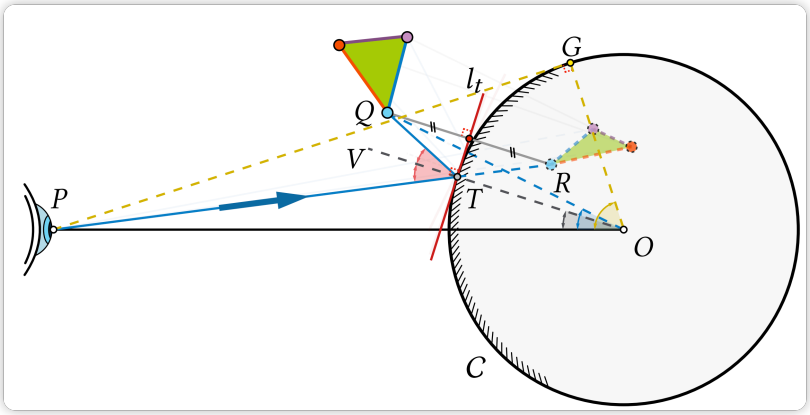
第一次上机实验报告

PB18061443 江昊霖

2023 年 3 月 24 日

1 实验目的



求解圆柱形镜面反射中像的位置：

在三维空间中，给定观察者位置 P ，圆柱形镜面的位置 O 和半径 r（假设高度无穷长），一点 Q，求 Q 被观察到的像的位置 R。我们进一步注意到，由于镜面是圆柱形的，所以 R 和 Q 一定有相同的高度。因此我们将该问题转化到二维平面上

假设、输入和输出如下：

- 假设：在 xy 平面上有一个镜面圆，可以反射光线，圆的圆心在原点处，半径为 1；观察点在 x 负半轴上；物点 Q 在第二象限且在圆外。
- 输入：
  - 观察点 P ， $P \in \{(x, y) \mid x < -1, y < 0\}$
  - 物点 Q ， $Q \in \{(x, y) \mid x < 0, y > 0, x^2 + y^2 > 1\}$
- 输出：反射点 T 的位置，像点 R 的位置，都用二维坐标 (x, y) 表示。

程序要求：可以改变输入参数并测试，输入参数包括三个：x_P, x_Q, y_Q 。比如，使用 `main` 函数的参数 `argv` 输入参数。又或者程序可以读取文件，文件的每一行是一组输入参数。

2 算法

由于 T 是反射点，有 $\angle PTV = \angle VTQ$。使用二分查找算法 [1] 找寻 $\theta \in (\pi/2, \pi)$ 满足

$$|\angle PTV - \angle VTQ| < \varepsilon.$$

则反射点 T 为

$$T = (\cos\theta, \sin\theta)$$

像点 R 为点 Q 对直线 l_t 的对称点。l_t 的方程为

$$\cos\theta x + \sin\theta y = 1$$

R 的坐标为

$$R = \big(x_Q - 2\cos\theta(T^T \cdot Q - 1), y_Q - 2\cos\theta(T^T \cdot Q - 1)\big)$$

```python
class Solution():

    def __init__(self, x_p, x_q, y_q) -> None:
        self.p = np.array([x_p, 0])
        self.q = np.array([x_q, y_q])

    def Angle(self, theta):
        t = np.array([math.cos(theta), math.sin(theta)])
        tp = np.array([self.p[0] - t[0], self.p[1] - t[1]])
        tq = np.array([self.q[0] - t[0], self.q[1] - t[1]])
        ptv = math.acos(
            np.dot(tp, t) / (np.linalg.norm(tp) * np.linalg.norm(t)))
        qtv = math.acos(
            np.dot(tq, t) / (np.linalg.norm(tq) * np.linalg.norm(t)))

        return ptv, qtv

    def binary_search(self):
        theta1 = np.arccos(0)
        theta2 = np.arccos(-1)
        theta3 = 0

        while True:
            theta3 = (theta1 + theta2)/2
            angle1, angle2 = self.Angle(theta3)

            if abs(theta1 - theta2) < epsilon:
                break
            if angle1 < angle2:
                theta2 = theta3
            else:
                theta1 = theta3

```

```
        t = np.array([math.cos(theta1), math.sin(theta1)])
        r = np.array([self.q[0] - 2 * t[0] * (np.dot(t, self.q) - 1),
                      self.q[1] - 2 * t[1] * (np.dot(t, self.q) - 1)])
        return t, r
```

3 实验结果

表 1: 实验结果

P	Q	T	R
$(-2.0, 0)$	$(-1.0, 1.0)$	$(-8.856698e-01, 4.643155e-01)$	$(-3.800570e-01, 6.749927e-01)$
$(-10.0, 0)$	$(-2.0, 1.0)$	$(-9.593115e-01, 2.823498e-01)$	$(3.042142e-01, 3.218110e-01)$
$(-1.000001, 0)$	$(-2.0, 2.0)$	$(-1.000000e+00, 1.999983e-06)$	$(7.999918e-06, 1.999996e+00)$
$(-2.0, 0)$	$(-1.0, 1e-06)$	$(6.123234e-17, 1.000000e+00)$	$(-1.000000e+00, 1.999999e+00)$
$(-2.33, 0)$	$(-3.0, 1.0)$	$(-9.892790e-01, 1.460376e-01)$	$(1.182424e+00, 3.825896e-01)$
$(-3.0, 0)$	$(-1.0, 0.5)$	$(6.123234e-17, 1.000000e+00)$	$(-1.000000e+00, 1.500000e+00)$
$(-3.0, 0)$	$(-2.0, 10.0)$	$(-8.270283e-01, 5.621602e-01)$	$(8.380296e+00, 2.944148e+00)$
$(-3.0, 0)$	$(-3.0, 1.0)$	$(-9.874085e-01, 1.581915e-01)$	$(1.187435e+00, 3.291362e-01)$
$(-10.0, 0)$	$(-2.0, 1.0)$	$(-9.593115e-01, 2.823498e-01)$	$(3.042142e-01, 3.218110e-01)$
$(-1024.0, 0)$	$(-8.0, 4.0)$	$(-9.700657e-01, 2.428425e-01)$	$(7.000894e+00, 2.447346e-01)$

4 实验结果和分析

使用 python 作为数值计算工具需要考虑到各种数据类型的精度，还有计算机进行除法运算时也会发生误差。例如使用 `np.arccos(0)` 相较于 `math.pi/2` 有更高的精度。

本次实验实现的二分搜索算法在 Q 靠近 x 轴时会发生点 T 的 x 坐标变为正号的情况。

参考文献

[1] K. Wu, R. Chen, X.-M. Fu, and L. Liu, “Computational mirror cup and saucer art,” *ACM Transactions on Graphics (TOG)*, vol. 41, no. 5, pp. 1–15, 2022.