```python
from rclpy.node import Node
from std_msgs.msg import Float32
import lgpio
import time
import threading


class MotorPWMNode(Node):
    def __init__(self):
        super().__init__('motor_pwm_node')

        # ConfiguraciГіn GPIO
        self.CHIP = 4
        self.PWM_PIN = 12  # BCM18 (pin fГsico 12)
        self.freq = 1000   # Hz
        self.period = 1 / self.freq
        self.duty = 0.0    # duty [0.0 - 1.0]

        # Inicializar GPIO
        self.h = lgpio.gpiochip_open(self.CHIP)
        lgpio.gpio_claim_output(self.h, self.PWM_PIN)

        # Suscriptor
        self.subscription = self.create_subscription(
            Float32,
            '/motor_vel',
            self.vel_callback,
            10
        )

        # Hilo para generar PWM
        self.running = True
        self.pwm_thread = threading.Thread(target=self.pwm_loop)
        self.pwm_thread.start()

        self.get_logger().info("MotorPWMNode iniciado. Escuchando en /motor_vel (0-100).")

    def vel_callback(self, msg):
        # Escala 0-100 a 0.0-1.0
        self.duty = max(0.0, min(1.0, msg.data / 100.0))
        self.get_logger().info(f"Duty actualizado: {self.duty*100:.1f}%")

    def pwm_loop(self):
        while self.running:
            if self.duty > 0:
                lgpio.gpio_write(self.h, self.PWM_PIN, 1)
                time.sleep(self.period * self.duty)
                lgpio.gpio_write(self.h, self.PWM_PIN, 0)
                time.sleep(self.period * (1 - self.duty))
            else:
                lgpio.gpio_write(self.h, self.PWM_PIN, 0)
                time.sleep(self.period)

    def destroy_node(self):
        self.running = False
        self.pwm_thread.join()
        lgpio.gpio_write(self.h, self.PWM_PIN, 0)
        lgpio.gpiochip_close(self.h)
        super().destroy_node()


def main(args=None):
    rclpy.init(args=args)
    node = MotorPWMNode()
    try:
        rclpy.spin(node)
    except KeyboardInterrupt:
        pass
    finally:
        node.destroy_node()
        rclpy.shutdown()
```