10.6.4

EE25BTECH11049-Sai Krishna Bakki

October 4, 2025

Question

Draw a circle of radius 5cm. From a point 8cm away from its centre, construct a pair of tangents to the circle.

Theoretical Solution

Let's take center as origin $\mathbf{O}$ and a point 8cm aways from its center as $\mathbf{h} = \begin{pmatrix} 8 \\ 0 \end{pmatrix}$.

The equation of a circle is given by

$$g\left(\mathbf{x}\right) = \left\|\mathbf{x}\right\|^2 + 2\mathbf{u}^T\mathbf{x} + f = 0 \tag{1}$$

for

$$\mathbf{center} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \text{since } \mathbf{c} = -\mathbf{u} \tag{2}$$

we get

$$\mathbf{u} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{3}$$

we also know for any circle

$$\mathbf{V} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{4}$$

radius(r) $= 5$cm, we know that $r^2 = \|u\|^2 - f$ which gives us

$$f = -25 \tag{5}$$

By using below equation, we can determine the direction vectors of the tangent lines from an external point

$$\mathbf{m}^T \left[\left(\mathbf{Vh} + \mathbf{u} \right) \left(\mathbf{Vh} + \mathbf{u} \right)^T - \mathbf{V} g \left(\mathbf{h} \right) \right] \mathbf{m} = 0 \tag{6}$$

$$g\left(h\right) = 39 \tag{7}$$

where $\mathbf{m} = \begin{pmatrix} m_x \\ m_y \end{pmatrix}$ is the direction vectors of a tangent line.

Substituting values in (**??**), we get

$$\begin{pmatrix} m_x & m_y \end{pmatrix} \begin{pmatrix} 25 & 0 \\ 0 & -39 \end{pmatrix} \begin{pmatrix} m_x \\ m_y \end{pmatrix} = 0 \tag{8}$$

$$25 m_x^2 - 39 m_y^2 = 0 \tag{9}$$

$$\tag{10}$$

The slopes of the tangent line is given by $k = \frac{d_x}{d_x}$. we solve for the slopes:

$$k^2 = \frac{25}{39} \implies k = \pm\frac{5}{\sqrt{39}} \tag{11}$$

Now, normal vectors of tangent lines are

$$\mathbf{n_1} = \begin{pmatrix} 5 \\ \sqrt{39} \end{pmatrix}, \mathbf{n_2} = \begin{pmatrix} 5 \\ -\sqrt{39} \end{pmatrix} \tag{12}$$

Equations of tangent lines which passes through a point 8cm away from the center are

$$\mathbf{n_1^T x} = c, \mathbf{n_2^T x} = c \tag{13}$$

substituting **h** in line equation to get c,we get

$$c = 40 \tag{14}$$

$$\begin{pmatrix} 5 & \sqrt{39} \end{pmatrix} \mathbf{x} = 40 \tag{15}$$

$$\begin{pmatrix} 5 & -\sqrt{39} \end{pmatrix} \mathbf{x} = 40 \tag{16}$$

Now, solve for points of contact, for that we use the following formulae

$$\mathbf{q_i} = \left(\pm r \left(\frac{\mathbf{n_i}}{\|\mathbf{n_i}\|} \right) - \mathbf{u} \right) \tag{17}$$

we get

$$\mathbf{q_1} = \begin{pmatrix} \frac{25}{8} \\ \frac{5\sqrt{39}}{8} \end{pmatrix}, \mathbf{q_2} = \begin{pmatrix} \frac{25}{8} \\ -\frac{5\sqrt{39}}{8} \end{pmatrix} \tag{18}$$

C Code

```c
#include <stdio.h>
#include <math.h>

    // Define a simple structure for a 2D point.
    struct Point {
        double x;
        double y;
    };

    // This function calculates the two tangent points (t1, t2)
        for a circle
    // centered at the origin with a given radius 'r', from an
        external point 'p'.
    // The results are returned via pointers.
    void calculate_tangents(double r, struct Point p, struct
        Point* t1, struct Point* t2) {
        // 1. Calculate the x-coordinate of the polar line.
        // This is where the line connecting the tangent points
            intersects the x-axis.
```

C Code

```
        double x_contact = (r * r) / p.x;

        // 2. Use the circle equation (x^2 + y^2 = r^2) to find
            the y-coordinates.
        double y_contact_sq = (r * r) - (x_contact * x_contact);

        // If y_contact_sq is negative, the point is inside the
            circle, and no tangents exist.
        if (y_contact_sq < 0) {
            // Set coordinates to NaN (Not a Number) to indicate
                an error.
            t1->x = t1->y = NAN;
            t2->x = t2->y = NAN;
            return;
        }
double y_contact = sqrt(y_contact_sq);
// 3. Assign the coordinates to the output structures.
        t1->x = x_contact;
        t1->y = y_contact;
```

C Code

```
        t2->x = x_contact;
        t2->y = -y_contact;
    }
// Main function to demonstrate the tangent calculation for the
    specified problem.
    int main() {
        // Problem parameters: Circle radius 5, external point at
            (8, 0).
        double radius = 5.0;
        struct Point external_point = {8.0, 0.0};

        // Structures to hold the results.
        struct Point tangent_point1;
        struct Point tangent_point2;

        // Call the function to perform the calculation.
        calculate_tangents(radius, external_point, &
            tangent_point1, &tangent_point2);
```

C Code

```
    // Print the results.
    printf(Problem: Find tangents to a circle (radius=%.1f)
        from point (%.1f, %.1f)\n,
        radius, external_point.x, external_point.y);
    printf(
        -------------------------------------------------------
        n);
    printf(Calculated Tangent Point 1 (T1): (%.4f, %.4f)\n,
        tangent_point1.x, tangent_point1.y);
    printf(Calculated Tangent Point 2 (T2): (%.4f, %.4f)\n,
        tangent_point2.x, tangent_point2.y);

    return 0;
}
```

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

# --- C LIBRARY INTEGRATION ---

# Define a Python class that mirrors the C Point struct.
class Point(ctypes.Structure):
    _fields_ = [(x, ctypes.c_double),
                (y, ctypes.c_double)]

# Load the shared C library.
# This assumes the compiled library is in the same directory.
lib_path = './circs.so'
if not os.path.exists(lib_path):
    print(Error: Shared library 'circs.so' not found.)
    print(Please compile the C code first with: gcc -shared -o
        libtangent.so -fPIC tangent_calc.c)
```

Python Through Shared Output Code

```python
else:
    tangent_lib = ctypes.CDLL(lib_path)

    # Define the function signature for the tangent calculation
        function.
    calculate_tangents_c = tangent_lib.calculate_tangents
    calculate_tangents_c.argtypes = [ctypes.c_double, Point,
        ctypes.POINTER(Point), ctypes.POINTER(Point)]
    calculate_tangents_c.restype = None

    # --- PROBLEM SETUP AND C FUNCTION CALL ---

    # 1. Define the problem: circle of radius 5, point 8cm away.
    circle_radius = 5.0
    external_point_p = Point(8.0, 0.0)

    # Create empty Point structures to hold the results from the
        C function.
```

Python Through Shared Output Code

```
    tangent_point_t1 = Point()
    tangent_point_t2 = Point()

    # 2. Call the C function to perform the calculation.
    calculate_tangents_c(
        circle_radius,
        external_point_p,
        ctypes.byref(tangent_point_t1),
        ctypes.byref(tangent_point_t2)
    )

    # 3. Print the results calculated by the C code.
    print(fResults from C function:)
    print(fTangent Point 1 (T1): ({tangent_point_t1.x:.4f}, {
        tangent_point_t1.y:.4f}))
    print(fTangent Point 2 (T2): ({tangent_point_t2.x:.4f}, {
        tangent_point_t2.y:.4f}))
# --- PLOTTING ---
# Helper function to generate circle points.
```

Python Through Shared Output Code

```
def circ_gen(center, r):
    theta = np.linspace(0, 2 * np.pi, 100)
    x = center[0] + r * np.cos(theta)
    y = center[1] + r * np.sin(theta)
    return x, y

# Generate circle for plotting.
x_circ, y_circ = circ_gen([0, 0], circle_radius)

# Plot the circle.
plt.plot(x_circ, y_circ, label='Circle: $x^2 + y^2 = 25$')
# Plot the key points.
plt.scatter([0], [0], color='black', label='Center O(0,0)')
plt.scatter([external_point_p.x], [external_point_p.y], color
    ='red', label=f'External Point P({external_point_p.x:.0f
    }, {external_point_p.y:.0f})')
plt.scatter([tangent_point_t1.x, tangent_point_t2.x], [
    tangent_point_t1.y, tangent_point_t2.y], color='green',
    label='Tangent Points')
```

Python Through Shared Output Code

```
# Plot the tangent lines.
plt.plot([external_point_p.x, tangent_point_t1.x], [
    external_point_p.y, tangent_point_t1.y], 'r--')
plt.plot([external_point_p.x, tangent_point_t2.x], [
    external_point_p.y, tangent_point_t2.y], 'r--', label='
    Tangents')
plt.text(tangent_point_t1.x + 0.5, tangent_point_t1.y, f'T1
    ({tangent_point_t1.x:.2f}, {tangent_point_t1.y:.2f})')
plt.text(tangent_point_t2.x + 0.5, tangent_point_t2.y - 0.5,
    f'T2 ({tangent_point_t2.x:.2f}, {tangent_point_t2.y:.2f})
    ')
plt.title('Construction of Tangents to a Circle (C + Python)'
    )
plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.gca().set_aspect('equal', adjustable='box')
plt.grid(True)
plt.legend()
plt.show()
```

Python Code

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

# Helper function to generate points for a line segment
def line_gen(A, B):
    Generates points for a line segment between A and B.
    # Extend the line for better visualization by using a wider range
    len = 20
    x_AB = np.zeros((2, len))
    lam_1 = np.linspace(-0.5, 1.5, len) # Use a wider range to draw a longer line
    for i in range(len):
        temp1 = A + lam_1[i] * (B - A)
        x_AB[:, i] = temp1.flatten()
    return x_AB

# Helper function to generate points for a circle
def circ_gen(O, r):
```

Python Code

```
    Generates points for a circle with center O and radius r.
    len = 100
    theta = np.linspace(0, 2 * np.pi, len)
    x_circ = np.zeros((2, len))
    x_circ[0, :] = r * np.cos(theta)
    x_circ[1, :] = r * np.sin(theta)
    x_circ = (x_circ.T + O.flatten()).T
    return x_circ

# 1. DEFINE CIRCLE AND EXTERNAL POINT
# Circle parameters
O = np.array([[0], [0]]) # Center at origin
r = 5 # Radius 5cm
# External point P (8cm from the center)
P = np.array([[8], [0]])

# 2. CALCULATE TANGENT POINTS
# The points of contact lie on the polar line x = r^2 / P_x
x_contact = r**2 / P[0, 0]
```

Python Code

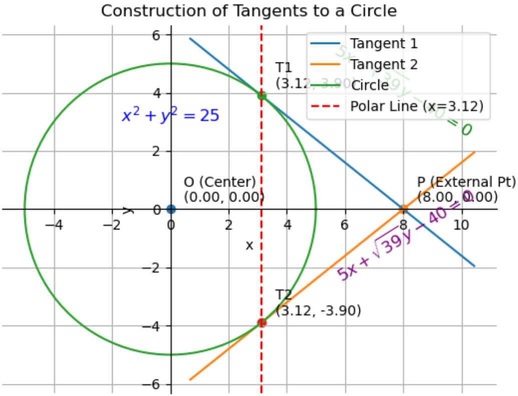
```
# Find the y-coordinates by substituting x into the circle
    equation x^2 + y^2 = r^2
y_contact_sq = r**2 - x_contact**2
y_contact = np.sqrt(y_contact_sq)

# The two points of contact
T1 = np.array([[x_contact], [y_contact]])
T2 = np.array([[x_contact], [-y_contact]])

# 3. GENERATE GEOMETRIES FOR PLOTTING
# Generate the circle
x_circ = circ_gen(O, r)
# Generate the two tangent lines
x_tangent1 = line_gen(P, T1)
x_tangent2 = line_gen(P, T2)

# 4. PLOTTING
# Plot the tangent lines and the circle
```

```python
plt.plot(x_tangent1[0, :], x_tangent1[1, :], label='Tangent 1')
plt.plot(x_tangent2[0, :], x_tangent2[1, :], label='Tangent 2')
plt.plot(x_circ[0, :], x_circ[1, :], label='Circle')

# Plot the polar line
plt.axvline(x=x_contact, color='r', linestyle='--', label=f'Polar Line (x={x_contact:.2f})')

# Plot and label the key points
points = {'O (Center)': O, 'P (External Pt)': P, 'T1': T1, 'T2': T2}
for label, point in points.items():
    plt.scatter(point[0], point[1])
    plt.annotate(f'{label}\n({point[0,0]:.2f}, {point[1,0]:.2f})',
                 (point[0,0], point[1,0]),
                 textcoords=offset points,
                 xytext=(10,5),
                 ha='left')
```

Python Code

```
plt.text(0, 3, r'$x^2 + y^2 = 25$', fontsize=12, color='blue', ha
    ='center')
plt.text(5.5, 2.5, r'$5x - \sqrt{39}y - 40 = 0$', fontsize=12,
    color='green', rotation=-32)
plt.text(5.5, -2.5, r'$5x + \sqrt{39}y - 40 = 0$', fontsize=12,
    color='purple', rotation=32)
# Set plot properties
ax = plt.gca()
ax.spines['left'].set_position('zero')
ax.spines['bottom'].set_position('zero')
ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
plt.legend()
plt.grid(True)
plt.axis('equal')
plt.title('Construction of Tangents to a Circle')
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

Figure:


Construction of Tangents to a Circle
Tangent 1
Tangent 2
Circle
Polar Line (x=3.12)
T1
(3.12, 3.90)
$x^2 + y^2 = 25$
O (Center)
(0.00, 0.00)
P (External Pt)
(8.00, 0.00)
$5x + \sqrt{39}y - 40 = 0$
T2
(3.12, -3.90)
x
y