Source Code: Enhanced ACC with Prediction

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

class EnhancedAdaptiveCruiseControl:
    def __init__(self, desired_distance=10, kp=0.6, ki=0.08, kd=0.25, k_predict=0.3):
        self.desired_distance = desired_distance
        self.kp = kp
        self.ki = ki
        self.kd = kd
        self.k_predict = k_predict  # Lead vehicle prediction gain
        self.integral = 0
        self.prev_error = 0

    def predict_lead_behavior(self, lead_acceleration):
        return self.k_predict * lead_acceleration

    def update(self, current_distance, lead_acceleration, dt):
        error = self.desired_distance - current_distance
        self.integral += error * dt
        derivative = (error - self.prev_error) / dt if dt > 0 else 0
        prediction = self.predict_lead_behavior(lead_acceleration)
        output = (self.kp * error + self.ki * self.integral + self.kd * derivative) + prediction
        self.prev_error = error
        return output

class VehicleSimulator:
    def __init__(self, init_speed=20):
        self.speed = init_speed
        self.position = 0

    def update(self, acceleration, dt):
        self.speed += acceleration * dt
        self.speed = max(self.speed, 0)
        self.position += self.speed * dt
        return self.position

# Simulation
dt = 0.1
acc = EnhancedAdaptiveCruiseControl()
ego = VehicleSimulator()
lead = VehicleSimulator(init_speed=20)

times, distances, speeds = [], [], []

for t in np.arange(0, 50, dt):
    lead_acc = 0.1 if t < 25 else -0.15
    lead.update(lead_acc, dt)
    distance = lead.position - ego.position
    control = acc.update(distance, lead_acc, dt)
    ego.update(control, dt)
```

```
    times.append(t)
    distances.append(distance)
    speeds.append(ego.speed)

# Plot results
plt.figure(figsize=(10, 6))
plt.subplot(2, 1, 1)
plt.plot(times, distances, label="Distance to Lead")
plt.axhline(y=acc.desired_distance, color='r', linestyle='--', label="Desired Distance")
plt.title("Adaptive Cruise Control Enhancement - Distance Tracking")
plt.legend()

plt.subplot(2, 1, 2)
plt.plot(times, speeds, label="Ego Vehicle Speed", color='g')
plt.title("Ego Vehicle Speed Profile")
plt.xlabel("Time (s)")
plt.tight_layout()
plt.show()
```