Calculating ellipse overlap areas

Gary B. Hughes · Mohcine Chraibi

Abstract We present an approach for finding the overlap area between two ellipses that does not rely on proxy curves. The Gauss-Green formula is used to determine a segment area between two points on an ellipse. Overlap between two ellipses is calculated by combining the areas of appropriate segments and polygons in each ellipse. For four of the ten possible orientations of two ellipses, the method requires numerical determination of transverse intersection points. Approximate intersection points can be determined by solving the two implicit ellipse equations simultaneously. Alternative approaches for finding transverse intersection points are available using tools from algebraic geometry, e.g., based on solving an Eigen-problem that is related to companion matrices of the two implicit ellipse curves. Implementations in C of several algorithm options are analyzed for accuracy, precision and robustness with a range of input ellipses.

Keywords Ellipse segment · Ellipse sector · Ellipse area · Ellipse overlap · Quartic formula · Ellipse · Algorithm

1 Introduction

Ellipses are useful in many applied scenarios, and in widely disparate fields. In our research, we have encountered a common need for efficiently calculating the overlap area between two ellipses. In one case, the design for a solar calibrator onboard an orbiting satellite required an efficient routine for finding ellipse overlap areas. In a more down-to-earth setting, calculating ellipse overlap areas is useful for modeling pedestrian dynamics. The approach described in [4] surrounds each pedestrian by an elliptical footprint area that the model uses to anticipate obstacles and other pedestrians in or near the intended path. A force-based model produces a repulsive force between ellipses, causing the pedestrians to slow down or change course when the exclusion force becomes large. To calibrate the strength of the repulsive force a quantity describing the amount of overlapping between two ellipses was defined, which requires calculating the overlap area between many different ellipses in general orientations and the overlap algorithm must be efficient, so as not to bog down the simulation.

Overlap area between two ellipses can be determined with high accuracy and precision by Monte Carlo integration (e.g., [17]), but the method is numerically intensive. More efficient approaches for determining ellipse areas based on approximating the ellipse curves with polygons [9,11], adaptive polygons [10], polytopes [1], and polyarcs [8] have also been described. For these methods, accuracy of the overlap area value is governed by the number of sample points used to represent the original curve with a polygon or other proxy curve. Other than Monte Carlo integration, we have yet to find any ellipse overlap area algorithms in the peer-reviewed literature that do not rely on use of proxy curves. An un-published algorithm that does not use proxy curves is available online [5].

– In this paper, we provide an algorithm that determines the overlap area between two general ellipses without using proxy curves. We first find the area of an ellipse segment, which is the area between a secant line and the ellipse

G. B. Hughes (✉)
California Polytechnic State University, San Luis Obispo, CA, USA
e-mail: gbhughes@calpoly.edu

M. Chraibi
Forschungszentrum JüLich, JüLich Supercomputing Centre,
Jülich, Germany
e-mail: m.chraibi@fz-juelich.de

boundary. The segment algorithm then forms the basis of an application for calculating the overlap area between two general ellipses, using points of intersection between the two ellipses to identify appropriate segment areas. Intersection points can be found by solving the two implicit ellipse equations simultaneously. Alternative methods for determining intersection points can be adopted from Algebraic Geometry, such as solving an Eigen-problem that is formed from companion matrices of the two implicit ellipse curves. Accuracy of the area estimate is determined by accuracy and precision of calculated intersection points. The algorithm is implemented in C and tested with a range of input ellipses.

2 Ellipse equations and areas

2.1 Ellipse parametric and implicit polynomial equations

Consider an ellipse that is centered at the origin, with its axes aligned to the coordinate axes. If the semi-axis length along the x-axis is A, and the semi-axis length along the y-axis is B, then the ellipse is defined by a locus of points that satisfy the implicit polynomial Eq. (1a), or parametrically as in Eq. (1b):

$$\frac{x^2}{A^2} + \frac{y^2}{B^2} = 1 \tag{1a}$$

$$\left.\begin{array}{l} x(t) = A \cdot cos(t) \\ y(t) = B \cdot sin(t) \end{array}\right\} \quad 0 \le t \le 2\pi \tag{1b}$$

More generally, a rotated ellipse can be defined parametrically using Eq. (1b) with a general 2-D rotation through counterclockwise angle φ:

$$\begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} cos(\varphi) & -sin(\varphi) \\ sin(\varphi) & cos(\varphi) \end{bmatrix} \cdot \left[\begin{bmatrix} A \cdot cos(t) \\ B \cdot sin(t) \end{bmatrix}\right] \quad 0 \le t \le 2\pi$$
$$\left.\begin{array}{l} x(t) = A \cdot cos(\varphi) \cdot cos(t) - B \cdot sin(\varphi) \cdot sin(t) \\ y(t) = A \cdot sin(\varphi) \cdot cos(t) + B \cdot cos(\varphi) \cdot sin(t) \end{array}\right\} \quad 0 \le t \le 2\pi \tag{2}$$

In the most general case, the rotated ellipse can also be translated away from the origin by an amount (h, k), and the parametric form of a rotated-then-translated ellipse is given by:

$$\left.\begin{array}{l} x(t) = A \cdot cos(\varphi) \cdot cos(t) - B \cdot sin(\varphi) \cdot sin(t) + h \\ y(t) = A \cdot sin(\varphi) \cdot cos(t) + B \cdot cos(\varphi) \cdot sin(t) + k \end{array}\right\} \quad 0 \le t \le 2\pi \tag{3}$$

A rotated-then-translated ellipse can be defined by the set of parameters $\{A, B, h, k, \varphi\}$, with the understanding that the rotation through φ is performed before the translation through (h, k).

Parametric formulation of ellipses is commonly used in models, since the parameters have intuitive geometric analogs. However, finding intersection points between two ellipses is generally accomplished by implicitizing the parametric form. For a general ellipse that may be rotated and translated, the implicit polynomial form is written in the conventional way as:

$$AA \cdot x^2 + BB \cdot x \cdot y + CC \cdot y^2 + DD \cdot x + EE \cdot y + FF = 0 \tag{4a}$$

Equation (4a) can also be written as the matrix equation $\mathrm{X}^{\mathrm{T}} \cdot \mathrm{A} \cdot \mathrm{X} = 0$, where $\mathrm{X}^{\mathrm{T}} = [x, y, 1]$ and A is the symmetric matrix of coefficients $[(a_{i,j})]$:

$$a_{1,1} \cdot x^2 + 2 \cdot a_{1,2} \cdot x \cdot y + a_{2,2} \cdot y^2 + 2 \cdot a_{1,3} \cdot x + 2 \cdot a_{2,3} \cdot y + a_{3,3} = 0 \tag{4b}$$

$$\begin{bmatrix} x & y & 1 \end{bmatrix} \cdot \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{1,2} & a_{2,2} & a_{2,3} \\ a_{1,3} & a_{2,3} & a_{3,3} \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0 \tag{4c}$$

Equivalence of Eqs. (4b) and (4c) is easily verified by multiplying the matrices in Eq. (4c). The matrix form will be convenient for several operations in the overlap area algorithm.

Implicitization from parametric form of Eq. (3) to implicit form of Eq. (4a) is commonly required in applications, and in particular for determining intersection points. Parametric ellipse parameters can be converted algebraically to implicit polynomial coefficients. Suppose the original ellipse is defined by the set of parameters $\{A, B, h, k, \varphi\}$. Any point (x, y) on the original ellipse can be translated by $(-h, -k)$, and then rotated through an angle $-\varphi$:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} cos(-\varphi) & -sin(-\varphi) \\ sin(-\varphi) & cos(-\varphi) \end{bmatrix} \cdot \left\{\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} -h \\ -k \end{bmatrix}\right\}$$
$$= \begin{bmatrix} cos(\varphi) \cdot (x - h) + sin(\varphi) \cdot (y - k) \\ -sin(\varphi) \cdot (x - h) + cos(\varphi) \cdot (y - k) \end{bmatrix}$$

The locus of points (x', y') all satisfy the implicit polynomial form of Eq. (1a), with parameters A and B:

$$\frac{[cos(\varphi) \cdot (x - h) + sin(\varphi) \cdot (y - k)]^2}{A^2} + \frac{[-sin(\varphi) \cdot (x - h) + cos(\varphi) \cdot (y - k)]^2}{}$$

$$AA = a_{1,1} = \frac{\cos^2(\varphi)}{A^2} + \frac{\sin^2(\varphi)}{B^2} \tag{5a}$$

$$BB = 2 \cdot a_{1\cdot 2} = \frac{2 \cdot \sin(\varphi) \cdot \cos(\varphi)}{A^2} - \frac{2 \cdot \sin(\varphi) \cdot \cos(\varphi)}{B^2} \tag{5b}$$

$$CC = a_{2,2} = \frac{\sin^2(\varphi)}{A^2} + \frac{\cos^2(\varphi)}{B^2} \tag{5c}$$

$$DD = 2 \cdot a_{1,3} = \frac{-2 \cdot \cos(\varphi) \cdot [h \cdot \cos(\varphi) + k \cdot \sin(\varphi)]}{A^2} + \frac{2 \cdot \sin(\varphi) \cdot [k \cdot \cos(\varphi) - h \cdot \sin(\varphi)]}{B^2} \tag{5d}$$

$$EE = 2 \cdot a_{2,3} = \frac{-2 \cdot \sin(\varphi) \cdot [h \cdot \cos(\varphi) + k \cdot \sin(\varphi)]}{A^2} + \frac{2 \cdot \cos(\varphi) \cdot [h \cdot \sin(\varphi) - k \cdot \cos(\varphi)]}{B^2} \tag{5e}$$

$$FF = a_{3,3} = \frac{[h \cdot \cos(\varphi) + k \cdot \sin(\varphi)]^2}{A^2} + \frac{[h \cdot \sin(\varphi) - k \cdot \cos(\varphi)]^2}{B^2} - 1 \tag{5f}$$

Note that for an ellipse which is centered at the origin, as in Eq. (2), we have $(h, k) = (0, 0)$, and some implicit polynomial coefficients are simplified, as $DD = 0$, $EE = 0$, and $FF = -1$. Also note that for an ellipse which is both centered and axis aligned, we have $(h, k) = (0, 0)$ and $\varphi = 0$, resulting in $AA = (1/A)^2$, $BB = 0$, $CC = (1/B)^2$, $DD = 0$, $EE = 0$, and $FF = -1$, and Eq. (4a) simplifies exactly to Eq. (1), as expected.

2.2 Ellipse area

The area of a centered and axis-aligned ellipse can be found using the parametric form of Eq. (1b) with the Gauss-Green formula:

$$\text{Area} = \frac{1}{2}\int_{t_1}^{t_2} \left[x(t) \cdot y'(t) - y(t) \cdot x'(t)\right] dt \tag{6}$$

$$\begin{aligned}
&= \frac{1}{2}\int_0^{2\pi} [A \cdot cos(t) \cdot B \cdot cos(t) \\
&\quad -B \cdot sin(t) \cdot (-A) \cdot sin(t)]\, dt \\
&= \frac{1}{2}\int_0^{2\pi} A \cdot B \cdot \left[cos^2(t) + sin^2(t)\right] dt \\
&= \frac{A \cdot B}{2}\int_0^{2\pi} dt \\
&= \pi \cdot A \cdot B
\end{aligned} \tag{7}$$

Rotation of an ellipse is an area-preserving transformation, and it is easily verified that the Gauss-Green integrand $x(t) \cdot y'(t) - y(t) \cdot x'(t)$ for a rotated, centered ellipse simplifies to $A \cdot B$ when using the parametric form in Eq. (2). It is also clear that translation by (h, k) is an area-preserving transformation.

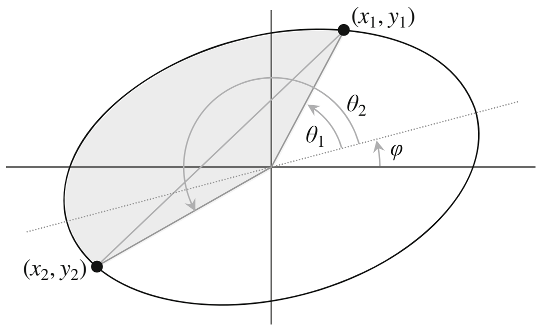


Fig. 1 The area of a sector between two points on an ellipse that is centered at the origin is the area swept out by a vector from the origin to the first point (x_1, y_1) as the vector tip travels along the ellipse in a counter-clockwise direction to the second point (x_2, y_2)

2.3 Ellipse sector area

The *ellipse sector* between two points (x_1, y_1) and (x_2, y_2) on an ellipse is the area that is swept out by a vector that begins at the ellipse center and ends on the ellipse curve, starting the sweep at the first point (x_1, y_1), as the vector end travels along the ellipse in a counter-clockwise direction from the point (x_1, y_1) to the point (x_2, y_2). An example is shown in Fig. 1 for a centered, rotated ellipse.

The Gauss-Green formula (Eq. (6)) can be used to determine the area of a centered ellipse sector by:

$$\text{Sector Area} = \frac{A \cdot B}{2}\int_{\theta_1}^{\theta_2} dt = \frac{(\theta_2 - \theta_1) \cdot A \cdot B}{2} \tag{8}$$

The parametric angle θ corresponding to a point (x, y) on the ellipse is formed between the ellipse axis branch that corresponds to the positive x-axis when $\varphi = 0$, with positive θ in the counter-clockwise direction. Using the principal-valued inverse trigonometric functions that return angles in the in the range $0 \le \theta \le \pi$ for $\theta = \arccos(z)$, and in the range $-\pi/2 \le \theta \le \pi/2$ for $\theta = \arcsin(z)$, ellipse parametric angles can be found with the relations in Table 1.

For calculating a sector area, the Gauss-Green formula is sensitive to the direction of integration. For the larger goal of determining ellipse overlap areas, we follow the convention that the sector area is calculated from the first point (x_1, y_1) to the second point (x_2, y_2) in a counter-clockwise direction. For example, if the points (x_1, y_1) and (x_2, y_2) of Fig. 1 were to have their labels switched, then the ellipse sector defined by the new points will have an area that is complementary to that of the sector in Fig. 1, as shown in Fig. 2.

The definitions in Table 1 will always produce an angle in the range $0 \le \theta < 2\pi$ for any point on the ellipse; as such,

Table 1 Relations for finding the parametric angle corresponding to a given point (x, y) on a centered, rotated ellipse defined with parameters $\{A, B, 0, 0, \varphi\}$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} cos(\varphi) \cdot (x-h) + sin(\varphi) \cdot (y-k) \\ -sin(\varphi) \cdot (x-h) + cos(\varphi) \cdot (y-k) \end{bmatrix}$$

Quadrant II ($x' < 0$ and $y' \geq 0$)	**Quadrant I** ($x' \geq 0$ and $y' \geq 0$)
$\theta = \arccos(x'/A)$	$\theta = \arccos(x'/A)$
$= \pi - \arcsin(y'/B)$	$= \arcsin(y'/B)$
Quadrant III ($x' < 0$ and $y' < 0$)	**Quadrant IV** ($x' \geq 0$ and $y' < 0$)
$\theta = 2\pi - \arccos(x'/A)$	$\theta = 2\pi - \arccos(x'/A)$
$= \pi - \arcsin(y'/B)$	$= 2\pi + \arcsin(y'/B)$

The parametric angle θ is formed between the ellipse axis branch that corresponds to the positive x-axis when $\varphi = 0$ and a line drawn from the origin to the given point, with counterclockwise being positive. Using standard (principal-valued) inverse trigonometric functions, the resulting angle will be in the range $0 \leq \theta < 2\pi$ for any point on the ellipse

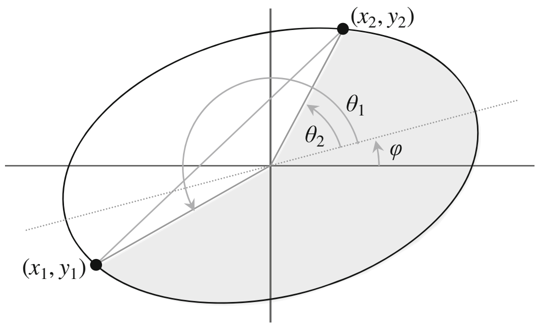


Fig. 2 The ellipse sector area is calculated from the first point (x_1, y_1) to the second point (x_2, y_2) in a counter-clockwise direction

with the point orientations shown in Fig. 2, the corresponding angles will be ordered as $\theta_1 > \theta_2$. The first angle can be transformed by subtracting 2π to restore the condition that $\theta_1 < \theta_2$, and the sector area formula of Eq. (8) can then be used, with the integration angle from $(\theta_1 - 2\pi)$ through θ_2.

2.4 Ellipse segment area

For the overall goal of determining overlap areas between two ellipses, a useful measure is the area of an *ellipse segment*. A secant line drawn between two points on an ellipse partitions the ellipse area into two fractions, as shown in Figs. 1 and 2. An ellipse segment is the area confined by the secant line and the portion of the ellipse from the first point (x_1, y_1) to the second point (x_2, y_2) traversed in a counter-clockwise direction. The segment's complement is the second of the two areas demarcated by the secant line. For the ellipse shown in Fig. 1, the segment area defined by the secant line through the points (x_1, y_1) and (x_2, y_2) is the area of the sector minus the area of a triangle defined by the two points and the ellipse center (at the origin). The coordinates for the triangle's vertices are known, i.e., as (x_1, y_1), (x_2, y_2) and the origin (0, 0), and the triangle area can be found by:

$$\text{Triangle Area} = \frac{1}{2} \cdot |x_1 \cdot y_2 - x_2 \cdot y_1| \tag{9}$$

For the case depicted in Fig. 1, subtracting the triangle area (Eq. (9)) from the area of the ellipse sector (Eq. (8)) gives the area between the secant line and the ellipse, i.e., the area of the ellipse segment counter-clockwise from (x_1, y_1) to (x_2, y_2).

For the ellipse of Fig. 2, the area of the segment shown is the sector area *plus* the area of the triangle. The key difference between the cases in Figs. 1 and 2 that requires the area of the triangle to be either subtracted from, or added to, the sector area is the size of the integration angle. If the integration angle is less than π, then the triangle area must be subtracted from the sector area to give the segment area. If the integration angle is greater than π, the triangle area must be added to the sector area.

$$\text{Segment Area} = \frac{(\theta_2 - \theta_1) \cdot A \cdot B}{2} \pm \frac{1}{2} \cdot |x_1 \cdot y_2 - x_2 \cdot y_1| \tag{10}$$

An algorithm for determining the area of an ellipse segment for a general ellipse is achieved through a generalization of the cases illustrated in Figs. 1, 2 and Eq. (10). Points are passed to the algorithm as (x_1, y_1) and (x_2, y_2), both of which must be on the ellipse. The ELLIPSE_SEGMENT algorithm is outlined below:

1. (x_1, y_1), (x_2, y_2) on ellipse $\{A, B, h, k, \varphi\}$ Inputs to the algorithm
2. $\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} cos(\varphi) \cdot (x_i - h) + sin(\varphi) \cdot (y_i - k) \\ -sin(\varphi) \cdot (x_i - h) + cos(\varphi) \cdot (y_i - k) \end{bmatrix}$, $i = 1, 2$
3. $\theta_1 = \begin{cases} \arccos(x_1'/A), & y_1' \geq 0 \\ 2\pi - \arccos\left(x_1'/A\right), & y_1' < 0 \end{cases}$
   $\theta_2 = \begin{cases} \arccos(x_2'/A), & y_2' \geq 0 \\ 2\pi - \arccos(x_2'/A), & y_2' < 0 \end{cases}$
4. $\hat{\theta}_1 = \begin{cases} \theta_1, & \theta_1 < \theta_2 \\ \theta_1 - 2\pi, & \theta_1 > \theta_2 \end{cases}$
5. $\text{Area} = \frac{\left(\theta_2 - \hat{\theta}_1\right) \cdot A \cdot B}{2} + \frac{sign\left(\theta_2 - \hat{\theta}_1 - \pi\right)}{2} \cdot |x_1 \cdot y_2 - x_2 \cdot y_1|$

where:

the ellipse is defined by the set of parameters $\{A, B, h, k, \varphi\}$, under the convention that rotation through φ is performed before translation by (h, k)
(x_1, y_1) is the first given point, which must be on the ellipse
(x_2, y_2) is the second given point, which must be on the ellipse
θ_1 and θ_2 are the parametric angles corresponding to the points (x_1, y_1) and (x_2, y_2)
Area = segment area proceeding from (x_1, y_1) counterclockwise to (x_2, y_2)

3 Ellipse-ellipse overlap area

We seek a method to determine the overlap area between two general ellipses that are defined parametrically as $(A_1, B_1, h_1, k_1, \varphi_1)$ and $(A_2, B_2, h_2, k_2, \varphi_2)$. The ellipses are first characterized as belonging to one of the ten possible relative positions of two ellipses. Six of the 10 relative positions do not involve transverse intersection points, and overlap area can be computed directly. For the four remaining relative positions, overlap area is calculated by combining the areas of appropriate segments and polygons in each ellipse based on the location of intersection points, which are found numerically.

3.1 Relative position of two ellipses

An algorithm for ellipse overlap area benefits from first determining relative ellipse positions. A robust approach for characterizing the relative positions of two ellipses without knowledge of intersection points is described by [7]. For two ellipses with coefficient matrices $\mathrm{A} = [(a_{i,j})]$ and $\mathrm{B} = [(b_{i,j})]$ as annotated in Eq. (4b), the characteristic polynomial of the pencil $\lambda\mathrm{A} + \mathrm{B}$ is defined as:

$$f(\lambda) = \det(\lambda \cdot \mathrm{A} + \mathrm{B}) = \lambda^3 + a \cdot \lambda^2 + b \cdot \lambda + c \tag{11}$$

To summarize results from [7], it is not necessary to determine roots of the characteristic polynomial. Cases shown in Fig. 3 can be discerned by evaluating inequalities of algebraic combinations of the characteristic polynomial coefficients.

In the context of calculating ellipse area overlap, six of the 10 possible relative positions of two ellipses can be dispensed before attempting to determine points of intersection. Determining relative position reduces the overall calculations required by an overlap area routine that is called many times, as in a simulation. The approach may also avoid sending some ill-conditioned cases to the intersection routine. On the other hand, some ill-conditioned cases may be mis-classified by the relative position algorithm, leading to erroneous overlap area results; difficult cases are discussed in Sect. 3.6.

An algorithm adapted from [7] for determining the relative position of two general ellipses defined implicitly as $(AA_1, BB_1, CC_1, DD_1, EE_1, FF_1)$ and $(AA_2, BB_2, CC_2, DD_2, EE_2, FF_2)$ called ELLIPSE_CASE is outlined below.

1. $\{AA_1, BB_1, CC_1, DD_1, EE_1, FF_1\}, \{AA_2, BB_2, CC_2, DD_2, EE_2, FF_2\}$ Inputs to the algorithm
2. Calculate coefficients of $\det(\lambda \cdot [\mathrm{A}] + [\mathrm{B}]) = \lambda^3 + a \cdot \lambda^2 + b \cdot \lambda + c$

$$d = AA_1 \cdot \left(CC_1 \cdot FF_1 - EE_1^2\right) - \left(CC_1 \cdot DD_1^2 - 2 \cdot BB_1 \cdot DD_1 \cdot EE_1 + FF_1 \cdot BB_1^2\right)$$

$$\begin{aligned} a = \frac{1}{d} \cdot [&AA_1 \cdot (CC_1 \cdot FF_2 - 2 \cdot EE_1 \cdot EE_2 + FF_1 \cdot CC_2) \\ &+2 \cdot BB_1 \cdot (EE_1 \cdot DD_2 - FF_1 \cdot BB_2 + DD_1 \cdot EE_2) \\ &+2 \cdot DD_1 \cdot (EE_1 \cdot BB_2 - CC_1 \cdot DD_2) \\ &- \left(BB_1^2 \cdot FF_2 + DD_1^2 \cdot CC_2 + EE_1^2 \cdot b_{1,1}\right) \\ &+ (CC_1 \cdot FF_1 \cdot AA_2)] \end{aligned}$$

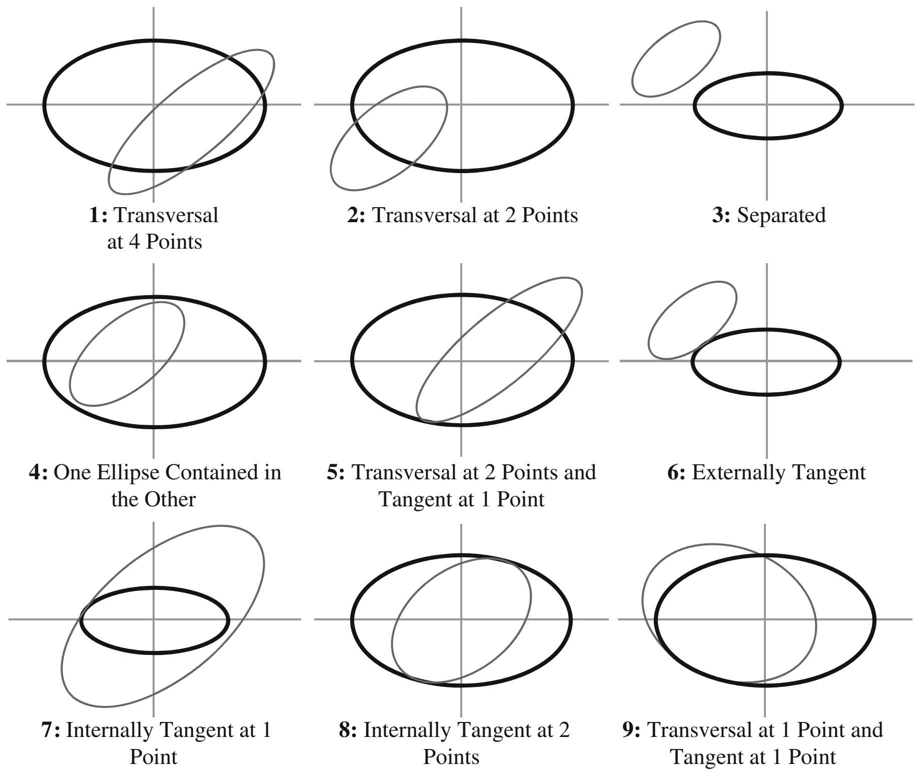
$$\begin{aligned} b = \frac{1}{d} \cdot \Big[&AA_1 \cdot \left(CC_2 \cdot FF_2 - EE_2^2\right) \\ &+2 \cdot BB_1 \cdot (EE_2 \cdot DD_2 - FF_2 \cdot BB_2) \\ &+2 \cdot DD_1 \cdot (EE_2 \cdot BB_2 - CC_2 \cdot DD_2) \\ &+CC_1 \cdot \left(AA_2 \cdot FF_2 - DD_2^2\right) \\ &+2 \cdot EE_1 \cdot (BB_2 \cdot DD_2 - AA_2 \cdot EE_2) \\ &+FF_1 \cdot \left(AA_2 \cdot CC_2 - BB_2^2\right)\Big] \end{aligned}$$

$$\begin{aligned} c = \frac{1}{d} \cdot \Big[&AA_2 \cdot \left(CC_2 \cdot FF_2 - EE_2^2\right) \\ &- \left(BB_2^2 \cdot FF_2 - 2 \cdot BB_2 \cdot DD_2 \cdot EE_2 + DD_2^2 \cdot CC_2\right)\Big] \end{aligned}$$

3. Interpret Coefficients of $\lambda^3 + a \cdot \lambda^2 + b \cdot \lambda + c$

$$\begin{aligned} s_4 = &-27 \cdot c^3 + 18 \cdot c \cdot a \cdot b + a^2 \cdot b^2 \\ &-4 \cdot a^3 \cdot c - 4 \cdot b^3 \end{aligned}$$

```
IF: $(s_4 < 0)$
    THEN: $\alpha < 0; \beta, \bar{\beta} \in \mathbb{C} \rightarrow$ Relative Position 2: RETURN
$s_1 = a, \quad s_2 = a^2 - 3 \cdot b, \quad s_3 = 3 \cdot a \cdot c + b \cdot a^2 - 4 \cdot b^2$
IF: $(s_4 > 0)$
    IF: $(s_1 > 0)$ AND $(s_2 > 0)$ AND $(s_3 > 0)$
        THEN: $\alpha < \beta < \gamma < 0 \rightarrow$ Relative Position 1 or 4
        $u = \frac{-a - \sqrt{s_2}}{3}, \quad v = \frac{-a + \sqrt{s_2}}{3}$
        $[\mathrm{M}] = u \cdot [\mathrm{A}] + [\mathrm{B}], \quad [\mathrm{N}] = v \cdot [\mathrm{A}] + [\mathrm{B}]$
        IF: $\left[\left(m_{2,2} \cdot \det[M] > 0\right)$ AND $(\det[M_{11}] > 0)\right]$
        OR: $\left[\left(n_{2,2} \cdot \det[N] > 0\right)$ AND $(\det[N_{11}] > 0)\right]$
            THEN: Relative Position 4: RETURN
        ELSE: Relative Position 1: RETURN
    ELSE: $\alpha < 0 < \beta < \gamma \rightarrow$ Relative Position 3: RETURN
ELSE:
    IF: $(s_1 > 0)$ AND $(s_2 > 0)$ AND $(s_3 < 0)$
        THEN: $\alpha < 0 < \beta, \beta \rightarrow$ Relative Position 6: RETURN
    ELSE IF: $(s_1 > 0)$ AND $(s_2 > 0)$ AND $(s_3 > 0)$
        THEN: $\alpha < 0;\ \beta, \beta < 0 \rightarrow$ Relative Position 4, 5, 7 or 8
        $\beta = \frac{9 \cdot c - a \cdot b}{2 \cdot s_2}, \alpha = \frac{4 \cdot a \cdot b - a^3 - 9 \cdot c}{s_2}$
        $[\mathrm{M}] = \beta \cdot [\mathrm{A}] + [\mathrm{B}], \quad [\mathrm{N}] = \alpha \cdot [\mathrm{A}] + [\mathrm{B}]$
        IF: $[\det[M_{33}] > 0\ ]$
            THEN IF: $[\ \det[N_{33}] > 0]$
                THEN: Relative Position 5: RETURN
            ELSE: Relative Position 7: RETURN
        ELSE IF: $[\det[M_{11}] + \det[M_{22}] > 0\ ]$
            THEN: Relative Position 4: RETURN
        ELSE: Relative Position 8: RETURN
    ELSE: $\alpha, \alpha, \alpha < 0 \rightarrow$ Relative Position 7,9 or 10
        $\alpha = \frac{-a}{3}$
        $[\mathrm{M}] = \alpha \cdot [\mathrm{A}] + [\mathrm{B}]$
        IF: $[(\det[M] \neq 0)$ AND $(\det[M_{33}] \leq 0)]$
        OR: $\left[(\det[M] \neq 0)\right.$ AND $(\det[M_{33}] \leq 0)$ AND $\left.\left(\frac{\det[M]}{(m_{11} + m_{22})} < 0\right)\right]$
            THEN: Relative Position 10: RETURN
        ELSE IF: $[(\det[M] \neq 0)$ AND $(\det[M_{33}] > 0)]$
            THEN: Relative Position 9: RETURN
        ELSE: Relative Position 7: RETURN
```



1: Transversal at 4 Points
2: Transversal at 2 Points
3: Separated
4: One Ellipse Contained in the Other
5: Transversal at 2 Points and Tangent at 1 Point
6: Externally Tangent
7: Internally Tangent at 1 Point
8: Internally Tangent at 2 Points
9: Transversal at 1 Point and Tangent at 1 Point

Fig. 3 Classification of the relative positions of two ellipses from [7]. A tenth possible configuration is represented by coincident ellipses, i.e., the two ellipses are the same. Overlap area for cases 3, 4, 6, 7, 8 and 10 can be calculated directly from Eq. (7). For the remaining four cases, overlap area is determined by combining the areas of appropriate segments and polygons in each ellipse, based on the position of transverse intersection points

3.2 Determining transverse intersection points I: a direct approach

A direct approach for finding intersection points, discussed in [5], seeks to solve the two implicit ellipse equations simultaneously. The two polynomials, in the form of Eq. (4a), can be re-written as quadratic polynomials in x whose coefficients are functions of y:

$$(AA_1) \cdot x^2 + (BB_1 \cdot y + DD_1) \cdot x + \Big(CC_1 \cdot y^2 + EE_1 \cdot y + FF_1\Big) = 0 \tag{12a}$$

$$(AA_2) \cdot x^2 + (BB_2 \cdot y + DD_2) \cdot x + \Big(CC_2 \cdot y^2 + EE_2 \cdot y + FF_2\Big) = 0 \tag{12b}$$

Any points of intersection (x, y) for the two ellipses will satisfy Eqs. (12a) and (12b) simultaneously. For convenience, the coefficients can be substituted:

$$\begin{aligned} &u_2 \cdot x^2 + u_1 \cdot x + u_0 = 0 \\ &\quad u_2 = (AA_1), u_1 = (BB_1 \cdot y + DD_1), \\ &\quad u_0 = \Big(CC_1 \cdot y^2 + EE_1 \cdot y + FF_1\Big) \end{aligned} \tag{13a}$$

$$\begin{aligned} &v_2 \cdot x^2 + v_1 \cdot x + v_0 = 0 \\ &\quad v_2 = (AA_2), v_1 = (BB_2 \cdot y + DD_2), \\ &\quad v_0 = \Big(CC_2 \cdot y^2 + EE_2 \cdot y + FF_2\Big) \end{aligned} \tag{13b}$$

Using a standard approach (developed in the nineteenth century), the two polynomials of Eq. (13) have a common root whenever their Bézout determinant is zero:

$$\begin{aligned} &(u_1 \cdot v_0 - u_0 \cdot v_1) \cdot (u_2 \cdot v_1 - u_1 \cdot v_2) \\ &\quad - (u_2 \cdot v_0 - u_0 \cdot v_2)^2 = 0 \end{aligned} \tag{14}$$

Using the definitions from Eq. (13) and substituting into Eq. (14) produces a quartic polynomial in y with the following coefficients:

$$\begin{aligned} &cy[4] \cdot y^4 + cy[3] \cdot y^3 + cy[2] \cdot y^2 \\ &\qquad + cy[1] \cdot y^1 + cy[0] = 0 \end{aligned} \tag{15}$$

where:

$$\begin{aligned}
cy[4] = {} & (BB_1 \cdot CC_2 - CC_1 \cdot BB_2) \cdot (AA_1 \cdot BB_2 - BB_1 \cdot AA_2) \\
& - (AA_1 \cdot CC_2 - CC_1 \cdot AA_2)^2 \\
cy[3] = {} & 2 \cdot (EE_1 \cdot AA_2 - AA_1 \cdot EE_2) \cdot (AA_1 \cdot CC_2 - CC_1 \cdot AA_2) \\
& + (DD_1 \cdot CC_2 + BB_1 \cdot EE_2 - EE_1 \cdot BB_2 - CC_1 \cdot DD_2) \\
& \cdot (AA_1 \cdot BB_2 - BB_1 \cdot AA_2) + (BB_1 \cdot CC_2 - CC_1 \cdot BB_2) \\
& \cdot (AA_1 \cdot DD_2 - DD_1 \cdot AA_2) \\
cy[2] = {} & 2 \cdot (FF_1 \cdot AA_2 - AA_1 \cdot FF_2) \cdot (AA_1 \cdot CC_2 - CC_1 \cdot AA_2) \\
& + (DD_1 \cdot EE_2 + BB_1 \cdot FF_2 - FF_1 \cdot BB_2 \\
& - EE_1 \cdot DD_2) \cdot (AA_1 \cdot BB_2 - BB_1 \cdot AA_2) \\
& + (DD_1 \cdot CC_2 + BB_1 \cdot EE_2 - EE_1 \cdot BB_2 - CC_1 \cdot DD_2) \\
& \cdot (AA_1 \cdot DD_2 - DD_1 \cdot AA_2) - (AA_1 \cdot EE_2 - EE_1 \cdot AA_2)^2 \\
cy[1] = {} & 2 \cdot (FF_1 \cdot AA_2 - AA_1 \cdot FF_2) \cdot (AA_1 \cdot EE_2 - EE_1 \cdot AA_2) \\
& + (DD_1 \cdot EE_2 + BB_1 \cdot FF_2 - FF_1 \cdot BB_2 - EE_1 \cdot DD_2) \\
& \cdot (AA_1 \cdot DD_2 - DD_1 \cdot AA_2) \\
& + (DD_1 \cdot FF_2 - FF_1 \cdot DD_2) \cdot (AA_1 \cdot BB_2 - BB_1 \cdot AA_2)
\end{aligned}$$

$$cy[0] = (DD_1 \cdot FF_2 - FF_1 \cdot DD_2) \cdot (AA_1 \cdot DD_2 - DD_1 \cdot AA_2) - (AA_1 \cdot FF_2 - FF_1 \cdot AA_2)^2$$

When roots of Eq. (15) are found, y-values where the polynomials of Eq. (12) intersect can be determined by:

$$\bar{y} = \frac{AA_1 \cdot (CC_2 \cdot \bar{y}^2 + EE_2 \cdot \bar{y} + FF_2) - AA_2 \cdot (CC_1 \cdot \bar{y}^2 + EE_1 \cdot \bar{y} + FF_1)}{AA_2 \cdot (BB_1 \cdot \bar{y} + DD_1) - AA_1 \cdot (BB_2 \cdot \bar{y} + DD_2)} \tag{16}$$

To find points of intersection, substitute each ȳ back into Eq. (12a) (or Eq. (12b)), and solve for two x̄ values, using the quadratic formula:

$$\bar{x} = \frac{-(BB_1 \cdot \bar{y} + DD_1) \pm AA_2 \cdot \sqrt{(BB_1 \cdot \bar{y} + DD_1)^2 - 4 \cdot AA_1 \cdot \left(CC_1 \cdot \bar{y}^2 + EE_1 \cdot \bar{y} + FF_1\right)}}{2 \cdot AA_1} \tag{17}$$

Pairs (x̄, ȳ) that satisfy both Eqs. (12a) and (12b) represent points of intersection of the two original ellipses.

Many options exist for numeric solution of the quartic polynomial in Eq. (15); see, e.g, [16] or [6] for a survey of modern methods, and [13] for a comprehensive list of root-finding references. However, polynomial root-finding remains a challenging problem, and until recently the accuracy of most numerical implementations has been limited, particularly when multiple roots are present or when inexact coefficients are supplied. Several algorithms have been introduced recently that provide improved accuracy. One such algorithm is described in [15], based on recursive numerical splitting of the input polynomial into the product of factors. Another routine with improved accuracy is `gsl_poly_complex_solve` from the GNU scientific library (GSL), which is based on [19]. A custom GSL extension specifically for quartic polynomials is also available [18]. The algorithm in [19] first calculates the multiplicity structure for the roots, and computes an initial approximation to the roots; the initial approximation is then refined to an optimal accuracy.

In the context of determining ellipse area overlap for a simulation, some benefit may be obtained from a non-iterative solution of the quartic equation, even if results are less accurate than iterative solvers can provide. An expedient (non-iterative) method for determining roots of the polynomial in Eq. (15) can be accomplished using Ferrari's quartic formula. A numerical implementation of Ferrari's formula is given in [14]. Four complex roots are returned, and any roots whose imaginary part is returned as zero is a real root. When polynomial coefficients are constructed as in Eq. (15), the general case of two distinct ellipses typically results in a quartic polynomial, i.e., the coefficient $cy[4]$ of Eq. (15) is non-zero. However, certain cases lead to polynomials of lesser degree. The solver in [14] is conveniently modular, providing separate functions BIQUADROOTS, CUBICROOTS and QUADROOTS to handle all the possible polynomial cases that arise when seeking points of intersection for two ellipses.

3.3 Determining transverse intersection points II: tools from algebraic geometry

To identify intersection points, the direct approach of solving the two implicit ellipse equations simultaneously is easily implemented. However, even with accurate root-finding algorithms, the direct approach encounters difficulties with some ill-conditioned cases. The root-finding algorithms in [15] and [19] are robust in general situations; even so, both algorithms are still susceptible to error propagation from inexact coefficients, particularly when there are multiple roots or roots that are clustered very close together. The direct approach calculates polynomial coefficients from Eq. (15), and will send difficult polynomials to the root-finding routine in cases where the curves intersect at a shallow angle. Robustness of the area overlap algorithm can be improved using methods from algebraic geometry that have been designed specifically for identifying intersection points of implicitly defined curves.

Several independent approaches are available for finding intersection points of two implicit curves. [12] describe a method for computing geometrically isolated higher order intersections of curves. A similar method is described in [3]. Both of these methods are based on solving an Eigen-problem that is related to companion matrices of the two implicit ellipse curves. In the context of finding intersection points, one advantage of the companion matrix approach is that accuracy of numerical approaches for Eigen-problems is well-studied, and error bounds for intersection points can be established.

3.4 Overlap area determination

In six of the 10 possible relative positions of two ellipses shown in Fig. 3, ellipse overlap area can be calculated directly. In relative positions 3 and 6, the ellipses are disjoint and the overlap area is 0. In relative position 10 (coincident ellipses), the overlap area is $\pi \cdot A \cdot B$. For relative positions 4, 7 and 8, the overlap area is the area of the contained ellipse, which must be the smaller of the two ellipse areas.

Relative positions 2 and 9 each have two intersection points. Overlap area is found by adding the area of two ellipse segments, one from each ellipse, as shown in Fig. 4. The two intersection points (x_1, y_1) and (x_2, y_2) must be passed to

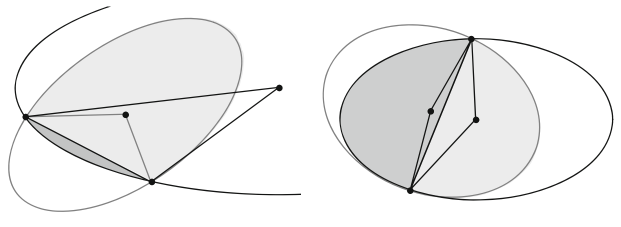


Fig. 4 Relative Positions 2 and 9 have two transverse intersection points. Overlap area is found by adding the area of two ellipse segments, one from each ellipse

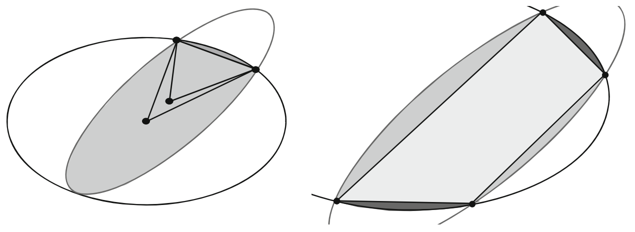


Fig. 5 Relative position 5 (*left*) has three intersection points, but only two are transverse. Overlap area is found by determining which two points are transverse, then adding the area of two ellipse segments, one from each ellipse. Relative position 1 (*right*) has four transverse intersection points. Overlap area is found by adding the area of four ellipse segments, two from each ellipse, and one quadrilateral

the segment area algorithm in the order that will return the correct segment from each ellipse. Only one area from each ellipse contributes to the overlap area. A check is made to determine which point order will return the desired segment area for each ellipse. First, the parametric angles θ_1 and θ_2 corresponding to (x_1, y_1) and (x_2, y_2) on the first ellipse are determined, by the rules in Table 1. Then, a point midway between (x_1, y_1) and (x_2, y_2) on the first ellipse is found using the parametric form:

$$x_{mid} = A \cdot cos(\varphi_1) \cdot cos\left(\frac{\theta_1 + \theta_2}{2}\right) - B \cdot sin(\varphi_1) \cdot sin\left(\frac{\theta_1 + \theta_2}{2}\right) + h_1 \tag{18a}$$

$$y_{mid} = A \cdot sin(\varphi_1) \cdot cos\left(\frac{\theta_1 + \theta_2}{2}\right) + B \cdot cos(\varphi_1) \cdot sin\left(\frac{\theta_1 + \theta_2}{2}\right) + k_1 \tag{18b}$$

The point $(x_{\text{mid}}, y_{\text{mid}})$ is on the first ellipse between (x_1, y_1) and (x_2, y_2) when travelling counter- clockwise from (x_1, y_1) and (x_2, y_2). If $(x_{\text{mid}}, y_{\text{mid}})$ is inside the second ellipse, then the desired segment of the first ellipse contains the point $(x_{\text{mid}}, y_{\text{mid}})$, and the segment algorithm should integrate counterclockwise from (x_1, y_1) to (x_2, y_2), which is the default order. Otherwise, the order of the points should be reversed before calling the segment algorithm. The implicit polynomial form can be used to determine whether $(x_{\text{mid}}, y_{\text{mid}})$ is inside or outside the second ellipse:

$$T = AA_2 \cdot x_{mid}^2 + BB_2 \cdot x_{mid} \cdot y_{mid} + CC_2 \cdot y_{mid}^2 + DD_2 \cdot x_{mid} + EE_2 \cdot y_{mid} + FF_2 \tag{19}$$

If $T < 0$ then the point $(x_{\text{mid}}, y_{\text{mid}})$ is inside the second ellipse. The desired segment area from the second ellipse is found in a manner similar to the first ellipse.

Relative position 5 has three intersection points, but only two are transverse. Overlap area can be found by the same method as relative positions 2 and 9, if the two transverse intersection points are used, as shown in Fig. 5. For ellipses in relative position 5, the transverse intersection points correspond to roots of multiplicity 1. The tangential intersection point corresponds to a root of multiplicity 2. For ellipses in relative position 9, there are two transverse intersection points; one of the points corresponds to a root of multiplicity 1, and the other corresponds to a root of multiplicity 3.

Relative position 1 consists four transverse intersection points, shown in Fig. 5. The two ellipse curves must cross at all four of the intersection points, resulting in a partial overlap. The overlap area consists of two segments from each ellipse, and a central convex quadrilateral. The four intersection points are sorted ascending in a counter-clockwise order around the first ellipse. The ordered set of intersection points is (x_1, y_1), (x_2, y_2), (x_3, y_3) and (x_4, y_4). The ordering allows a direct calculation of the quadrilateral area. The standard formula uses the cross-product of the two diagonals:

$$\begin{aligned} \text{area} &= \frac{1}{2}\left|(\mathrm{x}_3 - \mathrm{x}_1, \mathrm{y}_3 - \mathrm{y}_1) \times (\mathrm{x}_4 - \mathrm{x}_2, \mathrm{y}_4 - \mathrm{y}_2)\right| \\ &= \frac{1}{2}\left|(\mathrm{x}_3 - \mathrm{x}_1) \cdot (\mathrm{y}_4 - \mathrm{y}_2) - (\mathrm{x}_4 - \mathrm{x}_2) \cdot (\mathrm{x}_3 - \mathrm{x}_1)\right| \end{aligned} \tag{20}$$

The point ordering also simplifies the search for the appropriate segments of each ellipse that contribute to the overlap area. Suppose that the first two sorted points (x_1, y_1) and (x_2, y_2) demarcate a segment of the first ellipse that contributes to the overlap area. It follows that contributing segments from the first ellipse are between the points (x_1, y_1) and (x_2, y_2), and also between the points (x_3, y_3) and (x_4, y_4). In this case, the contributing segments from the second ellipse are between the points (x_2, y_2) and (x_3, y_3), and between the points (x_4, y_4) and (x_1, y_1). To determine which segments contribute to the overlap area, it suffices to test whether a point midway between the points (x_1, y_1) and (x_2, y_2) is inside or outside the second ellipse. The segment algorithm is used for each of the four areas, and added to the quadrilateral to obtain the total overlap area.

3.5 General overlap area algorithm

Combining the results presented above leads to an algorithm for determining the overlap area between two general ellipses defined parametrically as $(A_1, B_1, h_1, k_1, \varphi_1)$ and $(A_2, B_2, h_2, k_2, \varphi_2)$. The algorithm ELLIPSE_OVERLAP is outlined below. Condition of the implicitization operation can be improved for some difficult cases by translating both ellipses to put the midpoint of the two ellipse centers at the origin. Translation does not affect the relative position of the two original ellipses.

```
1. $\{A_1, B_1, \bar{h}_1, \bar{k}_1, \varphi_1\}, \{A_2, B_2, \bar{h}_2, \bar{k}_2, \varphi_2\}$ Inputs to the algorithm
2. $\begin{bmatrix} \bar{h}_i \\ \bar{k}_i \end{bmatrix} = \begin{bmatrix} h_i - \frac{h_1+h_2}{2} \\ k_i - \frac{k_1+k_2}{2} \end{bmatrix}, i = 1, 2$
3. $\{A_1, B_1, \bar{h}_1, \bar{k}_1, \varphi_1\} \to \{AA_1, BB_1, CC_1, DD_1, EE_1, FF_1\}$
   $\{A_2, B_2, \bar{h}_2, \bar{k}_2, \varphi_2\} \to \{AA_2, BB_2, CC_2, DD_2, EE_2, FF_2\}$
   by Eq. (5)
4. Determine relative position of the two implicitly defined ellipses by ELLIPSE_CASE algorithm adapted from [7], as given in Sect. 3.1
5. RELATIVE POSITION:
      CASE10: one ellipse contained in the other
        AREA $= \pi \cdot A_1 \cdot B_1 = \pi \cdot A_2 \cdot B_2$ : RETURN
      CASE 3, 6: no overlap area
          AREA=0: RETURN
      CASE 4, 7, 8: one ellipse contained in the other
        AREA $= \min\{\pi \cdot A_1 \cdot B_1, \pi \cdot A_2 \cdot B_2\}$ : RETURN
      CASE 1, 2, 5, 9: transverse intersections
          Calculate coefficients $cy[4]$, $cy[3]$, $cy[2]$, $cy[1]$, $cy[0]$ by Eq. (15)
          Solve quartic $cy[4] \cdot y^4 + cy[3] \cdot y^3 + cy[2] \cdot y^2 + cy[1] \cdot y^1 + cy[0] = 0$
          Calculate intersection points $(\bar{x}, \bar{y})$ by Eqs. (16) and (17)

          CASE 2, 5, 9: two transverse intersections
              Calculate segment area contributed by each ellipse using ELLIPSE_SEGMENT algorithm, as given in Sect. 2.4
              AREA = Segment 1 + Segment 2:RETURN
          CASE 1: four transverse intersections
              Calculate four segment areas contributed by each ellipse using ELLIPSE_SEGMENT algorithm, as given in Sect. 2.4
              Calculate quadrilateral area contributed by each ellipse using Eq. (20)
              AREA = Segment 1 + Segment 2 + Segment 3 + Segment 4+Quadrilateral :RETURN
```

3.6 Notes on accuracy, precision, and robustness

The algorithms presented in this paper describe an approach for determining the overlap area between two general ellipses. Any numerical implementation of the algorithm is susceptible to propagation of fixed-point round-off errors. Round-off errors that arise during implicitization of the ellipse parameters using Eq. (5) can produce inexact polynomial coefficients, leading to inaccurate roots. Inaccuracy the polynomial roots produces perturbed intersection points, and the error propagates to the determination of polygon and ellipse sector areas. Closed-form determination of overlap areas is not available for all but the simplest cases, so absolute accuracy of the algorithm is not obtainable. For ellipses that are commonly encountered in applications, errors in the overlap area algorithm are directly tied to root-finding, for which error bounds are available [15]. We have encountered some 'odd' cases, such as extremely eccentric ellipses, where the algorithm returns the wrong polynomial case; in these situations, the area returned by the algorithm is unrelated to the actual overlap area.

A relevant question is whether the algorithm as presented here will handle cases where one or both ellipses are circles. In such cases, inputs to the algorithm could include one or two ellipses defined using the parametric definition for an ellipse as $(A, A, h, k, 0)$, representing a circle of radius A centered at (h, k). The implicitization of Eq. (5) is still valid, and simplifies to:

$$AA = a_{1,1} = \frac{1}{A^2} \tag{21a}$$

$$BB = 2 \cdot a_{1,2} = 0 \tag{21b}$$

$$CC = a_{2,2} = \frac{1}{A^2} \tag{21c}$$

$$DD = 2 \cdot a_{1,3} = \frac{-2 \cdot h}{A^2} \tag{21d}$$

$$EE = 2 \cdot a_{2,3} = \frac{-2 \cdot k}{A^2} \tag{21e}$$

$$FF = a_{3,3} = \frac{h^2}{A^2} + \frac{k^2}{A^2} - 1 \tag{21f}$$

Following the algorithm by inputting circles as ellipses with the appropriate features, the polynomial coefficients of Eq. (15) are calculated, as shown in Eq. (21), and the algorithm returns the correct overlap area where one or both curves are circles. The eccentricity of a circle is zero, so circles do not result in eccentricity-related issues for the algorithm. Circles that are very close in size and position can still lead to difficult root-finding cases.

An implementation in C-code of the ELLIPSE_OVERLAP algorithm and all of its dependencies was compiled and run under Cygwin-1.7.7-1 and Debian 7.2 (Wheezy), and returns the following values for examples of the test cases presented in Fig. 3:

```
//-- CASE 1
A1 = 3.; B1 = 2.; H1 = 0.; K1 = 0.; PHI_1 = 0.;
A2 = 3.; B2 = 1.; H2 = 1.; K2 = -0.5; PHI_2 = pi/4.;
//-- CASE 2
A1 = 3.; B1 = 2.; H1 = 0.; K1 = 0.; PHI_1 = 0.;
A2 = 2.; B2 = 1.; H2 = -2.; K2 = -1.; PHI_2 = pi/4.;
//-- CASE 3
A1 = 2.; B1 = 1.; H1 = 0.; K1 = 0.; PHI_1 = 0.;
A2 = 1.5; B2 = 0.75; H2 = -2.5; K2 = 1.5; PHI_2 = pi/4.;
//-- CASE 4
A1 = 3.; B1 = 2.; H1 = 0.; K1 = 0.; PHI_1 = 0.;
A2 = 2.; B2 = 1.; H2 = -.75; K2 = 0.25; PHI_2 = pi/4.;
//-- CASE 5
A1 = 3.; B1 = 2.; H1 = 0.; K1 = 0.; PHI_1 = 0.;
A2 = 3.; B2 = 3.; H2 = 0.; K2 = 1.; PHI_2 = 0.;
//-- CASE 6
A1 = 2.; B1 = 1.; H1 = 0.; K1 = 0.; PHI_1 = 0.;
A2 = 2.; B2 = 1.; H2 = 0.; K2 = 2.; PHI_2 = 0.;
//-- CASE 7
A1 = 3.; B1 = 2.; H1 = 0.; K1 = 0.; PHI_1 = 0.;
A2 = 2.; B2 = 1.; H2 = -1.0245209260022; K2 = 0.25; PHI_2 = pi/4.;
//-- CASE 8
A1 = 3.; B1 = 2.; H1 = 0.; K1 = 0.; PHI_1 = 0.;
A2 = 1.; B2 = 2.; H2 = 0.; K2 = 0.; PHI_2 = 0.;
//-- CASE 9
AA1 = 9.; BB1 = 0.; CC1 = 100; DD1 = 0.; EE1 = 0.; FF1 = -81.;
AA2 = 143.; BB2 = 0.; CC2 = 773; DD2 = -267.; EE2 = -155.; FF2 = -509.;
//-- CASE 10
A1 = 3.; B1 = 2.; H1 = 0.; K1 = 0.; PHI_1 = 0.;
A2 = 3.; B2 = 2.; H2 = 0.; K2 = 0.; PHI_2 = 0.;
```

```
$ cc call_ee.c ellipse_overlap.c -o call_ee
$ ./call_ee
Calling ellipse_overlap.c
CASE 1: area =     16.93791852
CASE 2: area =      3.82254574
CASE 3: area =      0.00000000
CASE 4: area =      6.28318531
CASE 5: area =     17.60218852
CASE 6: area =      0.00000000
CASE 7: area =      6.28318531
CASE 8: area =      6.28318531
CASE 9: area =      3.42554080
CASE 10: area =      6.28318531
```

The accuracy of the algorithm, as implemented in C and C++, was tested extensively. We compare the results of 1,000 selected pairs of ellipses with the overlap areas and calculation times of the corresponding n-sided polygons. Figure 6 shows that implementation of the analytical solution on an Intel Core i7-2620M, 2,7 GHz, 4MB Cache, is 40–140 times faster than the solution based a polygon-approximation of ellipses. For the results presented in Fig. 6, we make use of the C++-library Boost-polygon [2]. Figure 7 shows a quantitative visualization of the calculated crossing points of selected ellipses. The code is open source and can be downloaded from https://github.com/chraibi/EEOver.

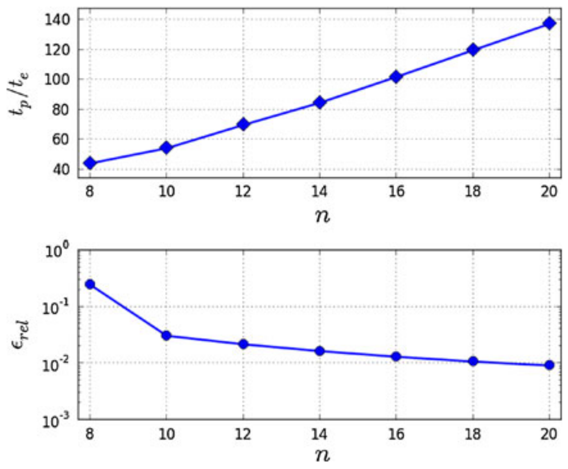


Fig. 6 *Top*: The ratio of the run-time of the polygon-based method (t_p) and the analytical solution presented in this paper as implemented in C++ (t_e) with respect to the number of edges. *Bottom*: The relative error of the calculated overlap areas with respect to the number of edges of a corresponding n-sided polygons used to approximate the area

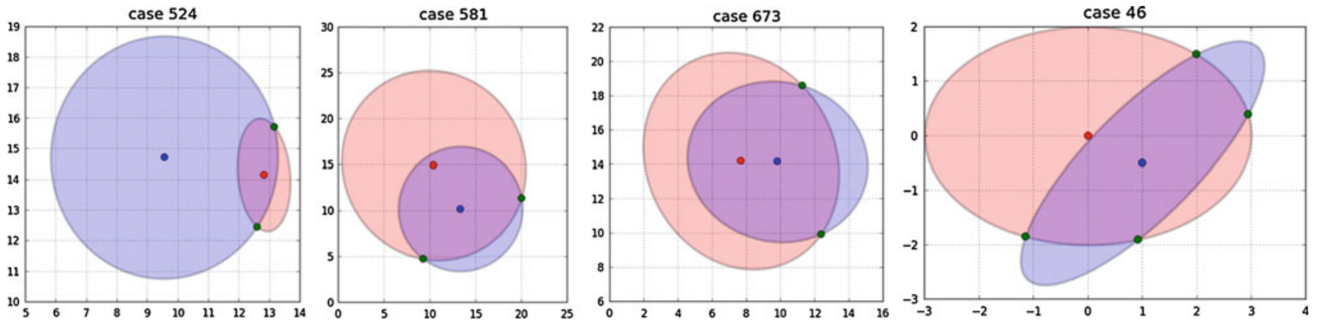


Fig. 7 Results of selected test cases. The *red* and *blue* points represent ellipse centers, and *green* points represent intersection points, calculated with the algorithm as implemented in C++

References

1. Böröczky, K., Reitzner, M.: Approximation of smooth convex bodies by random circumscribed polytopes. Ann. Appl. Prob. **14**(1), 239–273 (2004)
2. Boost The Boost. Polygon Library http://www.boost.org/doc/libs/1_54_0/libs/polygon/doc/index.htm
3. Busé, L., Khalil, H., Mourrain, B.: Resultant-based methods for plane curves intersection problems. Lecture notes in computer science, pp. 75–92 (2005)
4. Chraibi, M., Seyfried, A., Schadschneider, A.: Generalized centrifugal force model for pedestrian dynamics. Phys. Rev. E. **82**(4), 046111 (2010)
5. Eberly, D.: The area of intersecting ellipses. Accessed at http://www.geometrictools.com/Documentation/AreaIntersectingEllipses.pdf (2010)
6. Elkadi, M., Mourrain, B.: Symbolic-numeric tools for solving polynomial equations and applications. In: Dickenstein, A., Emiris, I. (eds.) Solving Polynomial Equations: Foundations, Algorithms, and Applications, vol. 14 of Algorithms and Computation in Mathematics, pp. 125–168. Springer, Berlin (2005)
7. Etayo, F., Gonzalez-Vega, L., Del Rio, N.: A new approach to characterizing the relative position of two ellipses depending on one parameter. Comput. Aided Geom. Des. **23**(4), 324–350 (2006)
8. Fu, P., Walton, O.R., Harvey, J.T.: Polyarc discrete element for efficiently simulating arbitrarily shaped 2D particles. Int. J. Numer. Methods Eng. **89**(5), 599–617 (2012)
9. Gruber, P.M.: Asymptotic estimates for best and stepwise approximation of convex bodies IV. Forum Math. **10**, 665–686 (1998)
10. Han, K., Feng, Y.T., Owen, D.R.J.: Polygon-based contact resolution for superquadrics. Int. J. Numer. Methods Eng. **66**, 485–501 (2006)
11. Ludwig, M.: Asymptotic approximation of convex curves. Arch. Math. **63**, 377–384 (1994)
12. Manocha, D., Demmel, J.: Algorithms for intersecting parametric and algebraic curves I: simple intersections. ACM Transactions on Graphics (TOG) **13**(1), 73–100 (1994)
13. Mcnamee, J.M.: A 2002 update of the supplementary bibliography on roots of polynomials. J. Comput. Appl. Math. **142**(2), 433–434 (2002)
14. Nonweiler, T.R.F.: CACM Algorithm 326: Roots of low order polynomials. Communications of the ACM. 11, 4, 269–270. Translated into C and programmed by M. Dow, ANUSF, Australian National University, Canberra, Australia. Accessed at http://www.netlib.org/toms/326 (1968)
15. Pan, V.Y.: Univariate polynomials: nearly optimal algorithms for numerical factorization and root-finding. J. Symb. Comput. **00**, 1–33 (2002)
16. Pan, V.Y.: Solving a polynomial equation: some history and recent progress. SIAM Rev. **39**(2), 187–220 (1997)
17. Press, W.H., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T.: Numerical Recipes: the Art of Scientific Computing. Cambridge University Press, New York (1992)
18. Steiner, A., Buckley, A.: GSL Extension for solving quartic polynomials with gsl_poly_complex_solve. Accessed at http://www.network-theory.co.uk/download/gslextras/Quartic/ (2004)
19. Zeng, Z.: Algorithm 835. ACM Trans. Math. Softw. **30**(2), 218–236 (2004)