1.5.1

Bhargav - EE25BTECH11013

August 22,2025

Question

The center of a circle whose endpoints of a diameter of the circle A, B are $(-6,3)$ and $(6,4)$ is

Theoretical Solution

Let the endpoints of the diameter of the circle be $\mathbf{A}$ and $\mathbf{B}$:

$$\mathbf{A} = \begin{pmatrix} -6 \\ 3 \end{pmatrix}, \mathbf{B} = \begin{pmatrix} 6 \\ 4 \end{pmatrix} \tag{1}$$

We can use the midpoint formula to find the center of the circle.

Equation

Midpoint C of the vectors A and B is given by

$$\mathbf{C} = \frac{\mathbf{A} + \mathbf{B}}{2} \tag{2}$$

Equation

$$\mathbf{C} = \frac{1}{2}\begin{pmatrix} -6+6 \\ 3+4 \end{pmatrix} \tag{3}$$

$$\mathbf{C} = \frac{1}{2}\begin{pmatrix} 0 \\ 7 \end{pmatrix} \tag{4}$$

$$\mathbf{C} = \begin{pmatrix} 0 \\ \frac{7}{2} \end{pmatrix}. \tag{5}$$

C Code - Midpoint formula

```c

#include <stdio.h>

void midpt(double x1, double y1, double x2, double y2, double* x, double* y) {
    *x = (x2 + x1) / (2);
    *y = (y2 + y1) / (2);
}
```

Python + C Code

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

# Load shared object
lib = ctypes.CDLL(./libmidpt.so)

# Define C function prototype
lib.midpt.argtypes = [ctypes.c_double, ctypes.c_double,
                      ctypes.c_double, ctypes.c_double,
                      ctypes.POINTER(ctypes.c_double), ctypes.
                          POINTER(ctypes.c_double)]

# Input diameter endpoints
x1, y1 = -6.0, 3.0
x2, y2 = 6.0, 4.0
```

Python + C Code

```
# Prepare output variables
x_mid = ctypes.c_double()
y_mid = ctypes.c_double()

# Call C function
lib.midpt(x1, y1, x2, y2, ctypes.byref(x_mid), ctypes.byref(y_mid))

cx, cy = x_mid.value, y_mid.value
print(Centre of circle:, (cx, cy))

# Radius = half distance between endpoints
r = np.sqrt((x2 - x1)**2 + (y2 - y1)**2) / 2

# Generate circle points
theta = np.linspace(0, 2*np.pi, 500)
x_circle = cx + r * np.cos(theta)
y_circle = cy + r * np.sin(theta)
```

Python + C Code

```
# Plot
plt.figure(figsize=(6,6))
plt.plot(x_circle, y_circle, label=Circle)
plt.scatter([x1, x2], [y1, y2], color=red, s=80, label=Diameter
    Endpoints)
plt.text(x1 - 1, y1 - 0.5, f({x1:.0f}, {y1:.0f}), color=red,
    fontsize=10)
plt.text(x2 + 0.5, y2, f({x2:.0f}, {y2:.0f}), color=red, fontsize
    =10)

plt.scatter(cx, cy, color=blue, marker=x, s=200, linewidths=3,
    label=Centre)
plt.text(cx + 0.5, cy + 0.5, f({cx:.2f}, {cy:.2f}), color=blue,
    fontsize=10)
plt.plot([x1, x2], [y1, y2], 'g--', label=Diameter)
```

Python + C Code

```python
plt.axis(equal)
plt.legend(loc=upper right)
plt.title(Circle from Diameter Endpoints)
plt.savefig(/Users/bhargavkrish/Documents/ee1030-2025/ee25btech11013/matgeo/1.5.1/figs/Figure_1.png)
plt.show()
```

Python Code

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

# Input diameter endpoints
x1, y1 = -6.0, 3.0
x2, y2 = 6.0, 4.0

# Midpoint (centre of circle)
cx = (x1 + x2) / 2
cy = (y1 + y2) / 2
print(Centre of circle:, (cx, cy))

# Radius = half distance between endpoints
r = np.sqrt((x2 - x1)**2 + (y2 - y1)**2) / 2
```

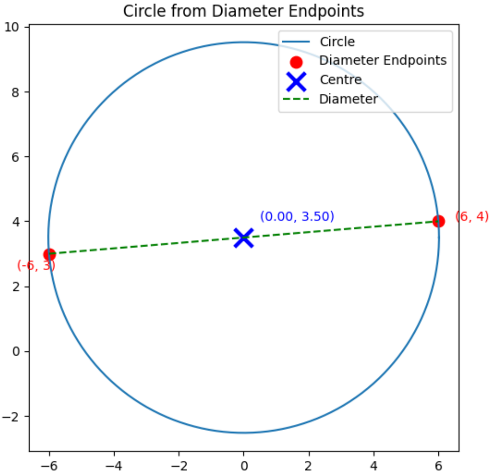
Python Code

```
# Generate circle points
theta = np.linspace(0, 2*np.pi, 500)
x_circle = cx + r * np.cos(theta)
y_circle = cy + r * np.sin(theta)

# Plot
plt.figure(figsize=(6,6))
plt.plot(x_circle, y_circle, label=Circle)

plt.scatter([x1, x2], [y1, y2], color=red, s=80, label=Diameter Endpoints)
plt.text(x1 - 1, y1 - 0.5, f({x1:.0f}, {y1:.0f}), color=red, fontsize=10)
plt.text(x2 + 0.5, y2, f({x2:.0f}, {y2:.0f}), color=red, fontsize=10)
```

Python Code

```
plt.scatter(cx, cy, color=blue, marker=x, s=200, linewidths=3,
    label=Centre)
plt.text(cx + 0.5, cy + 0.5, f({cx:.2f}, {cy:.2f}), color=blue,
    fontsize=10)

plt.plot([x1, x2], [y1, y2], 'g--', label=Diameter)
plt.axis(equal)
plt.legend(loc=upper right)
plt.title(Circle from Diameter Endpoints)
plt.savefig(/Users/bhargavkrish/Documents/ee1030-2025/
    ee25btech11013/matgeo/1.5.1/figs/Figure_1.png)
plt.show()
```

Plot


Circle from Diameter Endpoints
Circle
Diameter Endpoints
Centre
Diameter
(0.00, 3.50)
(6, 4)
(-6, 3)
10
8
6
4
2
0
−2
−6
−4
−2
0
2
4
6