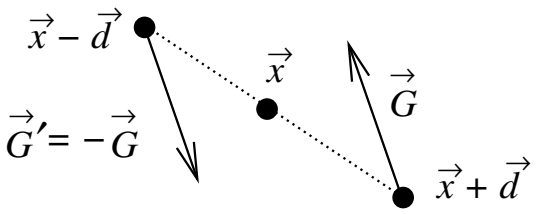
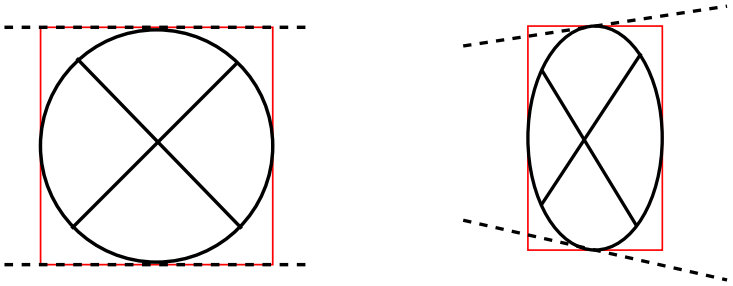


Figure 2: A point reflection at $\vec{x}$ maps the gradient $\vec{G}$ at $\vec{x}+\vec{d}$ onto the gradient $\vec{G}'=-\vec{G}$ at $\vec{x}-\vec{d}$.



(a) front view (b) skewed view

Figure 3: Due to perspective skew, the bounding box of a rotationally symmetric object can become rectangular.

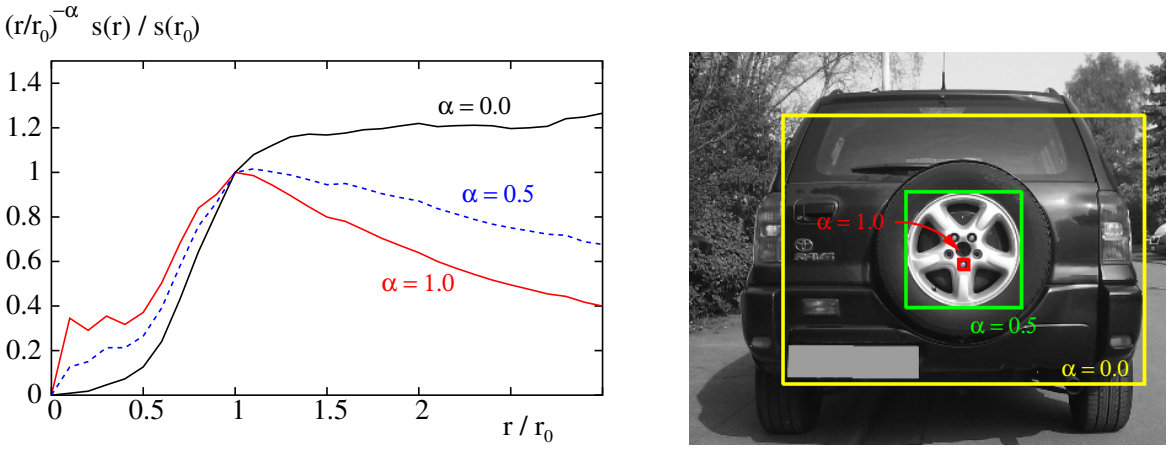
Algorithm 1: Gradient Product Transform

input : grayscale image F, r_{max}, α
output: float images S (symmetry score) and $\vec{R}=(R_x,R_y)$ (symmetry radius)

$G_x \leftarrow$ x-component of gradient of F with Sobel-Operator
$G_y \leftarrow$ y-component of gradient of F with Sobel-Operator
$\vec{G} \leftarrow (G_x, G_y)$
foreach *pixel position* $\vec{x}=(x,y)$ **do**
  $s_{sum} \leftarrow -\left\langle \vec{G}(x,y+1), \vec{G}(x,y-1)\right\rangle$
  foreach *x-component of radius* r_x *with* $0<r_x\le r_{max}$ **do**
    $s_{sum} \leftarrow s_{sum} - \left\langle \vec{G}(x+r_x,y+1), \vec{G}(x-r_x,y-1)\right\rangle$
      $- \left\langle \vec{G}(x+r_x,y), \vec{G}(x-r_x,y)\right\rangle - \left\langle \vec{G}(x+r_x,y-1), \vec{G}(x-r_x,y+1)\right\rangle$
    $\vec{r} \leftarrow (r_x, 1)$
    $s(\vec{r}) \leftarrow s_{sum}$ *// Score for candidate region with radius* $\vec{r}$.
  foreach *radius* $\vec{r}=(r_x,r_y)$ *with* $0<r_x,r_y\le r_{max}$ **do**
    $s(\vec{r}) \leftarrow s(r_x-1,r_y)+s(r_x,r_y-1)-s(r_x-1,r_y-1)$ *// Recursion formula (4).*
      $- \left\langle \vec{G}(x+r_x,y+r_y), \vec{G}(x-r_x,y-r_y)\right\rangle - \left\langle \vec{G}(x+r_x,y-r_y), \vec{G}(x-r_x,y+r_y)\right\rangle$
  $\vec{r} \leftarrow \arg_{\vec{r}}\max\{s(\vec{r})\cdot(r_x+r_y)^{-\alpha} \mid \max(r_x,r_y)\le r_{max}\}$ *// Symmetry radius after Equation (1).*
  $\vec{R}(\vec{x}) \leftarrow \vec{r}$
  $S(\vec{x}) \leftarrow s(\vec{r})\cdot(r_x+r_y)^{-\alpha}$ *// Symmetry score after Equation (2).*

2.2 Symmetry Size Normalization Parameter α

The parameter α has been introduced in [4] in Equations (1) and (2) to compensate for the observation that the raw symmetry score $s(\vec{x},\vec{r})$ tends to increase with the size of $\vec{r}$ due to background noise. Without this scale normalization, the radius with the highest score tends to be quite close to r_{max},



(a) average over all images

(b) an example image

Figure 4: Impact of α on the symmetry radius determination for all images from the data set [3] (left) and a different sample image (right). r_0 is for each image the actual ground truth radius of the symmetry region.

even when the actual symmetric object is much smaller.

To understand this phenomenon, we have computed the symmetry score as a function of the radius r with square shaped regions ($r_x = r_y = r$) for all ground truth symmetry points in the data set [3]. To make the measurements comparable, we have normalized all radii with the ground truth radius r_0, and all scores with the score $s(r_0)$. As can be seen in Figure 4(a) (curve for $\alpha = 0$), the score increases steeply for $r \leq r_0$, as expected, but still increases as r becomes larger than r_0, albeit not as steep. This is simply due to the fact that the number of summands in (3) grows with r so that random fluctuations can increase the sum.

To remedy this problem, the symmetry score needs to be normalized by a factor that depends on $\vec{r}$. Even though the number of summands in (3) is proportional to the area of the symmetry region, which is $r_x \cdot r_y$, only scalar products with large gradients will contribute significantly to the sum. As the gradient is large for edges, and the number of edge points is proportional to the surface of the symmetry region, it seems natural to normalize with $r_x + r_y$. As can be seen from the curve for $\alpha = 1.0$ in Figure 4(a), this indeed leads on average to a maximum score at the actual symmetry size, but it also has the side effect that the score values of small symmetry regions are increased.

To control the trade off between these two effects, we normalize with $(r_x + r_y)^\alpha$, where $\alpha = 0.5$ is a compromise between both effects. The effect of the choice of α on the symmetry radius detection for rectangular regions on a sample image can be seen in Figure 4(b), where we have set r_{max} to the size of the image and for each value of α the highest symmetry is shown.

3 Symmetry Detection

The GPT is merely an image filter that assigns a symmetry score to each pixel. Locating actual symmetry centers requires further analysis of the transformed image. The following subsections describe two different post-processing algorithms of the GPT image for the detection of rotationally symmetric objects, and for extracting symmetry axes in blood vessel images.

3.1 Finding Rotationally Symmetric Objects

In principle, the center point of a symmetric object is the location $\vec{x}$ of a local maximum in the GPT score image S with a high score value $S(\vec{x})$, and the borders of the object are given by the GPT radius images R_x and R_y. In practice, there are however two questions that need to be addressed:

1. Not only rotational symmetries, but also pure axial symmetries can obtain high symmetry scores (see Figure 5(a)). How can we discriminate between a merely reflectional and an actual rotational C_{2m} symmetry?

2. Local maxima are greater than their neighborhood, but are not guaranteed to be high in an absolute sense. What is a reasonable threshold for a "high" symmetry score?

(a) schematic example (b) example image (from [3]) (c) GPT scores of example image

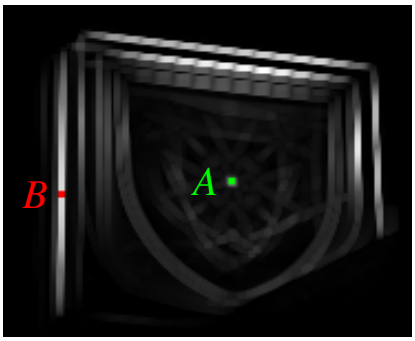
Figure 5: In some cases, C_2 symmetries are reflection symmetries, too. The GPT score image has been converted to grayscale and dilated in order to make high scores better visible.

An example for the first problem can be seen in Figure 5(c): for the axial symmetries the entire symmetry axis tends to obtain high symmetry scores, while, for the rotational symmetry with respect to the center of the cross sign, the symmetry score decays with the distance to the symmetry center independent from the direction. Inspired by this observation, four features for the discrimination between rotational and axial symmetries were defined in [4]. As these features measure local properties in a symmetric neighborhood of the symmetry score maximum, they are computed over square shaped windows. Table 1 lists sample values of these features for two different exemplary points, one representing a rotational symmetry center, and one an axial symmetry center:

Edge directedness. This feature is based on the angle histogram of the gradients of the symmetry transform in a 7×7 window centered at $\vec{x}$. We compute the gradients by applying a Sobel filter on the symmetry transform and build a weighted histogram with 16 bins of the gradient angles. Each angle is weighted in the histogram with the corresponding absolute value of the gradient. The edge directedness is then defined as the highest occurring frequency in this histogram, which should be higher for axial symmetries. In Table 1, this effect can be seen for two sample points.

Skeleton size. Starting from $\vec{x}$, we follow the skeleton until the score falls below $S(\vec{x})/2$. Here, "skeleton" does *not* refer to the skeleton extracted with the algorithm described in section 3.2. Instead it means the point sequence obtained by following in opposite directions the ridge of weakest descent from the local maximum. The feature is then the ratio between the skeleton length and the size $\sqrt{R_x^2 + R_y^2}$ of the symmetry region, where $\vec{R} = (R_x, R_y)$ is the circumradius (half diagonal) of the symmetry region. It should be higher for axial symmetries, as indeed is the case for the example in Table 1.

Antiparallel directions. This feature is based on the gradient of the original grayscale image. We compute the direction histogram with eight bins of all gradients in a window with the radius $\min\{R_x, R_y\}$. Only those gradients are taken into account for which the mirrored gradient is "antiparallel", i.e. the cosine of the angles between the gradients is less than -0.975. The



	point A	point B
edge directedness	0.1203	0.5303
skeleton size	0.0236	1.0000
antiparallelity	0.2095	0.4935
covariance eigenratio	0.8196	0.4389

Table 1: Values of the four features for two typical symmetry center candidate points for the image from Figure 5. Point A represents a rotational symmetry and point B an axial symmetry. The image on the left shows the (dilated) symmetry transform.

feature is the highest relative frequency in the direction histogram. The value for "antiparallel directions" should be lower for rotational symmetries. It should be higher for axial symmetries, as indeed is the case for the example in Table 1.

Covariance eigenratio. For the points in a 7×7 window around (x, y) in the symmetry transform image, we compute the covariance matrix $\mathbf{K}$ as[4]

$$\mathbf{K} = \frac{1}{N} \sum_{dx=-3}^{3} \sum_{dy=-3}^{3} S(x+dx, y+dy) \times \begin{pmatrix} dx\,dx & dx\,dy \\ dy\,dx & dy\,dy \end{pmatrix}, \tag{5}$$

where $S(x, y)$ is the symmetry transform value at (x, y), and the normalization factor N is the sum over all symmetry values in the window. The eigenvalues of $\mathbf{K}$ indicate how strongly the values spread in the direction of the corresponding eigenvector. Consequently, the ratio between the smaller and the larger eigenvalue should be higher for rotational symmetry, as can be seen in the example in Table 1.

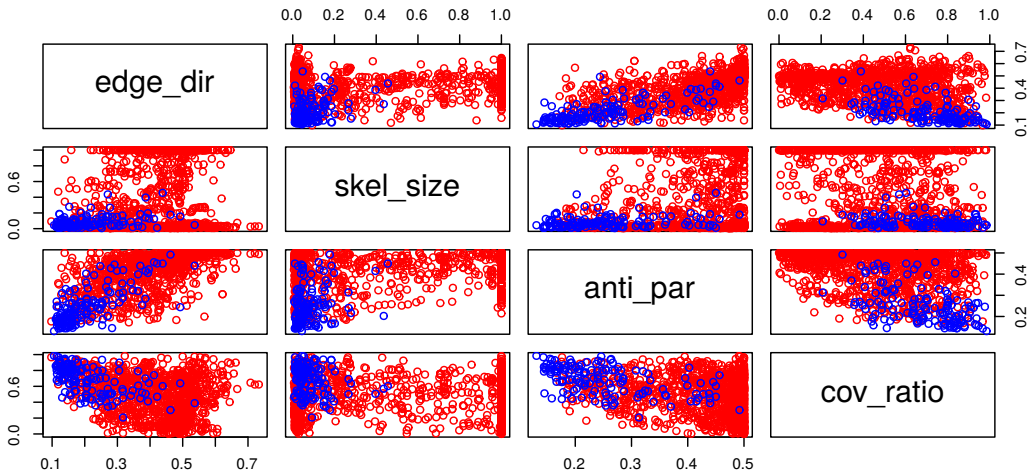


Figure 6: Scatterplots of the four features for the manually labeled rotational (blue) and axial (red) symmetry centers from the dataset [3].

Whilst Table 1 demonstrates the discriminating suitability of the four features for a particular example, the scatterplots in Figure 6 confirm these tendencies for a greater number of symmetry

[4]Beware that dx and dy are not derivatives, but displacements and the matrix $\mathbf{K}$ is thus unrelated to and must not be confused with the Hessian matrix.

center points. These points were taken from the data set published with [3], which includes a list of ground truth points that are labeled as either axial or rotational symmetry centers. In each individual scatterplot of Figure 6, there is, however, considerable overlap between the classes. It is thus better to use a combination of the features for discrimination. In [4], a quadratic discriminant analysis was therefore applied, which decides for the class (axial or rotational) with the largest discriminant function

$$g_i(\vec{\xi}) = -w_i - \|(\vec{\xi} - \vec{\mu}_i) \cdot W_i\|^2, \tag{6}$$

where the index $i \in \{a, r\}$ denotes the class (axial or rotational), and $\vec{\xi}$ is the row vector of the four features in the same order as in the list above. The parameters $\vec{\mu}_i$, w_i and W_i have been determined with the labeled ground truth data from [3] used as training data via the function *qda* as implemented in the MASS package or the R language for statistical computing[5]. The values reported in [4] are:

$$\begin{aligned} w_a &= -15.03338 & w_r &= -21.49041 \\ \vec{\mu}_a &= (0.4334, 0.4975, 0.4417, 0.4352) & \vec{\mu}_r &= (0.1859, 0.0654, 0.2302, 0.7136) \\ W_a &= \begin{pmatrix} -9.3921 & 3.1451 & 7.8727 & 0.8677 \\ 0 & -2.4364 & 0.3522 & 0.1780 \\ 0 & 0 & -18.5275 & 3.1907 \\ 0 & 0 & 0 & 4.3365 \end{pmatrix} & W_r &= \begin{pmatrix} 14.3116 & -6.1912 & 15.5453 & -6.7814 \\ 0 & 20.3183 & 0.3754 & -0.9648 \\ 0 & 0 & -20.5954 & -3.8730 \\ 0 & 0 & 0 & -7.7489 \end{pmatrix} \end{aligned} \tag{7}$$

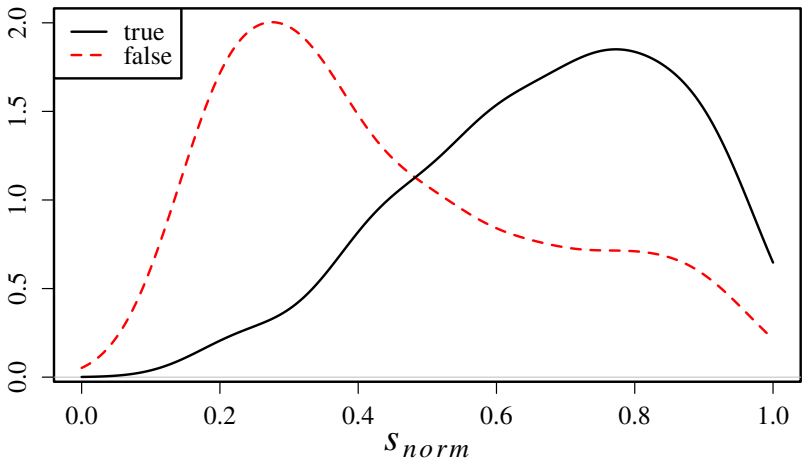


Figure 7: Kernel density plots of the distribution of s_{norm} for true and false high symmetry scores in the data set [3].

The second aforementioned question, i.e., how to define a threshold for the symmetry score representing an actual symmetry, has not yet been discussed in the literature. Here we propose such a threshold on a normalized score value that is guaranteed to fall in the range $[-1, 1]$. The highest symmetry score $s(\vec{x}, \vec{r})$ according to Equation (3) is obtained for a perfect C_{2m} symmetry. In this case, all mirrored gradients are exactly anti-parallel, such that $\langle \vec{G}, \vec{G}' \rangle = -\|\vec{G}\| \cdot \|\vec{G}'\|$. The greatest possible score value is thus

$$s_{max}(\vec{x}, \vec{R}(\vec{x})) = \sum_{d_x=1}^{R_x} \|\vec{G}(d_x, 0)\| \cdot \|\vec{G}(-d_x, 0)\| + \sum_{d_y=1}^{R_y} \sum_{d_x=-r_x}^{R_x} \|\vec{G}(d_x, d_y)\| \cdot \|\vec{G}(-d_x, -d_y)\|, \tag{8}$$

where $\vec{R}(\vec{x}) = (R_x, R_y)$ is the value of the GPT radius images at pixel $\vec{x}$. A normalized score in the range $[-1, 1]$ is thus

$$s_{norm}(\vec{x}) = \frac{s(\vec{x}, \vec{R}(\vec{x}))}{s_{max}(\vec{x}, \vec{R}(\vec{x}))} = \frac{S(\vec{x}) \cdot (R_x(\vec{x}) + R_y(\vec{x}))^\alpha}{s_{max}(\vec{x}, \vec{R}(\vec{x}))} \quad \in [-1, 1]. \tag{9}$$

[5] https://www.r-project.org/

To choose a reasonable threshold for s_{norm}, we took all images of the dataset [3] and extracted the five greatest local maxima in the symmetry score image $S(\vec{x})$ that were classified as rotational symmetry centers by the discriminant analysis (6) and (7). We have manually labeled these points as correctly or wrong identified and estimated the distribution of s_{norm} for the different classes with a kernel density estimator [16], as shown in Figure 7. From these data, we conclude that the threshold on s_{norm} should be at least 0.5 for points reported as a symmetry center. It should be noted that this threshold is not meant to discriminate between axial and rotational symmetries, but for suppressing responses from region not representing symmetries at all. In our reference implementation (see Section 6), the threshold defaults to 0.6 and can be changed by a command line option.

It should be noted that, for the purpose of finding symmetry center candidate points as local maxima in the score image, the symmetry score $S(\vec{x})$ cannot be replaced by $s_{norm}(\vec{x})$, because the normalization (9) has the side effect of increasing the score values for regions with low contrast and henceforth with small gradients. We therefore proceed as follows for detecting symmetry center points:

1. Find all local maxima $\vec{x}$ within a $k \times k$ window with $k = 5$[6] in the GPT score image $S(\vec{x})$. These points are subsequently processed in the descending order of their score values.
2. Test whether a point is a rotational symmetry with the quadratic discriminant (6).
3. Test whether $s_{norm}(\vec{x}) \geq t$, where $0 < t < 1$ is some threshold. If this is not the case, no further points are examined.

In order to avoid the detection of small symmetry regions inside a larger region (for example screw heads inside a wheel), we optionally ignore points lying inside already detected symmetry regions. The resulting symmetric object detection is formally summarized in Algorithm 2, and Figure 8 shows its result on a sample image.

Algorithm 2: Detection of rotationally symmetric objects

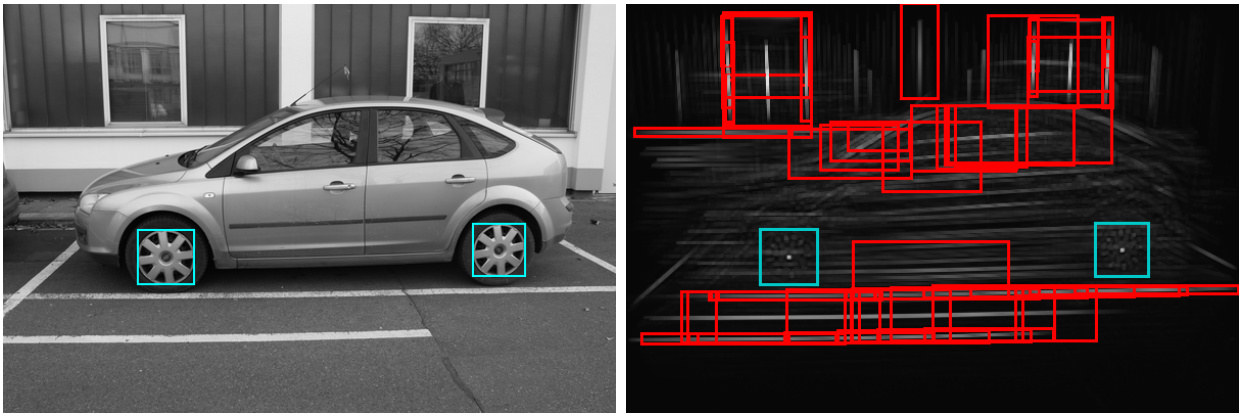
input : float images S (GPT score) and $\vec{R} = (R_x, R_y)$ (GPT radius) and $\vec{G} = (G_x, G_y)$ (gradient image), window size k, flag *skipinside*, threshold t
output: symmetry regions $A = \{(\vec{x}_1, \vec{r}_1), \ldots\}$

$A \leftarrow \{\}$, $P \leftarrow \{\}$
foreach *pixel position* $\vec{x} = (x, y)$ **do**
  if $S(\vec{x})$ *is highest score in* $k \times k$ *window around* $\vec{x}$ **then** $P \leftarrow P \cup \{\vec{x}\}$
sort P in descending order of $S(\vec{x})$
foreach $\vec{x} \in P$ **do**
  if *skipinside* **and** $\vec{x}$ *contained in a region from* A **then continue**
  if $g_a(\vec{x}) > g_r(\vec{x})$ **then continue** // *see Equation (6), requires* S, $\vec{R}$ *and* $\vec{G}$
  if $s_{\text{norm}}(\vec{x}) < t$ **then break** // *see Equation (8), requires* S, $\vec{R}$ *and* $\vec{G}$
  $A \leftarrow A \cup \{(\vec{x}, \vec{R}(\vec{x}))\}$

3.2 Extracting Vessel Skeletons

The problem of blood vessel extraction occurs for many types of medical images, e.g. for angiographs, fundus photographs, or MRT-TOF datasets. As blood vessels are locally C_2 symmetric (see Figure 5(a)), the GPT assigns high scores to the medial axis of the vessels. When the GPT symmetry

[6]The value $k = 5$ is somewhat arbitrary and can be optionally changed on the command line (see section 6). We have not noticed much differences on our test images, though, whether it is set to 5, 7, or 11.



(a) image with detected symmetries (b) dilated GPT scores with local maxima regions

Figure 8: An example image with the GPT score image and the symmetric objects (cyan) detected with Algorithm 2. Red rectangles have been sorted out by the quadratic discriminant analysis. For this 600×400 image, we used $r_{max} = 80$ and set the symmetry threshold to $t = 0.6$.

(a) dark blood vessels (from [17]) (b) original GPT score (c) GPT score with $\Theta(+)$

Figure 9: Effect of Heaviside's step function on the GPT score image ($r_{max} = 3$). The GPT score images have been converted to grayscale and their histograms were equalized in order to make the high scores better visible.

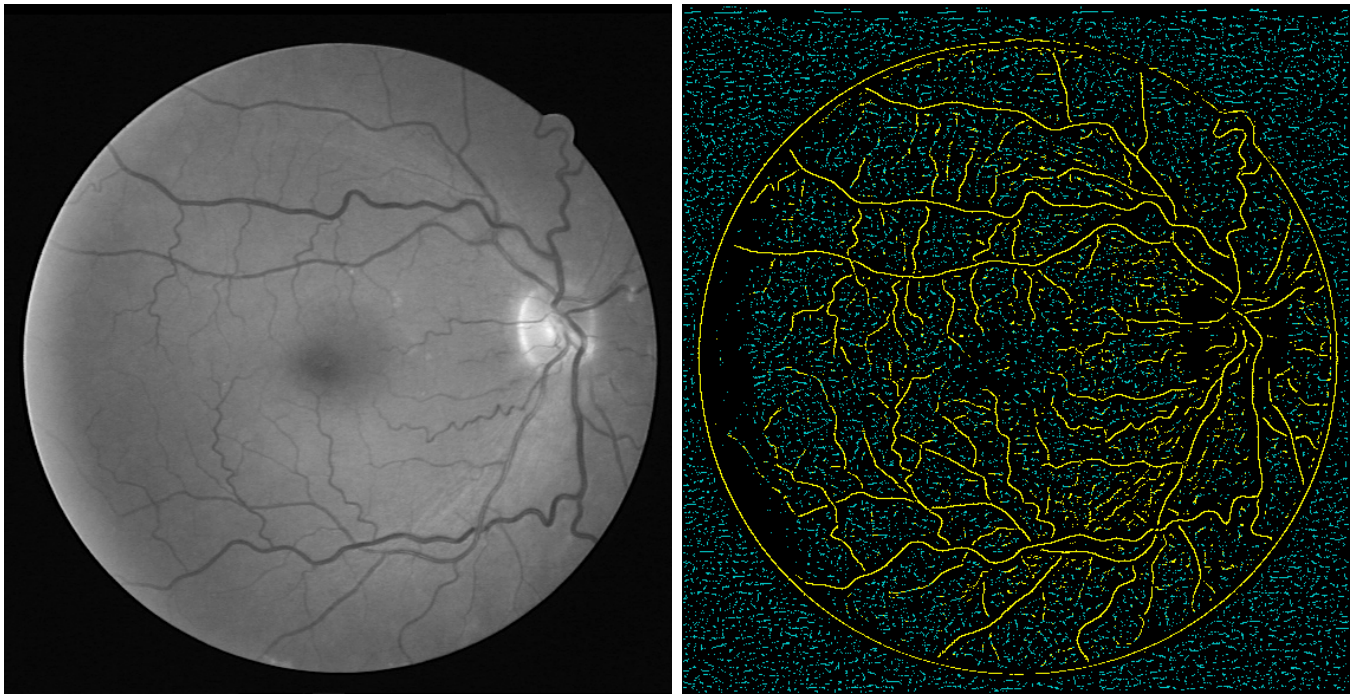
score image is interpreted as a height map, the problem of vessel skeleton extraction is thus transformed into the problem of ridge extraction. In practice, there are however two questions that need to be addressed:

1. Parallel vessels lead to a high score in the middle of the background between them, too. How can we ensure that only symmetries in the foreground are detected?

2. How do we extract the dominant ridges?

An example for the first problem can be seen in Figure 9(b): many high GPT scores occur in the background, too. We can, however, easily circumvent this problem by suppressing contributions to the symmetry score for which the gradient points into the wrong direction with respect to the symmetry center. We do this by introducing Heaviside's step function Θ (zero for negative arguments, and one for positive arguments) as a factor in each summand of the symmetry score in order to suppress contributions from gradients pointing to (or away from) the symmetry center

$$s(\vec{x}, \vec{r}) = - \sum_{\substack{\vec{d}=(d_x,d_y)\neq\vec{0} \\ 0\leq d_x,d_y\leq r_{max}}} \left\langle \vec{G}(\vec{x}+\vec{d}), \vec{G}(\vec{x}-\vec{d}) \right\rangle \cdot \Theta\left(\pm \left\langle \vec{d}, \vec{G}(\vec{x}+\vec{d}) \right\rangle\right) \cdot \Theta\left(\pm \left\langle -\vec{d}, \vec{G}(\vec{x}-\vec{d}) \right\rangle\right). \quad (10)$$



(a) fundus image (from [17])

(b) extracted GPT score ridges

Figure 10: Symmetry score ridges (yellow) extracted with Algorithm 3 from a fundus image. Blue pixels match criteria (11) and (12), but have been filtered out by threshold (14). The GPT has been applied with $r_{max} = 3$ and with $\Theta(+)$.

The sign of the argument in Θ determines which objects are detected. For a dark foreground, the plus sign should be used because it has the effect that only gradients pointing outwards contribute to the sum. The result can be seen in Figure 9(c).

The second problem, i.e., ridge detection, naturally occurs in the context of geological terrain maps, and different algorithms have been proposed based on simulating dropping water [19, 8], on building and pruning neighborhood graphs [2, 1], or on estimating local curvature features [17, 7]. For demonstrating the application of the GPT to vessel medial axis extraction, we have implemented the curvature based method by Staal et al., which defines the following condition for a ridge point $\vec{x}$ [17]

$$1 = -\frac{1}{2}\,\mathrm{sign}(\lambda_1) \cdot \left|\mathrm{sign}\left(\langle \vec{G}_S(\vec{x}+\vec{v}_1), \vec{v}_1\rangle\right) - \mathrm{sign}\left(\langle \vec{G}_S(\vec{x}-\vec{v}_1), \vec{v}_1\rangle\right)\right|, \tag{11}$$

where λ_1 is the eigenvalue with the largest absolute value of the Hessian matrix at position $\vec{x}$ of the GPT score image, $\vec{v}_1$ is the corresponding eigenvector (normalized to $\|\vec{v}_1\| = 1$), and $\vec{G}_S$ is the gradient of the GPT score image. The gradients and the Hessian matrix are computed with the Sobel operator. Note that, unlike Staal et al., we do not compute these derivatives on the original grayscale image, but on the GPT score image.

Application of criterion (11) does not result in a one pixel wide skeleton, but in a dilated skeleton at curved positions. We therefore added as an additional criterion for an actual ridge point the criterion of the first step of the profile-recognition by Chang and Sinha [2]. This imposes the condition that the score value at the center is greater than the values of opposite neighbors for at least one combination of neighbors (N-S, NE-SW, E-W, SE-NW), i.e.

$$S(\vec{x}+\vec{d}) < S(\vec{x}) \quad \text{and} \quad S(\vec{x}-\vec{d}) < S(\vec{x}), \tag{12}$$

for at least one of the step vectors $\vec{d} \in \{(0,1),(1,1),(1,0),(1,-1)\}$. These two conditions still lead to many ridges in regions with low GPT score values. To suppress these responses, we additionally

Algorithm 3: Medial axis extraction of vessels

input : float images S (GPT score) and $\vec{R} = (R_x, R_y)$ (GPT radius) and m (mean gradient absolute value of original image)
output: medial axis points $P = \{\vec{x}_1, \ldots\}$

$P \leftarrow \{\}$
$G_x \leftarrow$ Sobel filter of S in x-direction *// gradient $\vec{G}$ of GPT score*
$G_y \leftarrow$ Sobel filter of S in y-direction
$H_{xx} \leftarrow$ Sobel filter of G_x in x-direction *// Hessian H of GPT score*
$H_{yy} \leftarrow$ Sobel filter of G_y in y-direction
$H_{xy} \leftarrow H_{yx} \leftarrow$ Sobel filter of G_x in y-direction
foreach *pixel position* $\vec{x} = (x, y)$ **do**
  $r_x \leftarrow R_x(\vec{x})$ and $r_y \leftarrow R_y(\vec{x})$
  if $S(\vec{x}) \cdot (r_x + r_y)^\alpha < 2 \cdot \max(r_x, r_y) \cdot m^2$ **then continue** *// threshold (14)*
  compute eigenvalues and eigenvectors $H\vec{v}_i = \lambda_i \vec{v}_i$ where $|\lambda_1| > |\lambda_2|$ and $\|\vec{v}_i\| = 1$
  if $1 \neq -\frac{1}{2}\,\mathrm{sign}(\lambda_1) \cdot \left|\mathrm{sign}\left(\langle \vec{G}_S(\vec{x} + \vec{v}_1), \vec{v}_1\rangle\right) - \mathrm{sign}\left(\langle \vec{G}_S(\vec{x} - \vec{v}_1), \vec{v}_1\rangle\right)\right|$ **then**
    continue *// ridge criterion (11) matches*
  ridgefound $\leftarrow$ False
  foreach $\vec{d} \in \{(0,1), (1,1), (1,0), (1,-1)\}$ **do**
    if $S(\vec{x} + \vec{d}) < S(\vec{x})$ **and** $S(\vec{x} - \vec{d}) < S(\vec{x})$ **then**
      ridgefound $\leftarrow$ True *// ridge criterion (12) matches*
  if not *ridgefound* **then continue**
  compute $s_{norm}(\vec{x})$ after Equation (9)
  if $s_{\mathrm{norm}}(\vec{x}) \geq 0.4$ **then** $P \leftarrow P \cup \{\vec{x}\}$ *// threshold (13)*

apply two thresholds on the GPT score value

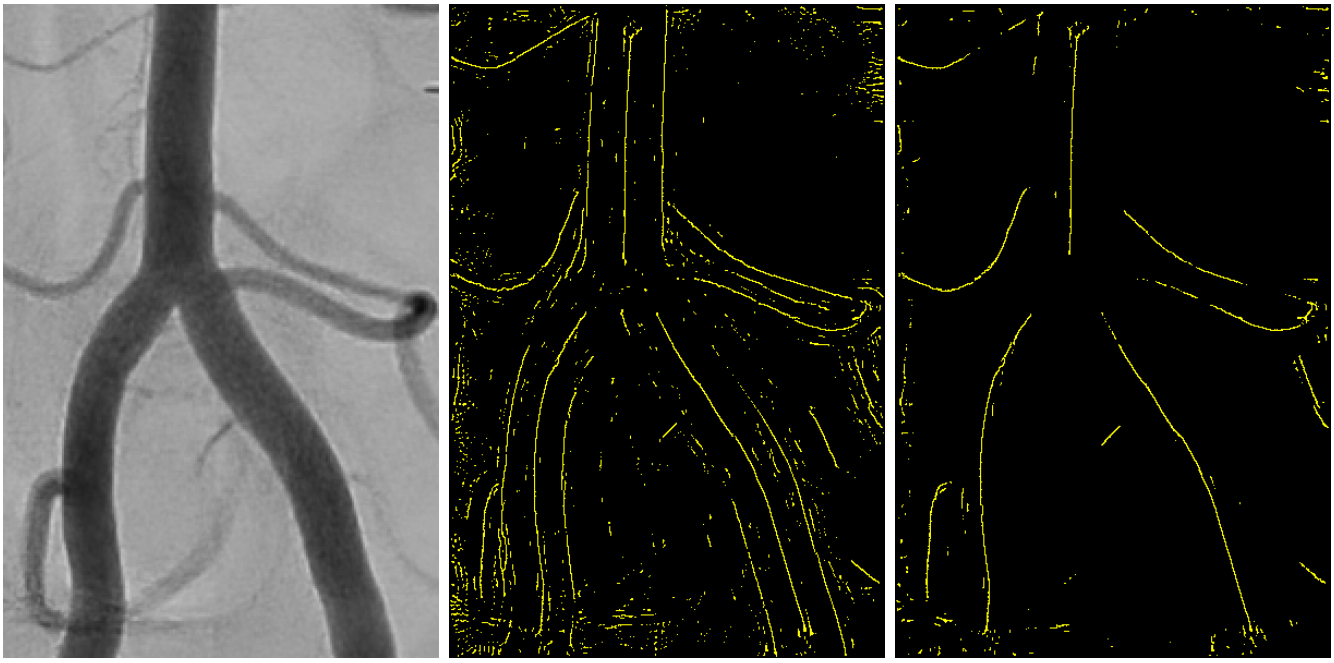
$$s_{norm}(\vec{x}) \geq 0.4, \tag{13}$$

$$S(\vec{x}) \cdot (R_x(\vec{x}) + R_x(\vec{x}))^\alpha \geq 2 \cdot \max(R_x(\vec{x}), R_y(\vec{x})) \cdot \mathrm{mean}(\|\vec{G}\|)^2, \tag{14}$$

where the mean absolute value is computed over the entire gradient image of the input image. The first threshold ensures that the returned ridge points are actual symmetries. The second threshold is only added to suppress responses from homogeneous regions, as can be seen in Figure 10(b). The reasoning behind the second threshold is that a symmetry axis should have significant contributions from at least the two longer border sides, which correspond to $2 \cdot \max(R_x, R_y)$ in Equation (10). The resulting algorithm for ridge extraction is formally summarized in Algorithm 3, and Figure 10 shows its result on a sample image. Note that the ridge criteria are applied in Algorithm 3 in such an order that the most computationally expensive test, which includes the computation of s_{norm}, is done last, and thus least frequently.

4 Runtime Considerations

There are a number of $O(n)$ operations in the GPT algorithms, where n is the number of image pixels, e.g. the Sobel operators in Algorithm 1, the location of local maxima in Algorithm 2, or the application of ridge criteria in Algorithm 3. All these are dwarfed, however, by the GPT score computation because the computation of all scores up to a radius r_{max} at an image point (x, y)



(a) angiography (b) GPT ridges (`-which all`) (c) GPT ridges (`-which dark`)

Figure 11: Examples showing the results of the GPT in combination with Algorithm 3. The option `-which dark` activates Heaviside's step function in Equation (10). For this 400×600 image, we have used $r_{max} = 80$.

according to Equations (3) and (4) requires $O(r_{max}^2)$ additions. The total runtime of the GPT is thus $O(nr_{max}^2)$.

It should be noted that, when the number of image pixels is increased by a factor a, the maximum symmetry radius r_{max} needs to be increased by a factor $\sqrt{a}$, too, in order to detect objects of the same maximum real world size. This means that the runtime is even $O(n^2)$ when large objects are to be detected. In order to speed up the runtime, we therefore use parallelization and (optional) downscaling.

As the GPT score is defined independently for each point, the computation of the GPT score and radius images can be performed independently in parallel. We have used OpenMP[7] and let the user specify the number of parallel threads at compilation time in the CMakefile. With the choice of four threads on an Intel Core i7-4770 CPU @ 3.40GHz, the runtime for detecting the symmetric objects in the 600×400 image in Figure 8(a) with $r_{max} = 80$ and $t = 0.75$ was about 1.95s.

When both the image size and r_{max} are large, it is reasonable to assume that the user is not interested in responses of very small region size. In this case, the runtime can be reduced considerably by downscaling the image before applying the GPT, and then remap the detected symmetry points and regions on the original input image. In our reference implementation, we do this when $\sqrt{n_{cols} \times n_{rows} \times r_{max}^2} > 10^5$. In this case the image is scaled down such that this value is 10^5. This behavior can be switched off by the command line option `-noscale`.

5 Examples and Cases of Failure

Figure 11 is an example for the vessel medial axis extraction: as the vessel foreground is dark, the GPT variant (10) with the plus sign should be used (command line option `-which dark`). Except from minor noise, Algorithm 3 extracts the medial axis even for vessels of different sizes. When

[7] https://www.openmp.org/

specifically smaller vessels are to be detected, r_{max} should be chosen smaller. Although rotational object detection is of no interest in this case, it is interesting to note that, in this case, no rotational symmetry is found by Algorithm 2 with $t = 0.6$, as it should be.

An example, for which Algorithm 2 works very well for detecting a symmetric object, can be seen in Figure 12(a): the global maximum in the GPT score image is the center of the dartboard, and its normalized score is $s_{norm} \approx 0.69$. With $t = 0.5$, no further rotationally symmetric objects are detected. Figure 12(c) shows the effect of the option *skipinside* in Algorithm 2: with this option, only the outermost symmetry including both bird's eyes is detected, while otherwise additional symmetries inside are detected.

Figure 12(f) shows a typical example, for which the algorithm fails: both the background pattern on the church wall and the window rosette in the center show rotational symmetries, but the contrast in the background is much greater, which leads to higher gradients and thus to higher GPT scores. Although the background pattern actually shows more than one symmetry (translational, rotational, and axial), there are clear local maxima at the brick centers which make the quadratic discriminant analysis decide for "rotational symmetry". Other problems can occur due to perspective skew, as shown in Figure 13. Although the outline of the clock is transformed to an ellipse in Figure 13(c), the interior is no longer C_2 symmetric and thus does not contribute sufficiently to the symmetry score.

6 Online Demo and Command Line Program

The source code of a command line program `gptsymmetry` that implements this algorithm is provided on the web page of this article[8]. On the same website, there is also an online demo built on top of this command line program. Users can choose among several pre-uploaded images or upload their own image in the PNG file format. Color images are automatically converted to grayscale by averaging over the three color channels. Here is a list of the options for the online demo with the corresponding command line parameters of the reference implementation given in parentheses:

Maximum radius (*-r*). The maximum radius r_{max} of the symmetry region in Algorithm 1.

Threshold (*-t*). The threshold on the normalized score s_{norm} in Algorithm 2. When negative (default), it is set to 0.6 if *which*=all, and to 0.4 otherwise. When $t \geq 1$, only the first rotational symmetry is reported.

Object foreground (*-which*). Whether *all* symmetries are to be detected or only those of *light* (or *dark*) objects. Defaults to *all*.

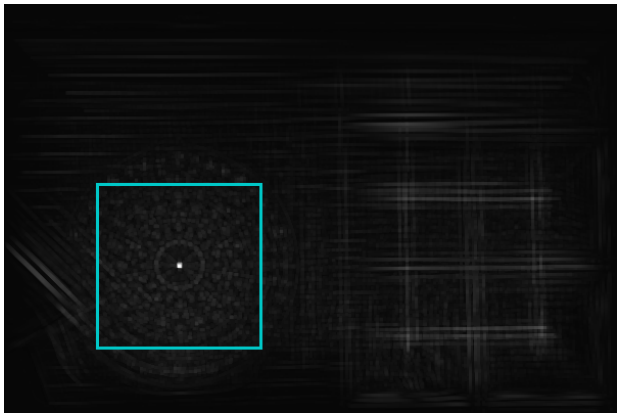
Exponent α (*-alpha*). Exponent for normalizing the symmetry score in Algorithm 2 with $(r_x + r_y)^\alpha$. Defaults to 0.5.

Skipinside (*-skipinside*). When set, symmetries with lower symmetry score inside larger symmetry region are not reported in Algorithm 2.

The online demo creates and displays four images: the GPT score image converted to grayscale, the ridges extracted with Algorithm 3, the symmetry regions detected by Algorithm 2 classified as axial (red) or rotational (cyan) overlaid on the GPT score image, and the rotational symmetries detected by Algorithm 2 overlaid on the input image. The detected symmetries are printed in CSV format separated by semicolons as

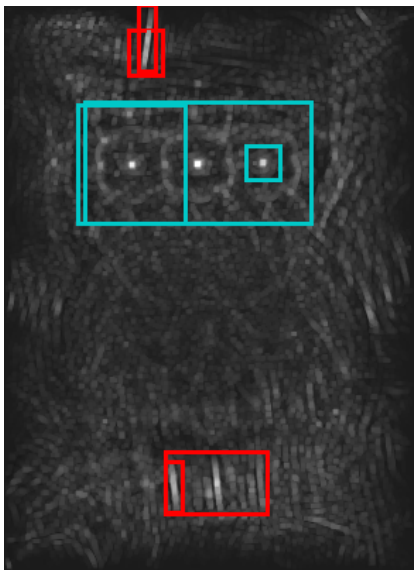
[8] https://doi.org/10.5201/ipol.2019.270

(a) dartboard

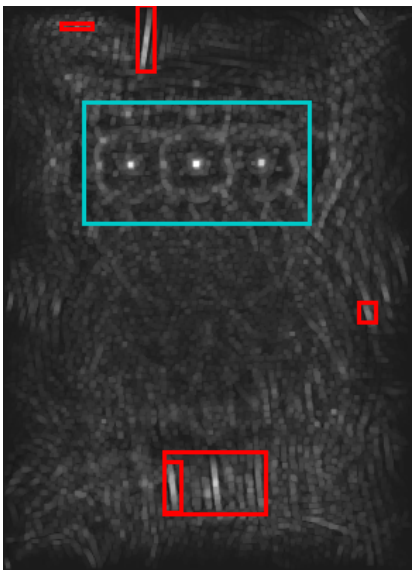


(b) GPT scores with local maxima regions

(c) metal bird

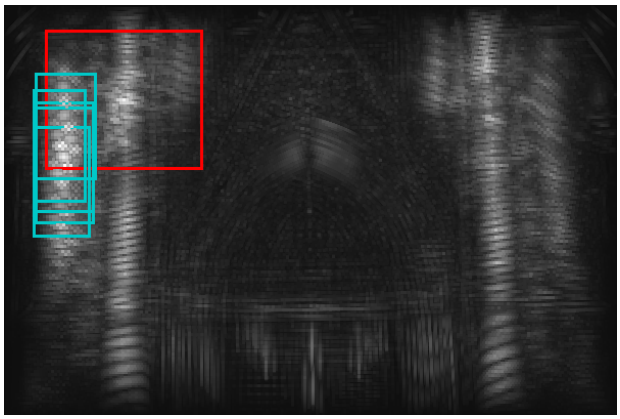


(d) all local maxima



(e) with *skipinside*

(f) church window



(g) GPT scores with local maxima regions

Figure 12: Examples showing the results of the GPT in combination with Algorithm 2. The GPT score images have been dilated. Cyan rectangles have been detected as rotational symmetries, and red rectangles have been sorted out by the quadratic discriminant analysis. All images had height 400, and we used $r_{max} = 80$ with a symmetry threshold of $t = 0.6$.

(a) front view (b) clock detected (c) skewed view (d) nothing detected

Figure 13: Example showing the effect of strong perspective skew on symmetry detection with Algorithm 2. Cyan rectangles have been detected as rotational symmetries. Both images had height 400, and we used $r_{max} = 80$ with a symmetry threshold of $t = 0.6$.

```
rot;x;y;rx;ry;S;s_norm
0;138;69;43;61;10402380.665264;0.806487
0;100;76;5;39;7279246.983320;0.992713
1;159;247;28;27;7207329.751625;0.924952
1;485;240;26;26;6579806.023407;0.876632
0;176;73;5;41;6529087.512724;0.989594
```

where *rot* indicates whether this was classified as a rotational (1) or axial symmetry (0), (x, y) and (rx, ry) denote the symmetry center and radius of the symmetry region, S is the GPT score, and s_{norm} is the score normalized between zero and one.

The command line program `gptsymmetry` has the following additional options not supported by the online demo:

`-k` ***k.*** Radius of the window for local maxima search. Defaults to 5.

`-bw` ***borderwidth.*** Width of the borders of symmetry regions drawn in result image. Defaults to 1.

`-o` ***outpng.*** Draw detected symmetry as rectangle in file *outpng*.

`-notrace.` do not write images of interim steps with prefix *trace-*. This will speed the program up.

`-noscale.` Do not scale down large images with large radius.

7 Conclusion

The Gradient Product Transform is an algorithm that has already been reported in the literature as a useful tool for symmetry detection or vessel skeleton extraction [3, 4]. The score normalization method presented in this article allows for absolute thresholds, which are helpful in both use cases.

A drawback of the algorithm is that it only uses the grayscale information of the image, but does not utilize color information. Another drawback is that all radii up to a user provided value r_{max} are tested, which becomes slow for the combination of a large image size with a large r_{max}. We solve this problem by scaling down the image in this case. A multiresolution approach based on a Gaussian image pyramid [6] would be more flexible and would save the user from specifying r_{max}. This would, however, introduce the problem of how to order the local maxima from different scales in Algorithm 2 and how to select the most salient symmetries. This would require extensive further experiments on new additional ground truth data, and it is unclear whether the results would justify the effort.

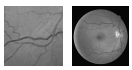
In its current form, the GPT already is a useful tool for symmetry detection. As the reference implementation in standard C++ coming with this article does not rely on third party libraries (except for *libpng*), it provides an effortless way to try out the algorithm for particular applications.

Acknowledgment

The authors are grateful to Tobias Bolten for creating the symmetry dataset [3], to Bram van Ginneken for making the DRIVE database [17] available to us, and to Christian Neumann for providing to us an angiography image from the study [14]. Moreover, we thank the anonymous reviewers for their valuable comments.

Image Credits


Tobias Bolten, partly from the dataset described in [3]


Joes Staal et al., from the dataset DRIVE ("Digital Retinal Images for Vessel Extraction") [17]


Christoph Dalitz


Christian Neumann, from the study [14]

References

[1] S. Bangay, D. de Bruyn, and K. Glass, *Minimum spanning trees for valley and ridge characterization in digital elevation maps*, in Proceedings of the 7th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa, AFRIGRAPH, ACM, 2010, pp. 73–82. https://doi.org/10.1145/1811158.1811171.

[2] Y-C. Chang and G. Sinha, *A visual basic program for ridge axis picking on DEM data using the profile-recognition and polygon-breaking algorithm*, Computers & Geosciences, 33 (2007), pp. 229–237. https://doi.org/10.1016/j.cageo.2006.06.007.

[3] C. Dalitz, R. Pohle-Fröhlich, and T. Bolten, *Detection of symmetry points in images*, in International Conference on Computer Vision Theory and Applications (VISAPP), 2013, pp. 577–585. http://dx.doi.org/10.5220/0004179405770585.

[4] C. Dalitz, R. Pohle-Fröhlich, F. Schmitt, and M. Jeltsch, *The gradient product transform for symmetry detection and blood vessel extraction*, in International Conference on

Computer Vision Theory and Applications (VISAPP), 2015, pp. 177–184. http://dx.doi.org/10.5220/0005234101770184.

[5] C. Funk, S. Lee, M.R. Oswald, S. Tsogkas, W. Shen, A. Cohen, S. Dickinson, and Y. Liu, *2017 ICCV challenge: Detecting symmetry in the wild*, in Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 1692–1701. https://doi.org/10.1109/ICCVW.2017.198.

[6] R.C. Gonzalez and R.E. Woods, *Digital Image Processing*, Prentice-Hall, New Jersey, 2nd ed., 2002.

[7] S.N. Kalitzin, J. Staal, B.M. ter Haar Romeny, and M.A. Viergever, *A computational method for segmenting topological point-sets and application to image analysis*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 23 (2001), pp. 447–459. https://doi.org/10.1109/34.922704.

[8] S. Koka, K. Nomaki, K. Sugita, K. Tsuchida, and T. Yaku, *Ridge detection with a drop of water principle*, in ACM SIGGRAPH ASIA, 2010, p. 34. https://doi.org/10.1145/1900354.1900392.

[9] J. Liu, G. Slota, G. Zheng, Z. Wu, M. Park, S. Lee, I. Rauschert, and Y. Liu, *Symmetry detection from real world images competition 2013: Summary and results*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013, pp. 200–205. https://doi.org/10.1109/CVPRW.2013.155.

[10] Y. Liu, H. Hel-Or, C.S. Kaplan, and L. Van Gool, *Computational symmetry in computer vision and computer graphics*, Foundations and Trends in Computer Graphics and Vision, 5 (2010), pp. 1–195. https://doi.org/10.1561/0600000008.

[11] D.G. Lowe, *Distinctive image features from scale-invariant keypoints*, International Journal of Computer Vision, 10 (2004), pp. 91–110. https://doi.org/10.1023/B:VISI.0000029664.99615.94.

[12] G. Loy and J.O. Eklundh, *Detecting symmetry and symmetric constellations of features*, in European Conference on Computer Vision (ECCV), 2006, pp. 508–521. https://doi.org/10.1007/11744047_39.

[13] N.J. Mitra, M. Pauly, M. Wand, and D. Ceylan, *Symmetry in 3D geometry: Extraction and applications*, in Computer Graphics Forum, vol. 32, 2013, pp. 1–23.

[14] C. Neumann, K. Tönnies, and R. Pohle-Fröhlich, *AngioUnet - a convolutional neural network for vessel segmentation in cerebral DSA series*, in International Conference on Computer Vision Theory and Applications (VISAPP), 2018, pp. 331–338. https://doi.org/10.5220/0006570603310338.

[15] R. Pohle-Fröhlich and D. Stalder, *3D-Symmetrietransformation zur Gefäßsegmentierung in MRT-TOF-Daten*, in Bildverarbeitung für die Medizin, Springer, 2014, pp. 144–149.

[16] S.J. Sheather and M.C. Jones, *A reliable data-based bandwidth selection method for kernel density estimation*, Journal of the Royal Society series B, 53 (1991), pp. 683–690.

[17] J. Staal, M.D. Abramoff, M. Niemeijer, M.A. Viergever, and B. van Ginneken, *Ridge-based vessel segmentation in color images of the retina*, IEEE Transactions on Medical Imaging, 23 (2004), pp. 501–509. https://doi.org/10.1109/TMI.2004.825627.

[18] P.A. Van der Helm, *Symmetry perception*, in Oxford handbook of perceptual organization, Johan Wagemans, ed., Oxford University Press, Oxford, UK, 2015, pp. 108–128. https://doi.org/10.1093/oxfordhb/9780199686858.013.056.

[19] R. Yokoyama, A. Kureha, T. Motohashi, H. Ogasawara, T. Yaku, and D. Yoshino, *Geographical concept recognition with the octgrid method for learning geography and geology*, in Seventh IEEE International Conference on Advanced Learning Technologies (ICALT 2007), 2007, pp. 470–471. https://doi.org/10.1109/ICALT.2007.150.