

- 曼陀罗花的数学几何及算法
 - 自然界中的曼陀罗花
 - 我研究的曼陀罗花
 - 曼陀罗花简介
 - 曼陀罗花的起源
 - 曼陀罗花花语
 - 我研究曼陀罗花的起源
 - 曼陀罗花的数学
 - 曼陀罗花的画图算法
 - 结束语

#! <https://zhuanlan.zhihu.com/p/698760383>

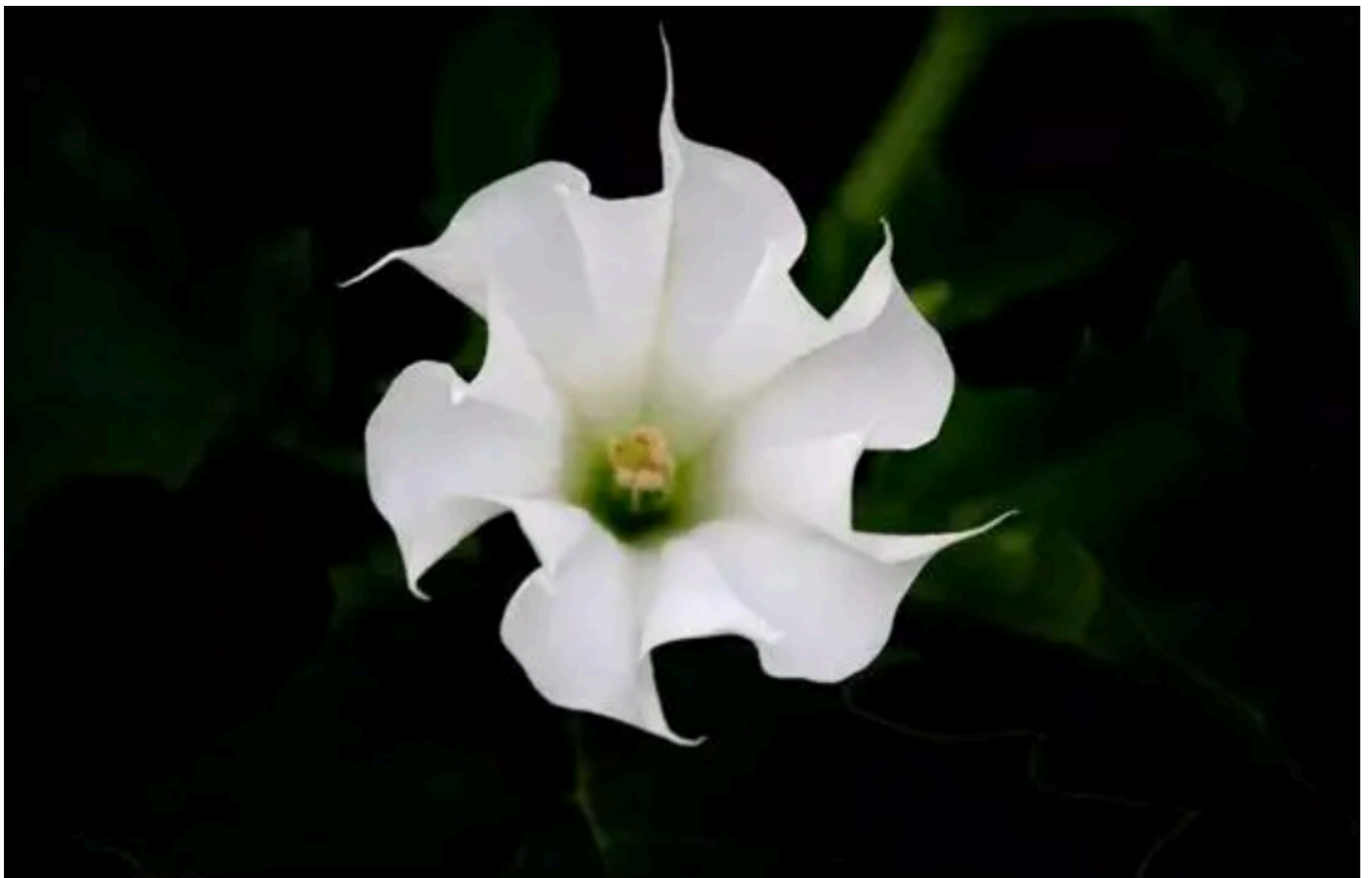
曼陀罗花的数学几何及算法

先上图，欣赏和看看曼陀罗花长什么样：

自然界中的曼陀罗花



六瓣白色曼陀罗花，按数学几何它应该是最基本最自然的一种曼陀罗花



五瓣曼陀罗花，它的数学应该与黄金比例联系吧



金色曼陀罗花，据说代表幸福



多层曼陀罗花，我一直以为自然界中只有单层曼陀罗花，多层只是我程序画出来的设

想，没想到存在即合理，但凡能想到的宇宙中就存在





懒得一张张找图贴了，干脆把网页截图给大家思路去搜索，图片来自百度百科



盛开的曼陀罗花_生...



曼陀罗花



【曼陀罗花】-动物...



【曼陀罗花摄影图片】生态摄影...



拥有曼陀罗花的梦幻



曼陀罗花



曼陀罗花_叶碧花艳,凄美而诡异...



【引用】神秘的曼陀罗花...



曼陀罗花



曼陀罗花_生态_poco...



曼陀罗花_生态_poco摄影_生态...



曼陀罗花_正版商业...

相关搜索

- 国画曼陀罗花
- 彼岸花的寓意
- 曼陀罗艺术



关东百花图(185)——金色曼陀...



曼陀罗花记 - 简书



【曼陀罗花摄影图片】生态摄影...



树上的白曼陀罗花高...



【曼陀罗花摄影图片】生态摄影...



【曼陀罗花摄影图片】生态摄影...



心灵摄影【美摄视界...



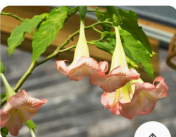
野曼陀罗花图片大全...



曼陀罗花的功效与食用禁忌 曼...



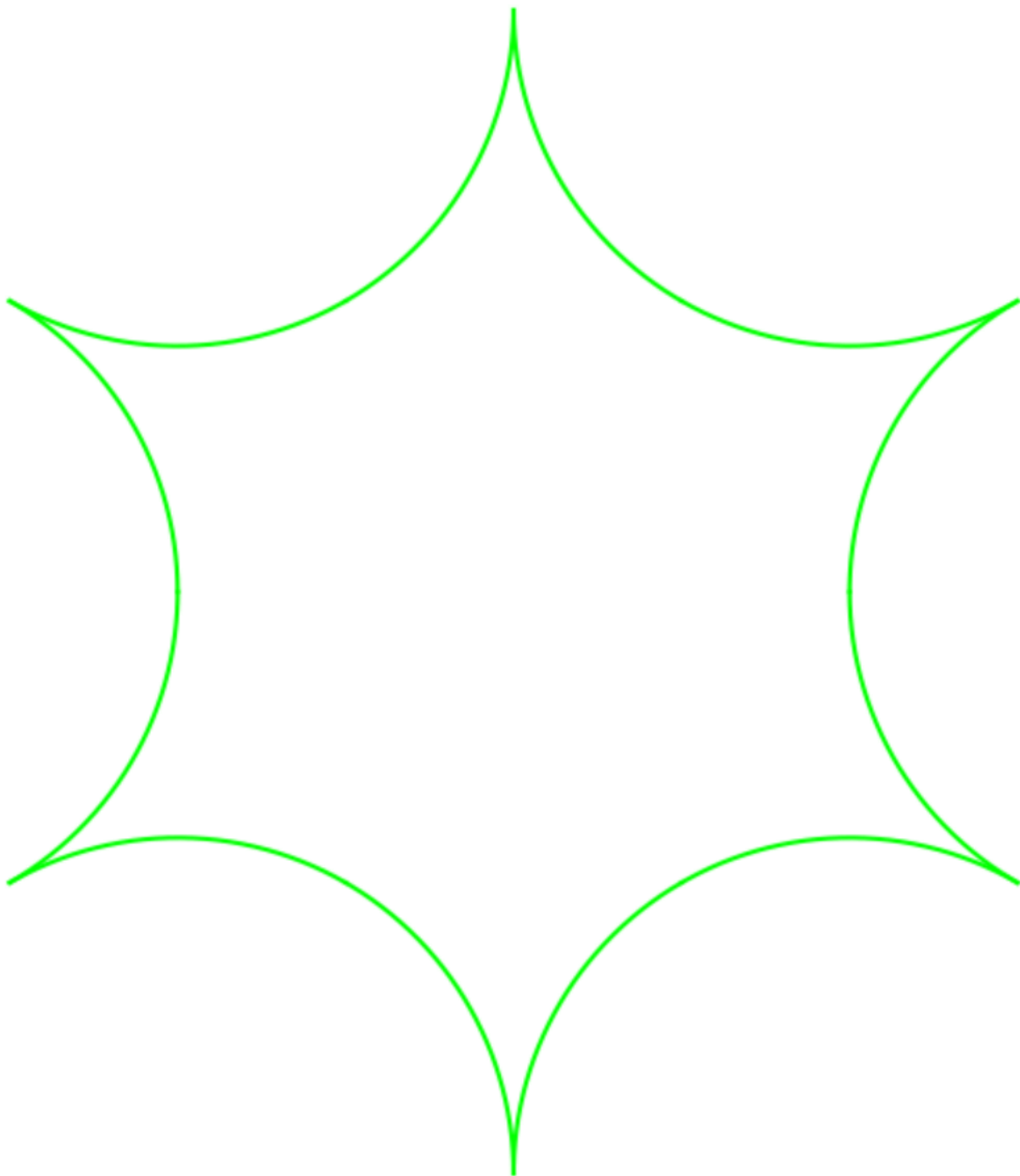
曼陀罗花图片_图片大全



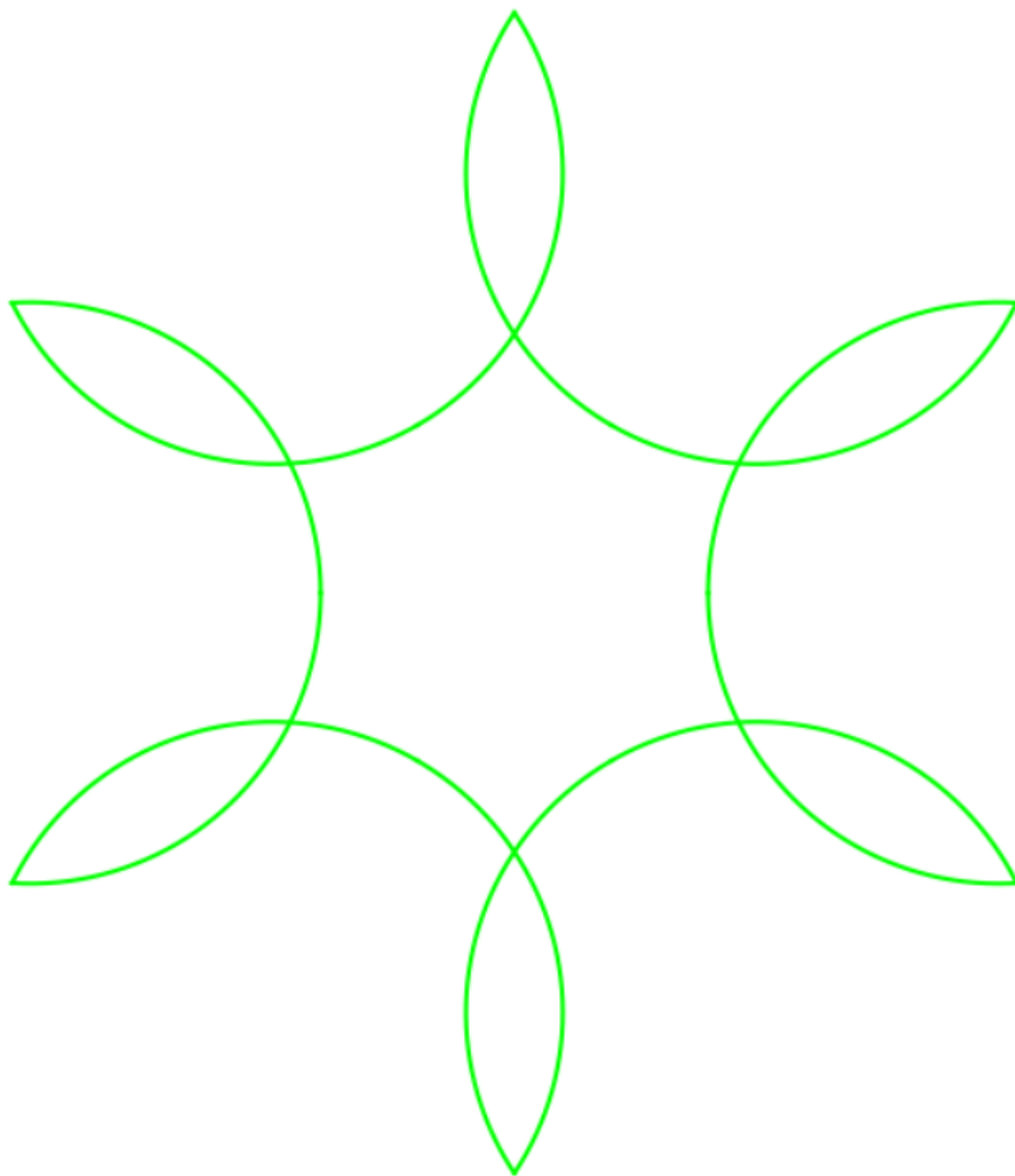
有市的曼陀罗花 - 第2页 - 上

我研究的曼陀罗花

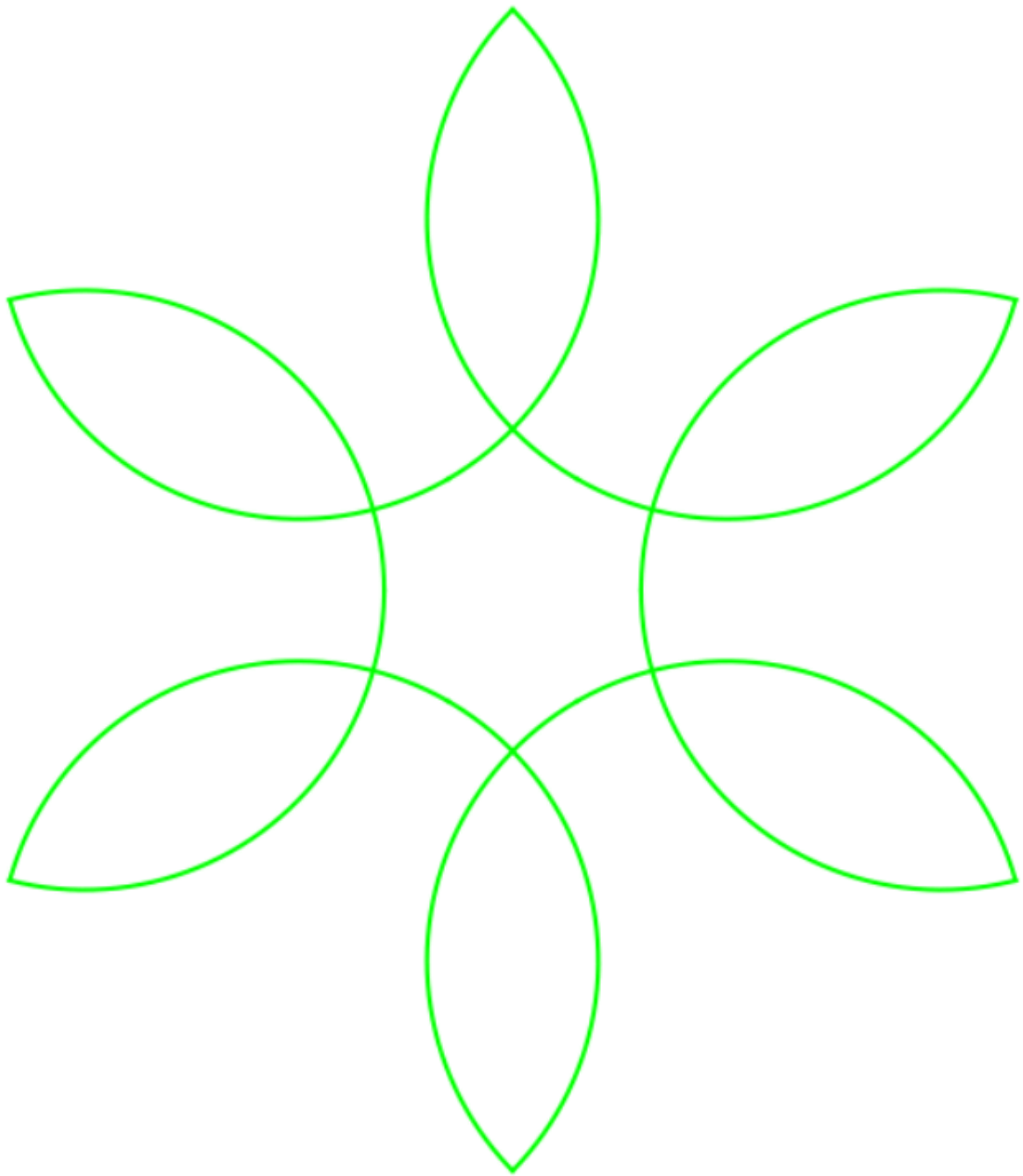
最简单的曼陀罗花就是几个相切圆弧其中 $R=1, r=0.5$



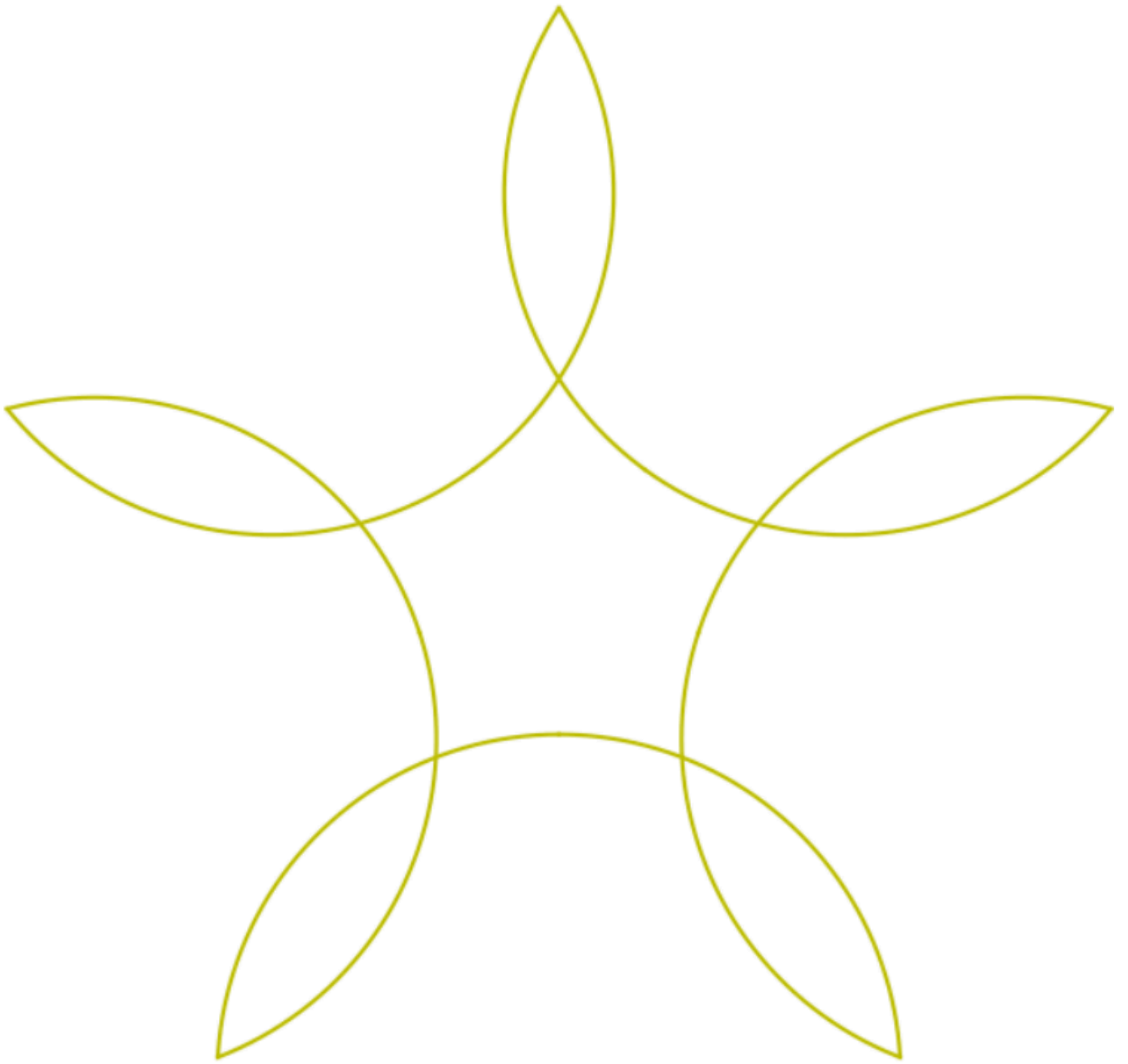
大半径不变，小半径增大，这里 $R=1, r=0.6$



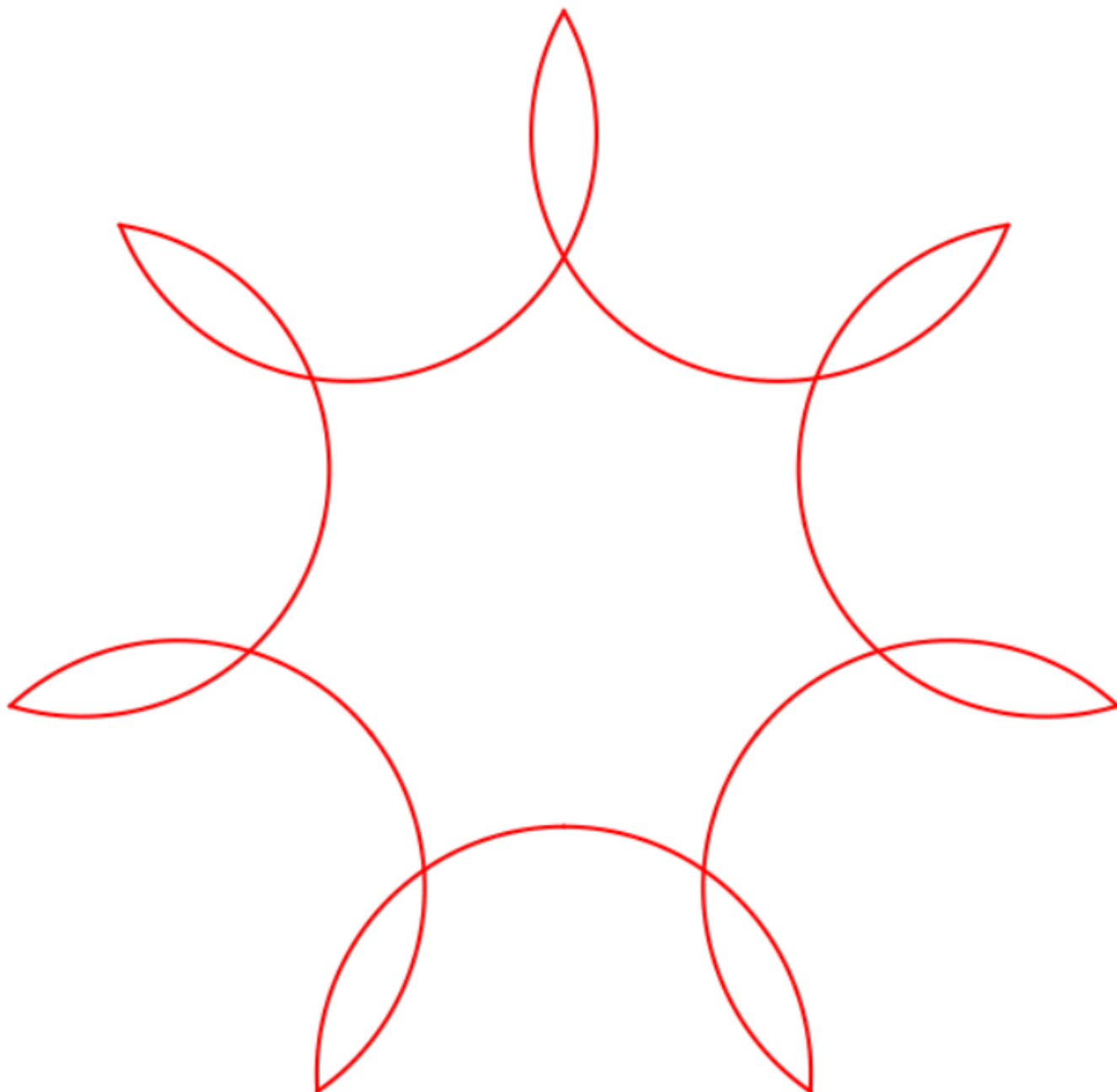
大半径不变，小半径再增大，花瓣就更饱满了，这里 $R=1$ ， $r=0.7$



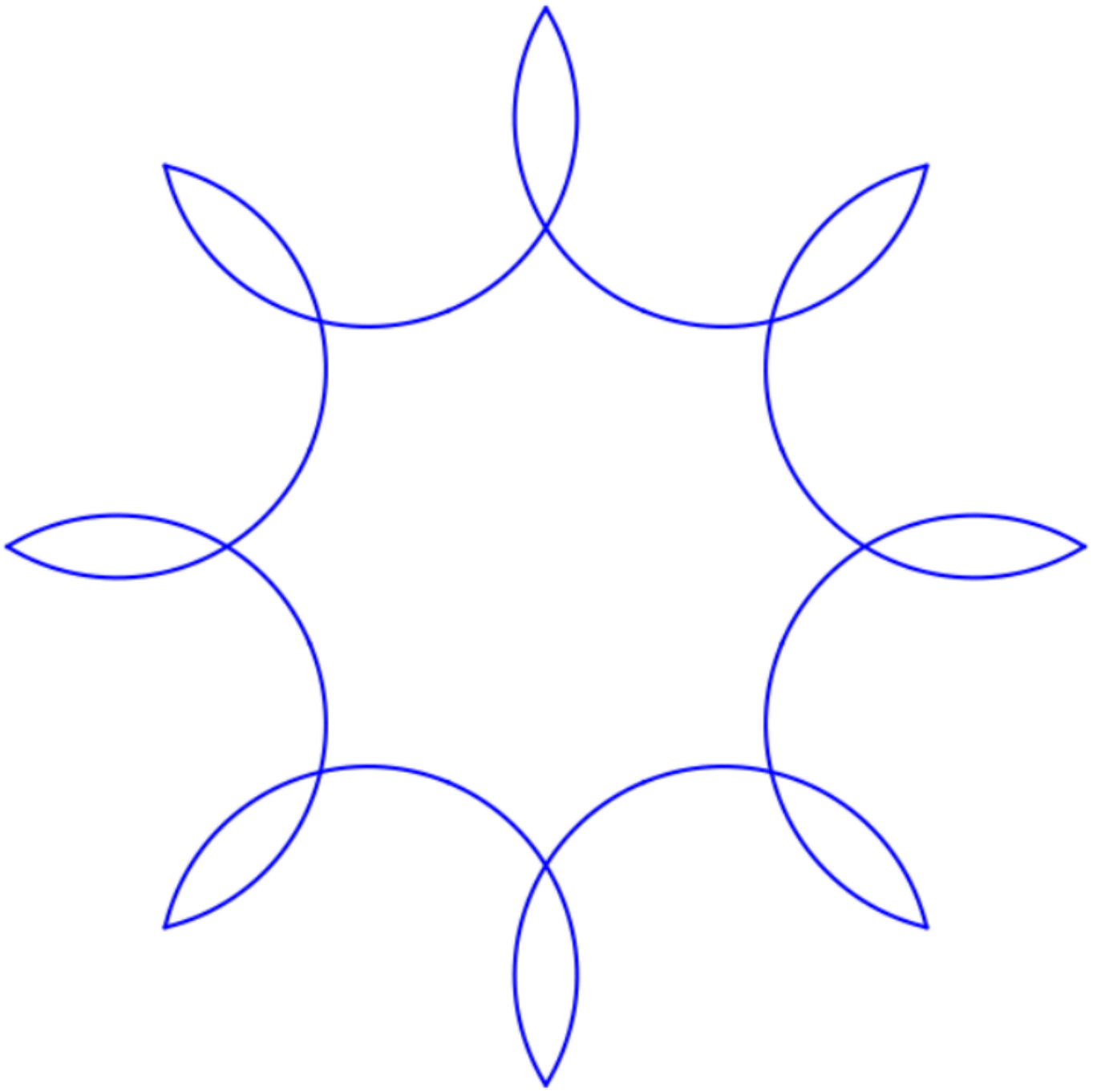
同样，5瓣花也可以，我的代码做了推广普适性，可以适用于任意数量的花瓣，这里 $R=1$ ， $r=0.7$ ， $n=5$ ， $N=5$ 但是我感觉它不是那么自然，因为尺规作图画这个很麻烦，毕竟涉及到黄金比例的东西负面的几何个人感觉非常人为不顺自然，而六瓣的最自然



7瓣单层曼陀罗花, $R=1$, $r=0.5$, $n=7$, $N=7$

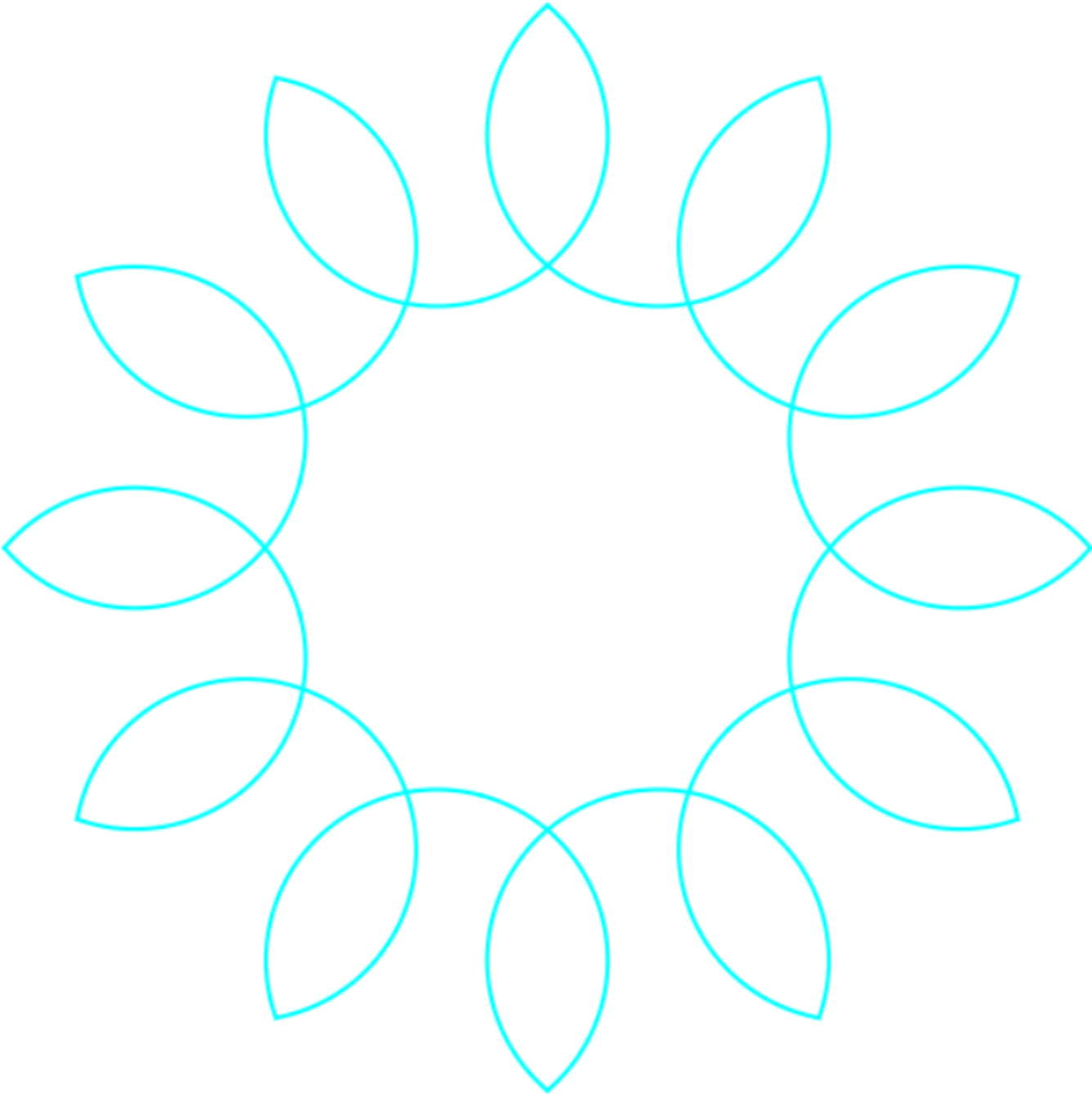


8瓣曼陀罗花在地球自然界中也许不存在，但在建筑宗教中的图里经常看到

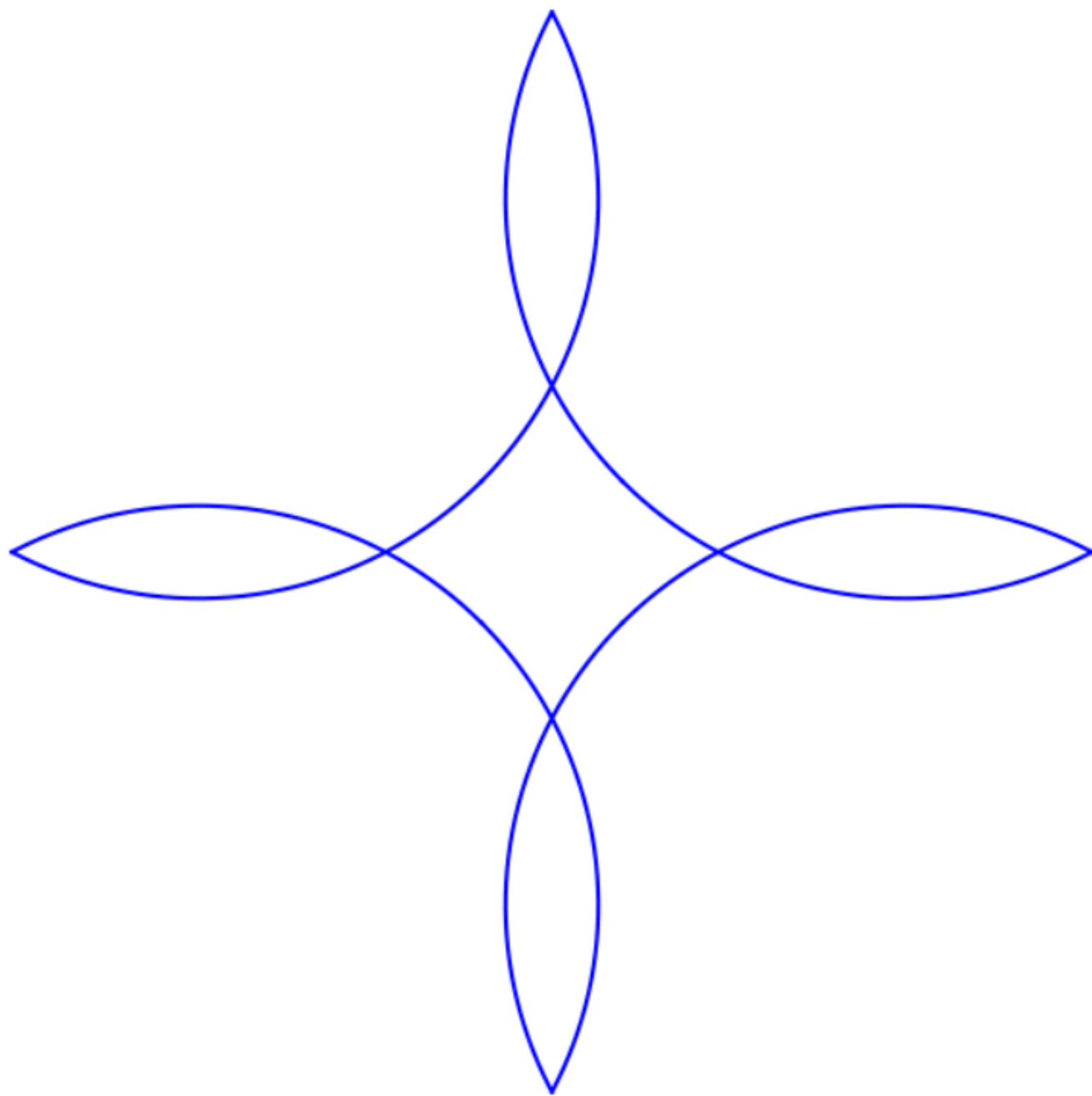


12瓣的曼陀罗花大概最完美，基于12进制的数学最完美不是吹的，在几何中是可以证明

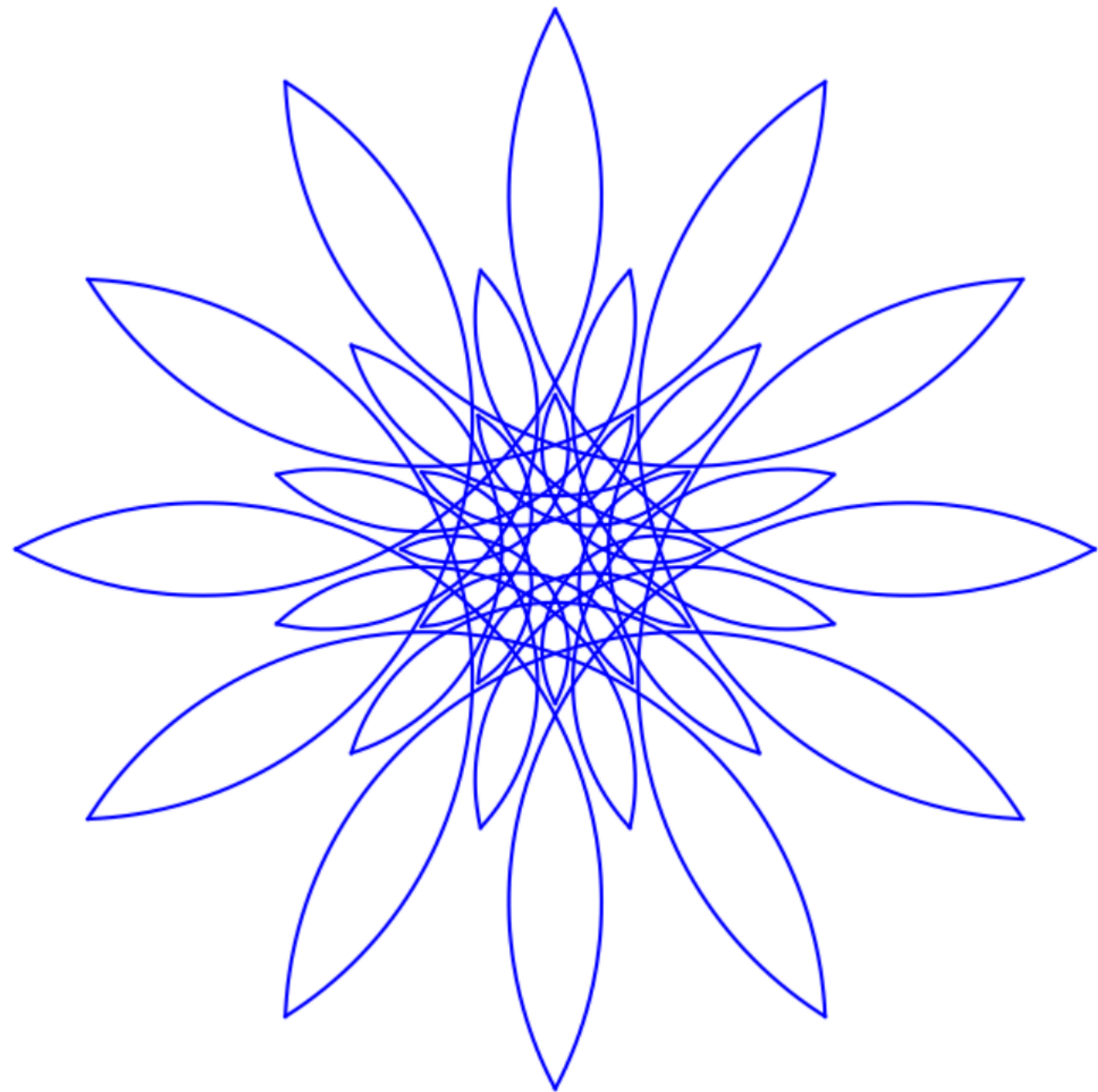
的, $R=1$, $r=0.4$, $n=12$, $N=12$



4瓣的也很完美，这里 $R=1$ ， $r=0.8$ ， $n=4$ ， $N=4$



下面的三层12瓣曼陀罗花基于上面的4瓣曼陀罗花组合构造，我已经把它写成Python函



曼陀罗花简介

曼陀罗花又叫洋金花、大喇叭花、山茄子、夕颜、醉心花、狗核桃、醉仙桃等，果实名为：狗核桃、毛苹果。为茄科曼陀罗属、一年生直立草本植物，原产印度，现广泛分布全球，我国各地均有野生或栽培，主产在华南地区，以广西最多。花期6 - 8月，果期6 - 10月，喜温暖、向阳及排水良好的砂质壤土。多野生在田间、沟旁、道边、河岸、山坡等地方，主要危害棉花、豆类、薯类、蔬菜等，有剧毒。曼陀罗中化学成分主要有东莨菪碱（Scopolamine）、莨菪碱（Hyoscyamine），其次有阿托品（atropine）、曼陀罗素（daturine）等。在古代曼陀罗常作为蒙汗药的主要成分。我国常见种有曼陀罗、毛曼陀罗、白花曼陀罗三种，花均为白色或微带淡黄绿色，单瓣。

拉丁学名	<i>Dature Stramonium Datura L</i>	科	茄科
别 名	醉心花、狗核桃、洋金花、枫茄花、万桃花、闹羊花、野麻子	属	曼陀罗属
界	植物界	种	植物种类
门	被子植物门	亚 门	种子植物亚门
纲	双子叶植物纲	亚 属	喇叭花属
		分布区域	印度，在我国主产在华南地区，以广西最多
		中文学名	曼陀罗

茎粗壮直立，株高50-150cm，茎上部二叉分枝，幼枝被短柔毛，下部木质化。全株光滑无毛，有时幼叶上有疏毛。叶宽卵形，顶端渐尖，基部不对称楔形，边缘有不规则波状浅裂，裂片三角形，两面沿脉及边缘俱疏生短柔毛，叶柄长3~5cm。花单生于叶腋和茎枝分叉处，直立，花色因品种不同而呈白色、红色、紫色等，花萼筒状，稍有棱裂，长4~6cm，顶端5裂，不紧贴花筒，花冠漏斗形，长5-10cm，直径3~4cm，5浅裂，裂片有短尖头，雄蕊5个，子房卵形，密被柔针毛，不完全4室，果实为蒴果直立，长约4cm，宽约3cm，表面密生硬刺，秋冬令成熟，成熟后作四瓣分裂，果仁稍带甜味，种子扁圆肾形，棕褐色，直径约1cm。

曼陀罗花喜生长在阳光充足的灌丛中、草地上、住宅旁、路边或河沟边，对栽培要求不严，有较强的抗旱能力，以种子越冬。曼陀罗花花期很长，只要温度合适一年四季都可开花，野外生长的曼陀罗花花期在5月到9月之间，果实成熟期于6月到10月之间。曼陀罗花靠种子繁殖。

曼陀罗花的起源

曼陀罗原产于印度，它的名称带着深厚的神秘宗教色彩。曼陀罗是梵文Mandala的译音。另译“曼荼罗”、“满拿啰”等。其意译为“坛”、“坛场”，取平坦之义。李时珍在《本草纲目》中详细地记载了曼陀罗花的来历，“法华经言佛说法时，天雨曼陀罗花。又道家北斗有陀罗星使者，手执此花。故后人因以名花。曼陀罗，梵言杂色也。”在佛经中，曼陀罗花是适意的意思，藏传佛教里有关微观宇宙的模型就叫“曼陀罗”，即佛家所谓“一花一世界，一叶一如来”。它包含着洞察幽明，超然觉悟，幻化无穷的精神

曼陀罗花花语

- 粉色曼陀罗——适意
- 绿色曼陀罗——希望
- 黑色曼陀罗——不可预知的黑暗、死亡和颠沛流离的爱
- 金色曼陀罗——幸福

白色曼陀罗——麻醉

紫色曼陀罗——恐怖

蓝色曼陀罗——爱情欺骗

我研究曼陀罗花的起源

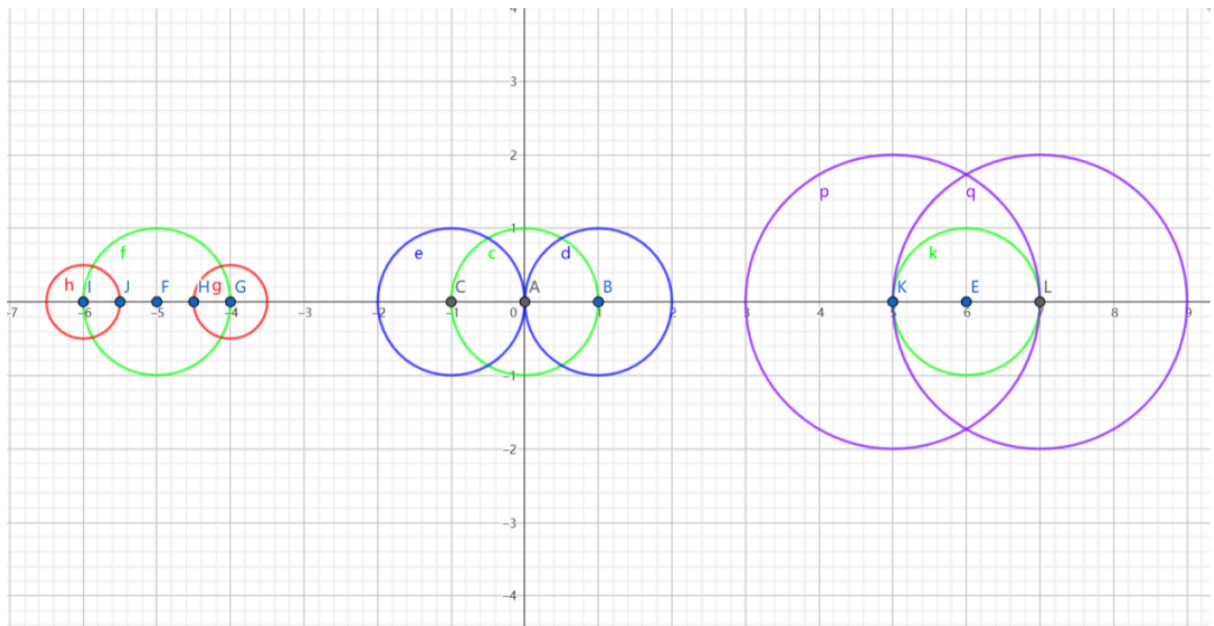
看我知乎的人大概都知道我在研究和发展一门我取名为《圈点动力学》的科学，三月份我已经发表了它的前奏《圈点艺术》(可点链接下载)，形成了基本的研究框架与思路，但由于《圈点动力学》涉及范围太广了，涉及数学，物理，化学，生物，计算机科学，音乐，佛学，道学，玄学...不是每个人都看得懂感兴趣和能理解，所以先来以艺术的方式欣赏，因为科学与艺术本就是一枚硬币的两面，不同人对这枚硬币的两面各有青睐，虽然我两面都喜欢都涉及。

其中看过我写的书和之前一些文章的朋友都知道花是我的重点研究对象，而且我研究莲花最早也最多(2013年我读高二就开始了)，后来我知道了莲花又分睡莲和荷花(马蹄莲不暂且不归入，因为马蹄莲是螺旋形成的)，而我最早研究的是睡莲，荷花是我看了一个昴宿星人的视频讲解麦田怪圈的几何图时才恍然大悟的(它是罗丹线圈几何的部分)。

之前一直在纸上尺规作图画着研究，所以很麻烦很费力而且精度不高进度很慢，大学读的软件工程专业也一直想通过程序解决来画着研究，但是学计算机磕磕绊绊，没有人带我，也不知道方向与门路，他们都热衷于烂大街的Java,赚钱的后端，易学的前端，都为来金钱去了，根本没有志同道合的人...直到毕业后，我天天刷知乎学习计算机知识，感觉大学四年还没有毕业后一年刷知乎的收获多，以前喜欢上CSDN，现在它就知道会员收费，果断放弃这些资本了，信息本来就应该共享，知乎在这方面好多了。刷知乎让我找到了技术实现的方式——Python，简洁高效，功能强大，应用广泛。于是乎去年开始我就尝试借助matplotlib库写Python代码来画图了，数学是计算机程序的灵魂，我琢磨半天莲花的几何怎么通过代码实现，终于找到了它的表达式，应而实现了睡莲图的绘画。接着过年开始写《圈点艺术》时想要画荷花，于是花了几天安析数学编码测试终于成功了。至于曼陀罗花我本来是没有涉及的，但根据完整性分析，还差一种花的在几何上是和里的，于是曼陀罗花也研究了。

看下图，(懒得重新画了，直接引用书中的)：

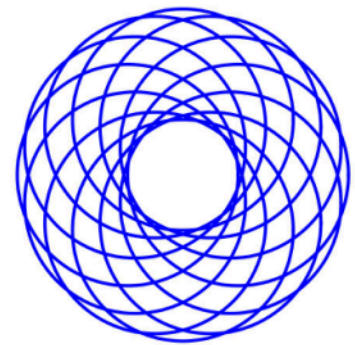
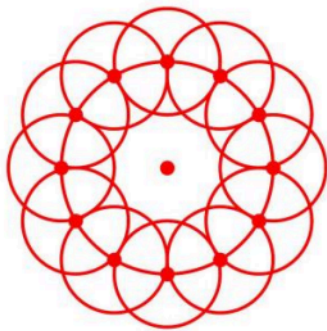
无非就是这三种情况，同样几何上也是这样：



$r < R$

$r = R$

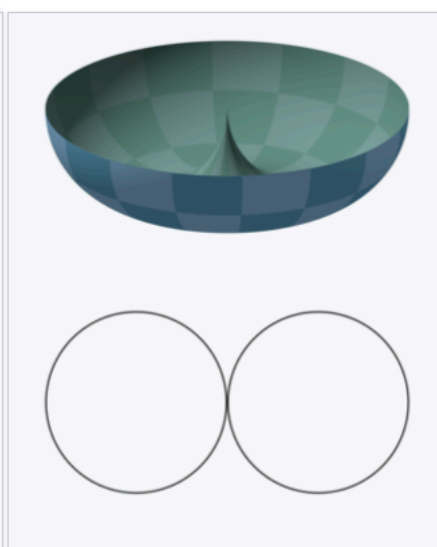
$r > R$



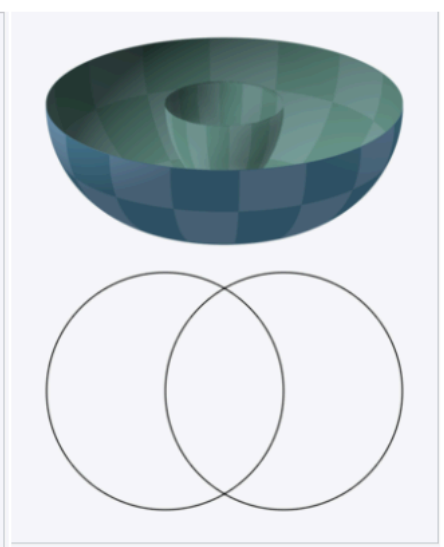
绿色中心圆上的振动圆(这里分别是红蓝紫圆)无非三种情况，其中左边的结构即是曼陀罗花，中间是睡莲结构，右边是荷花结构，这是二维情况，三维的对应环面分别是：(ring)环，角环，纺锤环。



$R > r$: ring torus or anchor ring



$R = r$: horn torus



$R < r$: self-intersecting spindle torus

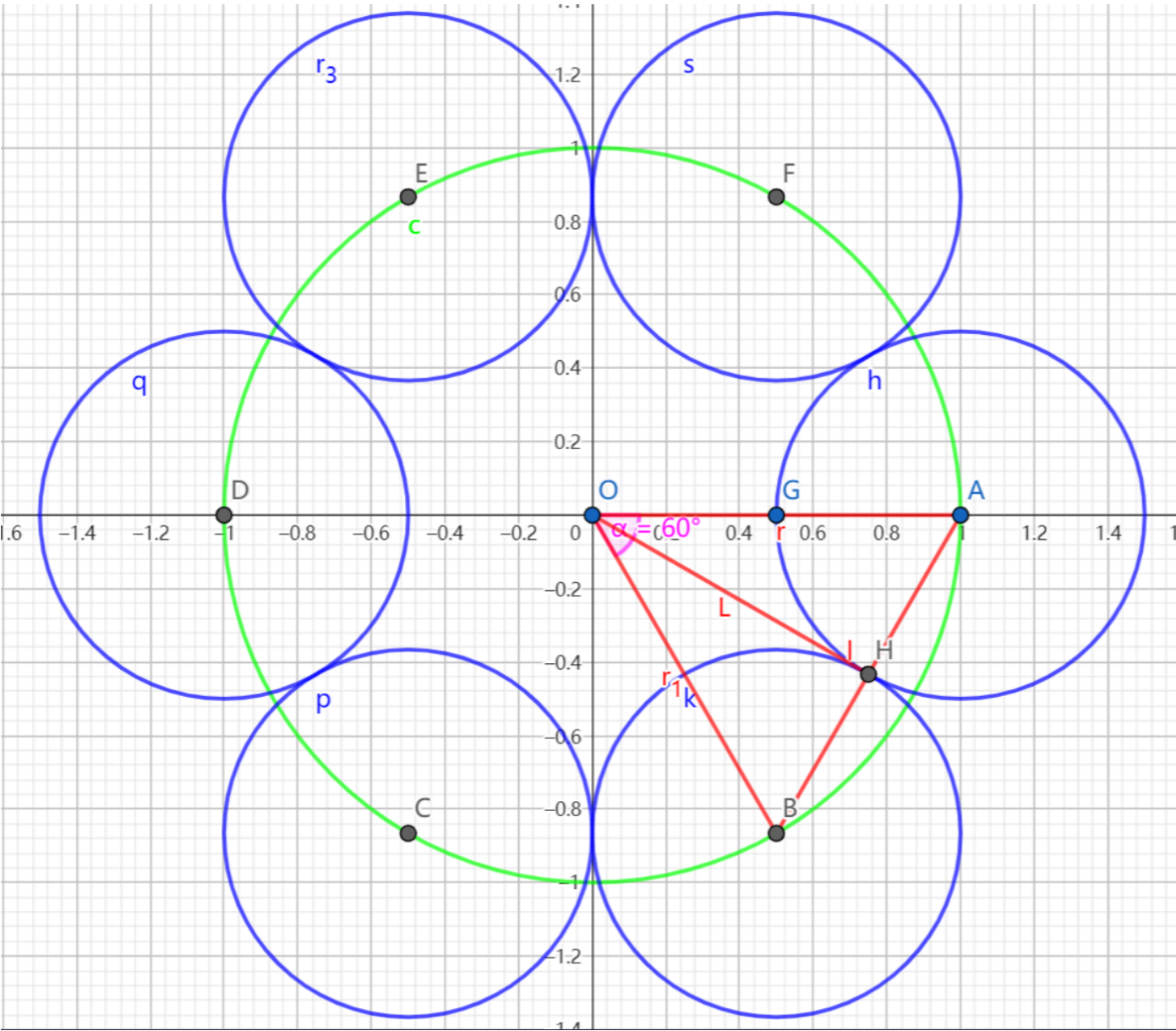
感兴趣去翻书，我就不偏离主题了。

但是让我没想到的是曼陀罗花的数学是最复杂的，睡莲最简单，荷花其次，但令我以外的是曼陀罗花的数学具有普适性，也就是说睡莲和荷花的数学是曼陀罗花的特殊情况,可以通过曼陀罗花的数学来演绎，这就导致我曼陀罗花程序编出来竟然可以改参数画睡莲和荷花！一劳永逸，早知道我就不那么费力的开发睡莲荷花程序了 $\circ(\pi\sim\pi)\circ$ ，这这情况在数学和物理上也屡见不鲜，当科学们在更高角度看以前的原理时，才发现好简单，只需要新理论演绎一下，麦克斯韦方程组的发现对电磁学的影响就是这样。

睡莲和荷花的数学我空了单独写文章，今天把曼陀罗花了事。

曼陀罗花的数学

好了，到此是数学爱好者和理科生的天下，文科生请留步，各位理科生同学打起精神，你们喜欢推导的公式来了^_^



我们用 n 表示单层曼陀罗花瓣的数量,对应于中心圆上均匀分布的波的振动点相位数。
 R 表示中心圆的半径， r 表示振动点形成圆的半径。对于曼陀罗花，这里 $r < R$

α 是中心圆周上相邻两振动点的圆心角，这里即 $\angle AOB$

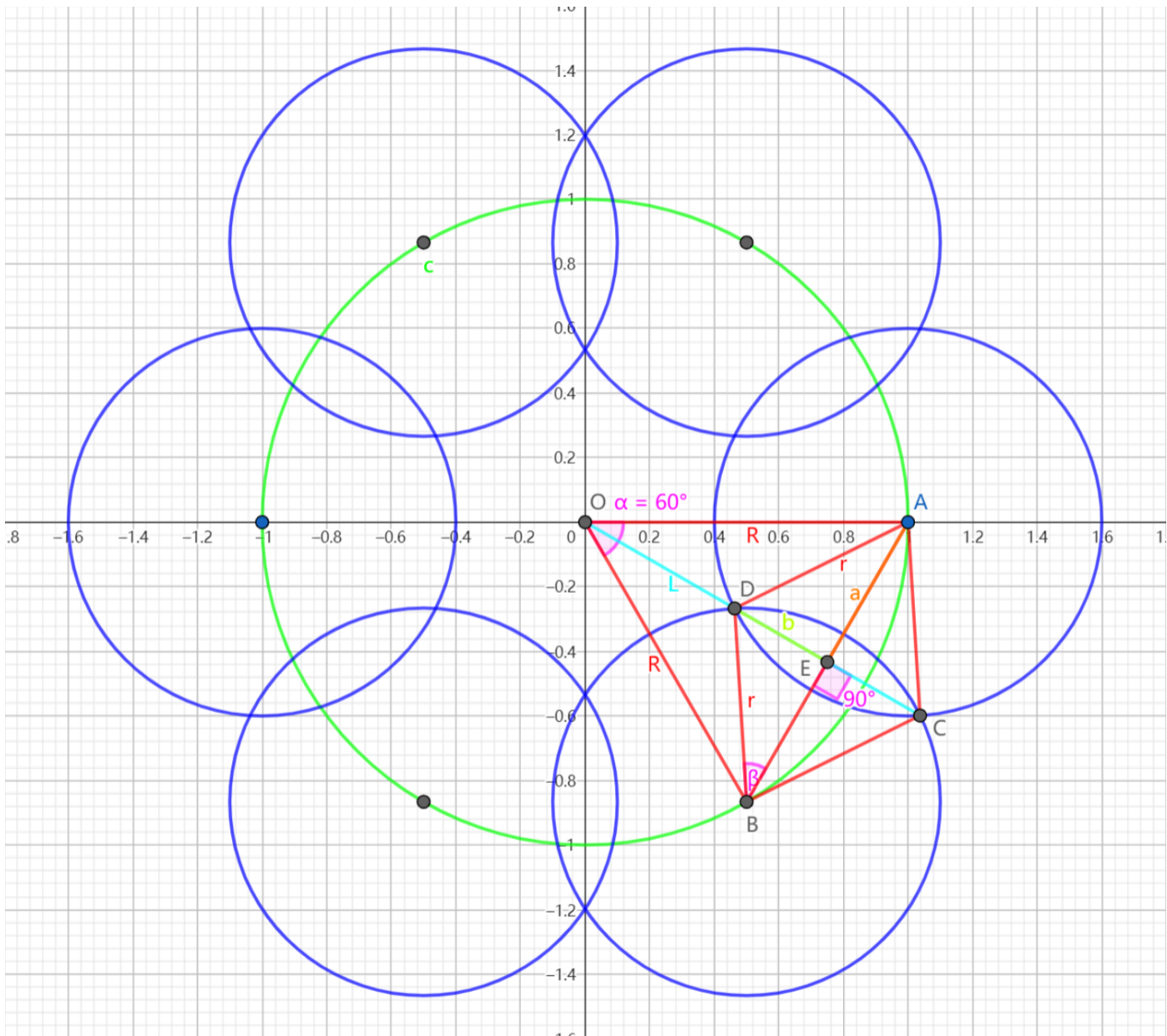
L 是花瓣长度，这里即是 OH , L_c 是中心到花瓣尖角的距离(L_c 的c取自单词center)，这里也是 OH ，即 $L = L_c$ ， L_s 是中心到花瓣的最短距离(L_s 的s取自单词short)

下面我们开始推导：

$$\alpha = \frac{2\pi}{n}$$

$$a = 2r \sin \frac{\alpha}{2}$$

由于这里 $r = \frac{R}{2} = 0.5$ ，所以 $a = R \sin \frac{\alpha}{2} = \frac{1}{2}$ ，这是曼陀罗花的特殊情况，我们试着推广一下。



$$L = DC, L_c = OC, L_s = OD$$

$$\alpha = \frac{2\pi}{n}$$

$$a = R \sin \frac{\alpha}{2} = r \cos \beta$$

$$b = r \sin \beta, a^2 + b^2 = r^2$$

$$\begin{aligned}\beta &= \arcsin \frac{b}{r} \\ &= \arcsin \frac{\sqrt{r^2 - a^2}}{r} \\ &= \arcsin \sqrt{1 - \frac{a^2}{r^2}} \\ &= \arcsin \sqrt{1 - \frac{(R \sin \frac{\alpha}{2})^2}{r^2}} \\ &= \arcsin \sqrt{1 - \frac{(R \sin \frac{\pi}{n})^2}{r^2}} \\ &= \arcsin \sqrt{1 - (\frac{R}{r})^2 \sin^2 \frac{\pi}{n}}\end{aligned}$$

或

$$\begin{aligned}\beta &= \arccos \frac{a}{r} = \arccos \frac{R \sin \frac{\alpha}{2}}{r} \\ &= \arccos \frac{R \sin \frac{\pi}{n}}{r}\end{aligned}$$

花瓣相关角度:

$$\begin{aligned}\theta_{arc} &= \angle OAB + \angle BAC = \frac{\pi}{2} - \frac{\alpha}{2} + \beta \\ &= \frac{\pi}{2} - \frac{\pi}{n} + \arcsin \sqrt{1 - (\frac{R}{r})^2 \sin^2 \frac{\pi}{n}} \\ &= \frac{\pi}{2} - \frac{\pi}{n} + \arccos \frac{R \sin \frac{\pi}{n}}{r}\end{aligned}$$

$$\begin{aligned}\theta_{petal} &= 2\theta_{arc} = \pi - \alpha + 2\beta \\ &= \pi - \frac{2\pi}{n} + 2 \arcsin \sqrt{1 - (\frac{R}{r})^2 \sin^2 \frac{\pi}{n}} \\ &= \pi - \frac{2\pi}{n} + 2 \arccos \frac{R \sin \frac{\pi}{n}}{r}\end{aligned}$$

花瓣相关长度:

$$\begin{aligned}L &= 2b = 2r \sin \beta \\ &= 2r \sqrt{1 - (\frac{R}{r})^2 \sin^2 \frac{\pi}{n}}\end{aligned}$$

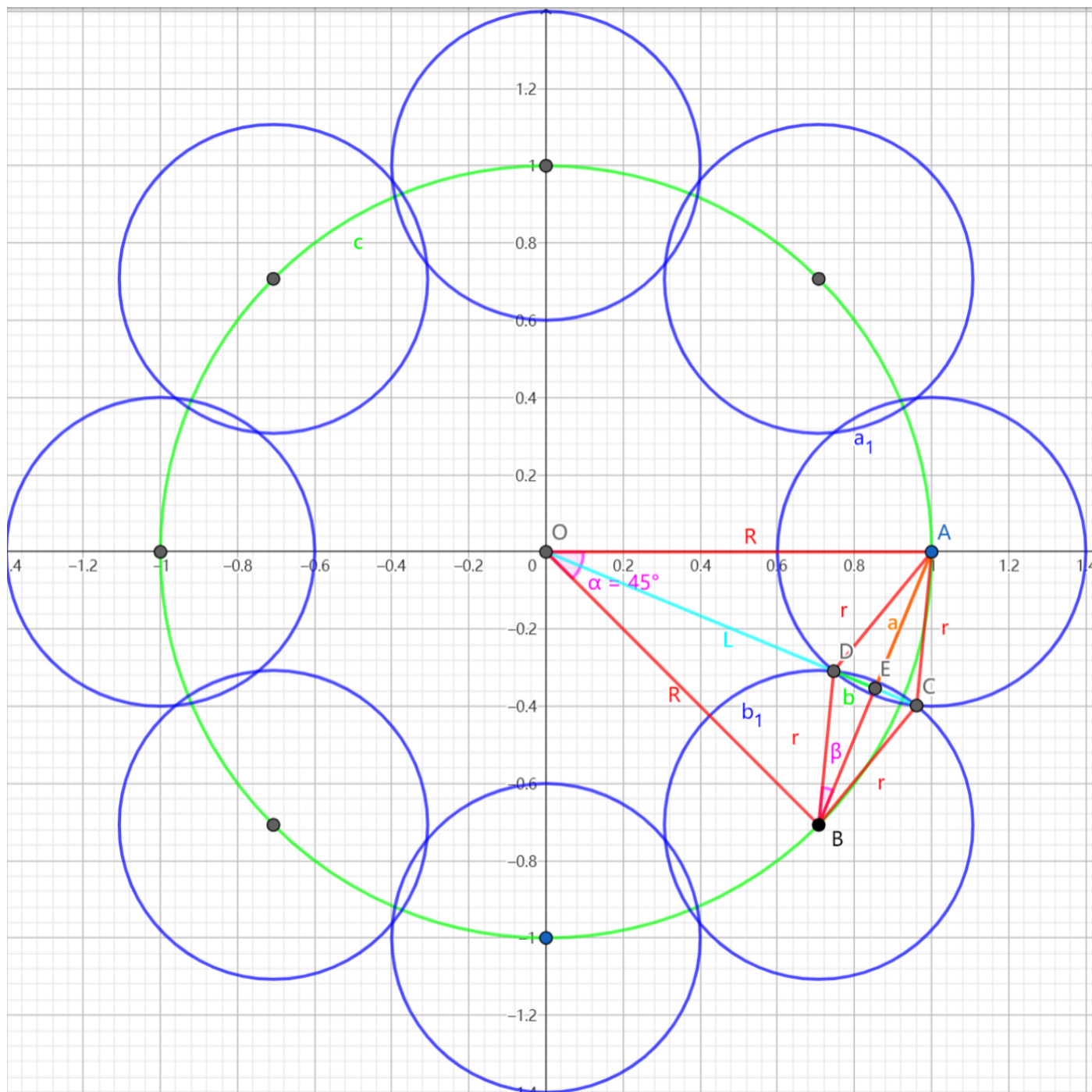
$$= 2\sqrt{r^2 - R^2 \sin^2 \frac{\pi}{n}}$$

$$\begin{aligned} L_c &= R \cos \frac{\alpha}{2} + b = R \cos \frac{\alpha}{2} + r \sin \beta \\ &= R \cos \frac{\pi}{n} + \sqrt{r^2 - R^2 \sin^2 \frac{\pi}{n}} \end{aligned}$$

$$\begin{aligned} L_s &= R \cos \frac{\alpha}{2} - b = R \cos \frac{\alpha}{2} - r \sin \beta \\ &= R \cos \frac{\pi}{n} - \sqrt{r^2 - R^2 \sin^2 \frac{\pi}{n}} \end{aligned}$$

下面我们再来看以上公式是否满足一般性：

当 $R = 1$ 不变, $n = 8$, $r = 0.4$ 时如下图：



可见与上图的区别就是 α, β 变小了，圆A，圆B依然相交如果再进一步推广，当圆A，圆B相切的时候有什么情况呢？

根据上面的图可以看出：当 $\beta = 0$ 也就是 $r = a$ 时，A, B两圆相切 由前面我们知道：

$$\begin{aligned} \because a &= R \sin \frac{\alpha}{2} = R \sin \frac{\pi}{n} \\ \therefore r &= R \sin \frac{\pi}{n} \end{aligned}$$

也就是说，当 n 为定值时：

$$\begin{cases} \text{若 } r < R \sin \frac{\pi}{n} : A, B \text{ 两圆相离，不能形成我们的曼陀罗花} \\ \text{若 } r = R \sin \frac{\pi}{n} : A, B \text{ 两圆相切，形成我们的特殊曼陀罗花} \\ \text{若 } r > R \sin \frac{\pi}{n} : A, B \text{ 两圆相交，形成我们的普通曼陀罗花} \end{cases}$$

曼陀罗花的画图算法

导入需要的matplotlib和numpy库：

```
import matplotlib.pyplot as plt
import numpy as np
```

画圆弧算法：

```
def arc_degree(center, radius, angle1, angle2, color='b'):
    if angle1 < angle2:
        angle = np.linspace(angle1, angle2, 1000)
        x = center[0] + radius * np.cos(angle)
        y = center[1] + radius * np.sin(angle)
        plt.axis('equal')
        plt.plot(x, y, color)

def arc_degree_inverse(center, radius, angle1, angle2, color='b'):
    angle = np.linspace(angle2-2*np.pi, angle1, 1000)
    x = center[0] + radius * np.cos(angle)
    y = center[1] + radius * np.sin(angle)
    plt.axis('equal')
    plt.plot(x, y, color)
```

arc_degree()函数在选定中心从起始角度angle1，终止角度angle2之间逆时针画圆弧，arc_degree_inverse()函数在选定中心从起始角angle1终止角度angle2之间顺时针画圆弧。

画花算法：

以下是核心算法,花瓣方式画花

```
# 花瓣

def n_mandala_petal(center, R, r, n, theta=0, color='b'):
    alpha = 2*np.pi/n
    a = R*np.sin(np.pi/n)
    beta = np.arccos((a)/r)
    theta_arc = np.pi/2-np.pi/n+np.arccos((a)/r)
    theta_petal = 2*theta_arc
    # circle((0, 0), R, 'g')
    # circle((0, 0), R/2, 'g')
    center1 = (np.cos(theta+alpha/2)*R +
```



```

        center[0], np.sin(theta+alpha/2)*R+center[1])
center2 = (np.cos(theta+alpha/2-2*np.pi/n)*R +
        center[0], np.sin(theta+alpha/2-2*np.pi/n)*R+center[1])
if abs(r - a) < 1e-12:
    arc_degree(center1, r, np.pi+alpha/2+theta,
        np.pi+alpha/2+theta+theta_arc, color)
    arc_degree(center2, r, np.pi/2+theta,
        np.pi/2+theta+theta_arc, color)
# if r == R:
#     print("r=", r, ",a=", a)
#     print('r=a,r=R形成睡莲花瓣。')
# elif r > R:
#     print("r=", r, ",a=", a)
#     print('r=a,r>R形成荷花花瓣。')
# elif r < R:
#     print("r=", r, ",a=", a)
#     print('r=a,r<R形成特殊曼陀罗花瓣。')
elif r > a:
    arc_degree(center1, r, np.pi+alpha/2+theta,
        np.pi+alpha/2+theta+theta_arc, color)
    arc_degree(center2, r, np.pi/2-beta+theta,
        np.pi/2-beta+theta+theta_arc, color)
# if r == R:
#     print("r=", r, ",a=", a)
#     print('r>a,r=R形成睡莲花瓣。')
# elif r > R:
#     print("r=", r, ",a=", a)
#     print('r>a,r>R形成荷花花瓣。')
# elif r < R:
#     print("r=", r, ",a=", a)
#     print('r>a,r<R形成普通曼陀罗花瓣。')
elif r < a:
    print("r=", r, ",a=", a)
    print('r<a,不能形成花瓣。')
# plt.plot(center1[0], center1[1], marker='o', color='r')
# plt.plot(center2[0], center2[1], marker='o', color='b')

```

最难也最头疼的部分，我想这个花了好几周，即使数学领悟了编码也费力好几天，直到有一天我静下心来读《妙法莲华经》一下午后突然有灵感，晚上睡觉前一下就编码成功了，还有我的很多研究灵感都出自《妙法莲华经》，第一次一个修行老师给我免费邮寄佛经让我选时由于爱好和缘分我就第一时间请了《妙法莲华经》，《楞严经》和《八十八佛大忏悔文》，开悟的楞严，成佛的法华不是吹动，巨大能量加持的同时给你宇宙真相的灵感，就科技方面我觉得《妙法莲华经》对人帮助很大，世界宇宙真相及哲学哲理方面《楞严经》不愧为王！

center参数是画图中心，R参数是花瓣外径（还记得那三种结构的绿圆吗？），r参数是花瓣内径（那三种结构中的红、蓝、紫圆），n参数是花瓣性质，theta参数是画图旋转角度，color参数是画图颜色。

$\alpha, \beta, \theta_{arc}, \theta_{petal}, a$ 分别是我们上面数学讨论的

center1, center2 分别是花瓣两个弧的两个中心点，然后根据条件判断可以调用 arc_degree() 画圆弧了

以下是上面的伴随算法，花弧方式画花

```
def n_mandala_arc(center, R, r, n, theta=0, color='b'):
    alpha = 2*np.pi/n
    a = R*np.sin(np.pi/n)
    beta = np.arccos((a)/r)
    theta_arc = np.pi/2-np.pi/n+np.arccos((a)/r)
    theta_petal = 2*theta_arc
    # circle((0, 0), R, 'g')
    # circle((0, 0), R/2, 'g')
    center1 = (np.cos(theta+alpha/2)*R +
               center[0], np.sin(theta+alpha/2)*R+center[1])
    center2 = (np.cos(theta+alpha/2-2*np.pi/n)*R +
               center[0], np.sin(theta+alpha/2-2*np.pi/n)*R+center[1])
    if abs(r - a) < 1e-12:
        arc_degree(center1, r, np.pi/2+alpha+theta,
                   np.pi/2+alpha+theta+theta_petal, color)
        arc_degree(center2, r, np.pi/2+theta,
                   np.pi/2+theta+theta_petal, color)
    if r == R:
        print("r=", r, ",a=", a)
        print('r=a,r=R形成睡莲花弧。')
    elif r > R:
        print("r=", r, ",a=", a)
        print('r=a,r>R形成荷花花弧。')
    elif r < R:
        print("r=", r, ",a=", a)
        print('r=a,r<R形成特殊曼陀罗花弧。')
    elif r > a:
        arc_degree(center1, r, np.pi/2+alpha-beta+theta,
                   np.pi/2+alpha-beta+theta+theta_petal, color)
        arc_degree(center2, r, np.pi/2-beta+theta,
                   np.pi/2-beta+theta+theta_petal, color)
    if r == R:
        print("r=", r, ",a=", a)
        print('r>a,r=R形成睡莲花弧。')
    elif r > R:
        print("r=", r, ",a=", a)
        print('r>a,r>R形成荷花花弧。')
    elif r < R:
        print("r=", r, ",a=", a)
        print('r>a,r<R形成普通曼陀罗花弧。')
    elif r < a:
        print("r=", r, ",a=", a)
        print('r<a,不能形成花瓣。')
    # plt.plot(center1[0], center1[1], marker='o', color='r')
    # plt.plot(center2[0], center2[1], marker='o', color='b')
```

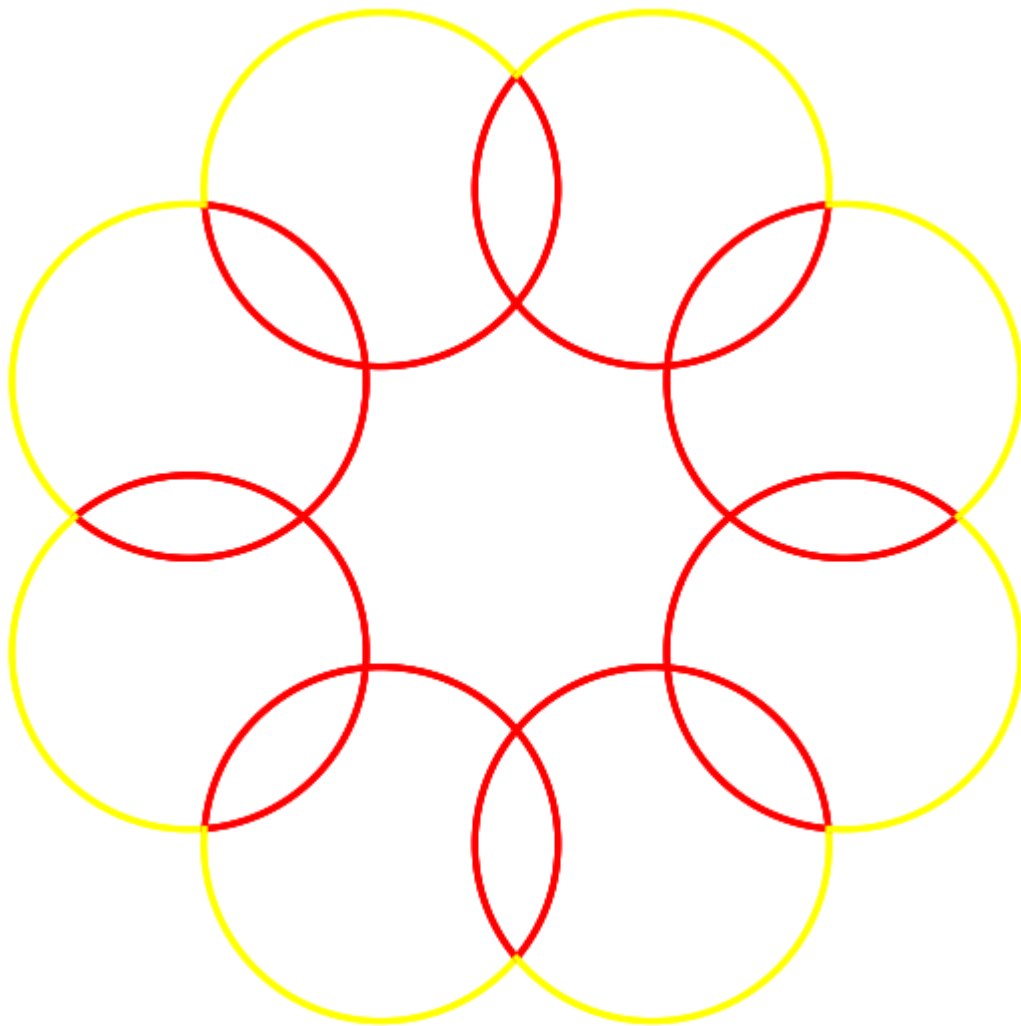
可能有同学不明白怎么两个算法，不嫌麻烦吗？因为画花场得用这个花弧方式画。代码差不多，只是画的角度参数不同

花弧带场算法

带场花弧

```
def n_mandala_arc_with_field(center, R, r, n, theta=0, color='b',
colorfield='#ff0'):
    alpha = 2*np.pi/n
    a = R*np.sin(np.pi/n)
    beta = np.arccos((a)/r)
    theta_arc = np.pi/2-np.pi/n+np.arccos((a)/r)
    theta_petal = 2*theta_arc
    # circle((0, 0), R, 'g')
    # circle((0, 0), R/2, 'g')
    center1 = (np.cos(theta+alpha/2)*R +
                center[0], np.sin(theta+alpha/2)*R+center[1])
    center2 = (np.cos(theta+alpha/2-2*np.pi/n)*R +
                center[0], np.sin(theta+alpha/2-2*np.pi/n)*R+center[1])
    if abs(r - a) < 1e-12:
        arc_degree(center1, r, np.pi/2+alpha+theta,
                    np.pi/2+alpha+theta+theta_petal, color)
        arc_degree_inverse(center1, r, np.pi/2+alpha+theta,
                            np.pi/2+alpha+theta+theta_petal, colorfield)
        arc_degree(center2, r, np.pi/2+theta,
                    np.pi/2+theta+theta_petal, color)
        arc_degree_inverse(center2, r, np.pi/2+theta,
                            np.pi/2+theta+theta_petal, colorfield)
    elif r > a:
        arc_degree(center1, r, np.pi/2+alpha-beta+theta,
                    np.pi/2+alpha-beta+theta+theta_petal, color)
        arc_degree_inverse(center1, r, np.pi/2+alpha-beta+theta,
                            np.pi/2+alpha-beta+theta+theta_petal, colorfield)
        arc_degree(center2, r, np.pi/2-beta+theta,
                    np.pi/2-beta+theta+theta_petal, color)
        arc_degree_inverse(center2, r, np.pi/2-beta+theta,
                            np.pi/2-beta+theta+theta_petal, colorfield)
    elif r < a:
        print("r=", r, ",a=", a)
        print('r<a,不能形成花瓣。')
    # plt.plot(center1[0], center1[1], marker='o', color='r')
    # plt.plot(center2[0], center2[1], marker='o', color='b')
```

没什么难度，就是上面的算法多了arc_degree_inverse()调用，同一个圆上的参数都相同，以此闭合形成圆，交点分界线内是花弧，分界线外是场。如下图。



下面的算法都是以上的应用了自己理解自己悟，搞不明白问AI，ChatGPT，copilot，codegeex这些都可以。

```
# 一朵向上花瓣
```

```
def one_mandala_petal(center, R, r, n, theta=0, color='b'):  
    n_mandala_petal(center, R, r, n, theta+np.pi/2, color)
```

```
# 一朵向上花弧
```

```
def one_mandala_arc(center, R, r, n, theta=0, color='b'):  
    n_mandala_arc(center, R, r, n, theta+np.pi/2, color)
```

```
# 一朵向上花瓣场
```

```
def one_mandala_arc_with_field(center, R, r, n, theta=0, color='b',  
colorfield='#ff0'):  
    n_mandala_arc_with_field(center, R, r, n, theta +  
np.pi/2, color, colorfield)
```

```
# 花瓣形成的单层花
```

```
def mandala_flower_by_petal(center, R, r, N, n, theta, color='b'):  
    for i in range(0, N):  
        one_mandala_petal(center, R, r, n, 2*i*np.pi/N+theta, color)
```

```
# 花弧形成的单层花
```

```
def mandala_flower_by_arc(center, R, r, N, n, theta, color='b'):  
    for i in range(0, N):  
        one_mandala_arc(center, R, r, n, 2*i*np.pi/N+theta, color)
```

```
# 单层花带场
```

```
def mandala_flower_by_arc_with_field(center, R, r, N, n, theta, color='b',  
colorfield='#ff0'):  
    for i in range(0, N):  
        one_mandala_arc_with_field(center, R, r, n, 2*i*np.pi/N+theta, color)
```

测试代码

```
...  
算法函数测试  
...  
# 一朵向上花瓣测试:  
one_mandala_petal((0, 0), 1, 0.5, 6, 0, '#00f')  
# 一朵向上花瓣场测试:  
one_mandala_arc_with_field((0, 0), 1, 0.6, 6, 0, 'b', '#ff0')  
# 花瓣形成的单层花测试:  
mandala_flower_by_petal((0, 0), 1, 0.6, 6, 6, 0, '#0f0')  
mandala_flower_by_petal((0, 0), 1, 0.8, 4, 4, 0, '#00f')  
# 单层花带场测试:  
mandala_flower_by_arc_with_field((0, 0), 1, 0.8, 8, 8, 0, 'b', '#ff0')  
...
```

用户代码，也就是大家可以尝试DIY的代码

```
...  
以下是中文写好方便大改参数DIY的代码，  
两个函数区别是下面那个函数多了花场颜色参数。  
可以多写几个函数一起调用，相当于在画板上多次绘画，方便DIY。  
...  
# 花瓣形成的单层花  
中心 = (0, 0)  
大半径 = 1  
小半径 = 0.5  
单层花瓣数 = 8
```

```

花瓣多边形数 = 8 # (花瓣的固有特性)
花瓣旋转角度 = 0
花瓣颜色 = 'red'
花场颜色 = 'gold'
mandala_flower_by_petal(中心, 大半径, 小半径, 单层花瓣数, 花瓣多边形数, 花瓣旋转角度, 花瓣颜色)
mandala_flower_by_arc_with_field(
    中心, 大半径, 小半径, 单层花瓣数, 花瓣多边形数, 花瓣旋转角度, 花瓣颜色, 花场颜色)
plt.axis('equal')
plt.axis("off")
plt.show()

```

注释写的很明白了，大家根据自己的理解去玩玩吧！ ^_^

以下是完整代码：

```

import matplotlib.pyplot as plt
import numpy as np

"""通过角度画圆弧
"""

def arc_degree(center, radius, angle1, angle2, color='b'):
    if angle1 < angle2:
        angle = np.linspace(angle1, angle2, 1000)
        x = center[0] + radius * np.cos(angle)
        y = center[1] + radius * np.sin(angle)
        plt.axis('equal')
        plt.plot(x, y, color)

def arc_degree_inverse(center, radius, angle1, angle2, color='b'):
    angle = np.linspace(angle2-2*np.pi, angle1, 1000)
    x = center[0] + radius * np.cos(angle)
    y = center[1] + radius * np.sin(angle)
    plt.axis('equal')
    plt.plot(x, y, color)

...
画花算法
...
# 花瓣

def n_mandala_petal(center, R, r, n, theta=0, color='b'):
    alpha = 2*np.pi/n
    a = R*np.sin(np.pi/n)
    beta = np.arccos((a)/r)

```



```

theta_arc = np.pi/2-np.pi/n+np.arccos((a)/r)
theta_petal = 2*theta_arc
# circle((0, 0), R, 'g')
# circle((0, 0), R/2, 'g')
center1 = (np.cos(theta+alpha/2)*R +
            center[0], np.sin(theta+alpha/2)*R+center[1])
center2 = (np.cos(theta+alpha/2-2*np.pi/n)*R +
            center[0], np.sin(theta+alpha/2-2*np.pi/n)*R+center[1])
if abs(r - a) < 1e-12:
    arc_degree(center1, r, np.pi+alpha/2+theta,
                np.pi+alpha/2+theta+theta_arc, color)
    arc_degree(center2, r, np.pi/2+theta,
                np.pi/2+theta+theta_arc, color)
    # if r == R:
    #     print("r=", r, ",a=", a)
    #     print('r=a,r=R形成睡莲花瓣。')
    # elif r > R:
    #     print("r=", r, ",a=", a)
    #     print('r=a,r>R形成荷花花瓣。')
    # elif r < R:
    #     print("r=", r, ",a=", a)
    #     print('r=a,r<R形成特殊曼陀罗花瓣。')
elif r > a:
    arc_degree(center1, r, np.pi+alpha/2+theta,
                np.pi+alpha/2+theta+theta_arc, color)
    arc_degree(center2, r, np.pi/2-beta+theta,
                np.pi/2-beta+theta+theta_arc, color)
    # if r == R:
    #     print("r=", r, ",a=", a)
    #     print('r>a,r=R形成睡莲花瓣。')
    # elif r > R:
    #     print("r=", r, ",a=", a)
    #     print('r>a,r>R形成荷花花瓣。')
    # elif r < R:
    #     print("r=", r, ",a=", a)
    #     print('r>a,r<R形成普通曼陀罗花瓣。')
elif r < a:
    print("r=", r, ",a=", a)
    print('r<a,不能形成花瓣。')
# plt.plot(center1[0], center1[1], marker='o', color='r')
# plt.plot(center2[0], center2[1], marker='o', color='b')

```

花弧

```

def n_mandala_arc(center, R, r, n, theta=0, color='b'):
    alpha = 2*np.pi/n
    a = R*np.sin(np.pi/n)
    beta = np.arccos((a)/r)
    theta_arc = np.pi/2-np.pi/n+np.arccos((a)/r)
    theta_petal = 2*theta_arc
    # circle((0, 0), R, 'g')
    # circle((0, 0), R/2, 'g')
    center1 = (np.cos(theta+alpha/2)*R +
                center[0], np.sin(theta+alpha/2)*R+center[1])
    center2 = (np.cos(theta+alpha/2-2*np.pi/n)*R +
                center[0], np.sin(theta+alpha/2-2*np.pi/n)*R+center[1])

```



```

        arc_degree_inverse(center2, r, np.pi/2+theta,
                           np.pi/2+theta+theta_petal, colorfield)
    elif r > a:
        arc_degree(center1, r, np.pi/2+alpha-beta+theta,
                   np.pi/2+alpha-beta+theta+theta_petal, color)
        arc_degree_inverse(center1, r, np.pi/2+alpha-beta+theta,
                           np.pi/2+alpha-beta+theta+theta_petal, colorfield)
        arc_degree(center2, r, np.pi/2-beta+theta,
                   np.pi/2-beta+theta+theta_petal, color)
        arc_degree_inverse(center2, r, np.pi/2-beta+theta,
                           np.pi/2-beta+theta+theta_petal, colorfield)
    elif r < a:
        print("r=", r, ",a=", a)
        print('r<a,不能形成花瓣。')
    # plt.plot(center1[0], center1[1], marker='o', color='r')
    # plt.plot(center2[0], center2[1], marker='o', color='b')

```

一朵向上花瓣

```

def one_mandala_petal(center, R, r, n, theta=0, color='b'):
    n_mandala_petal(center, R, r, n, theta+np.pi/2, color)

```

一朵向上花弧

```

def one_mandala_arc(center, R, r, n, theta=0, color='b'):
    n_mandala_arc(center, R, r, n, theta+np.pi/2, color)

```

一朵向上花瓣场

```

def one_mandala_arc_with_field(center, R, r, n, theta=0, color='b',
                                colorfield='#ff0'):
    n_mandala_arc_with_field(center, R, r, n, theta +
                              np.pi/2, color, colorfield)

```

花瓣形成的单层花

```

def mandala_flower_by_petal(center, R, r, N, n, theta, color='b'):
    for i in range(0, N):
        one_mandala_petal(center, R, r, n, 2*i*np.pi/N+theta, color)

```

花弧形成的单层花

```

def mandala_flower_by_arc(center, R, r, N, n, theta, color='b'):
    for i in range(0, N):
        one_mandala_arc(center, R, r, n, 2*i*np.pi/N+theta, color)

```

单层花带场

```

def mandala_flower_by_arc_with_field(center, R, r, N, n, theta, color='b',

```

```

colorfield='#ff0'):
    for i in range(0, N):
        one_mandala_arc_with_field(center, R, r, n, 2*i*np.pi/N+theta, color)

'''
算法函数测试
'''

# 一朵向上花瓣测试:
# one_mandala_petal((0, 0), 1, 0.5, 6, 0, '#00f')
# 一朵向上花瓣场测试:
# one_mandala_arc_with_field((0, 0), 1, 0.6, 6, 0, 'b', '#ff0')
# 花瓣形成的单层花测试:
# mandala_flower_by_petal((0, 0), 1, 0.6, 6, 6, 0, '#0f0')
# mandala_flower_by_petal((0, 0), 1, 0.8, 4, 4, 0, '#00f')
# 单层花带场测试:
# mandala_flower_by_arc_with_field((0, 0), 1, 0.8, 8, 8, 0, 'b', '#ff0')
'''

以下是中文写好方便大改参数DIY的代码,
两个函数区别是下面那个函数多了花场颜色参数。
可以多写几个函数一起调用,相当于在画板上多次绘画,方便DIY。
'''

# 花瓣形成的单层花
中心 = (0, 0)
大半径 = 1
小半径 = 0.5
单层花瓣数 = 8
花瓣多边形数 = 8 # (花瓣的固有特性)
花瓣旋转角度 = 0
花瓣颜色 = 'red'
花场颜色 = 'gold'
mandala_flower_by_petal(中心, 大半径, 小半径, 单层花瓣数, 花瓣多边形数, 花瓣旋转角度, 花瓣颜色)
mandala_flower_by_arc_with_field(
    中心, 大半径, 小半径, 单层花瓣数, 花瓣多边形数, 花瓣旋转角度, 花瓣颜色, 花场颜色)
plt.axis('equal')
plt.axis("off")
plt.show()

```

结束语

好了花了一早上终于大功告成,后面我有时间有精力会继续分析睡莲和荷花螺旋以及马蹄莲的数学几何及算法。哎,用电脑真是累心血耗损严重,奈何自己是这个专业的人o(π_π)o收拾收拾吃饭休息了,下午单位团委说县上安排的相亲还得好好对待...

完成于2024年5月20日, , 小满, 农历4月13,(1314520,天意真是巧啊)

Written by lbylzk on Windows11 with VSCode+Python+Markdown on hardware HUAWEI MateBook E 2022.

Completed in QuXian county SiChuan province.