Tarea: Control de Navegación del Robot mediante una Interfaz Gráfica

Objetivos:

1. Comprender y modificar interfaces gráficas en PyQt5.

2. Interactuar con ROS (Robot Operating System) para controlar la navegación del robot.

3. Implementar funcionalidades adicionales para mejorar el control y la automatización del robot.

Código Base:

```
from PyQt5 import QtWidgets, uic
import sys
import rospy
from move_base_msgs.msg import MoveBaseAction, MoveBaseGoal
import actionlib
from actionlib_msgs.msg import *
from geometry_msgs.msg import Pose, Point, Quaternion
from go_to_specific_point_on_map import GoToPose

class Ui(QtWidgets.QMainWindow):
    def __init__(self):
        super(Ui, self).__init__()
        uic.loadUi('sanfer.ui', self)

        # Encontrar el boton con el nombre "-------"
        self.button1 = self.findChild(QtWidgets.QPushButton, 'pushButton')
        self.button2 = self.findChild(QtWidgets.QPushButton, 'pushButton_2')
        self.button3 = self.findChild(QtWidgets.QPushButton, 'pushButton_3')

        self.show()

def go_to_position(position, quaternion):
    try:
        rospy.init_node('nav_test', anonymous=False)
        navigator = GoToPose()

        # Registrar la posicion y orientacion
        rospy.loginfo("Ir a la posicion (%s, %s) con la orientacion (%s, %s, %s, %s)",
                      position['x'], position['y'],
                      quaternion['r1'], quaternion['r2'], quaternion['r3'], quaternion['r4'])

        success = navigator.goto(position, quaternion)

        if success:
```

```
            rospy.loginfo("Alcanzada la ubicacion deseada")
        else:
            rospy.loginfo("No se ha podido alcanzar la ubicacion deseada")

        # Sleep para dar tiempo al ultimo mensaje de registro a ser enviado
        rospy.sleep(1)

    except rospy.ROSInterruptException:
        rospy.loginfo("Ctrl-C detectado. Cerrando")

app = QtWidgets.QApplication(sys.argv)
window = Ui()

# Enlazar botones con sus ubicaciones
window.button1.clicked.connect(lambda: go_to_position(
    {'x': 1.58, 'y': 0.41},
    {'r1': 0.000, 'r2': 0.000, 'r3': 0.000, 'r4': 1.000}  # Orientation for button1
))

window.button2.clicked.connect(lambda: go_to_position(
    {'x': 5.15, 'y': 0.38},
    {'r1': 0.000, 'r2': 0.000, 'r3': 0.707, 'r4': 0.707}  # Orientation for button2
))

window.button3.clicked.connect(lambda: go_to_position(
    {'x': 0, 'y': 0},
    {'r1': 0.000, 'r2': 0.000, 'r3': -0.707, 'r4': 0.707}  # Orientation for button3
))

window.show()
app.exec_()
```

PASOS:

Parte 1: Añadir Más Puntos de Navegación

Pregunta:
¿Cómo añadir uno o dos lugares más para que el robot pueda navegar a ellos mediante nuevos botones en la interfaz gráfica?

Respuesta:

Para añadir más lugares, debes seguir estos pasos:

1. Modificar el Archivo de Interfaz Gráfica (sanfer.ui):

Abre sanfer.ui en el editor de Qt Designer.

Añade uno o dos nuevos botones (QPushButton) a la interfaz.

Asigna nombres únicos a estos botones, por ejemplo, pushButton_4 y pushButton_5.

2. Actualizar el Código Python:

Encontrar los Nuevos Botones:

```
self.button4 = self.findChild(QtWidgets.QPushButton, 'pushButton_4')
self.button5 = self.findChild(QtWidgets.QPushButton, 'pushButton_5')
```

Conectar los Nuevos Botones con Funciones de Navegación:

```
window.button4.clicked.connect(lambda: go_to_position(
{'x': 2.5, 'y': 3.0},
{'r1': 0.000, 'r2': 0.000, 'r3': 0.382, 'r4': 0.924} # Orientación para botón4
))

window.button5.clicked.connect(lambda: go_to_position(
{'x': -1.0, 'y': 4.2},
{'r1': 0.000, 'r2': 0.000, 'r3': 0.707, 'r4': 0.707} # Orientación para botón5
))
```

Nota: Asegúrate de definir las coordenadas (x, y) y las orientaciones (r1, r2, r3, r4) según los puntos específicos en el mapa que estás utilizando.