▲ Subscripciones- “funciones” que van en otro lado, al principio

▲ Crear método de “servicio”

◄ Contante que hay que reordenar

▼ Llama de método ya creadas

▲ Revisar si no hay que asignarlo también en otro lugar

◄ Rehacer-Revisar

Tipo de mensaje:

Punto_web:

string[] orden

float64[] coordenadas

Página web js

M conectar localhost

(los 3 metodos siguientes se hacen iguales,tanto para subscriptor como para publicador, solo varia después según el método que se use al llamarlo, ” .publish() “ o “.subscribe() ”

M Definir Publicador de: CAMBIAR TODOS LOS VAR POR CONSTS

pub, cord_ros, std_msgs/Float64MultiArray

pub_orden, orden_web, niryo_controladores/punto_web

M Definir Suscriptor de: CAMBIAR TODOS LOS VAR POR CONSTS

Punto_ros, Puntodb, niryo_controladores/punto_web

Informe, Informe_web, std_msgs/String

Posicion_real, Pos_dy, std_msgs/Int16MultiArray

Punto_tomado, P_dy, std_msgs/Int16MultiArray

M Definir Suscriptor y publicador de: CAMBIAR TODOS LOS VAR POR CONSTS

Modo_actual, modoActuarTopic, std_msgs/Bool

Tipo_modo_lectura, tipoModoLectura, std_msgs/Bool

Ver diferencia entre “ var “ y “const”

var modo_funcionamiento=1; ◄

F mostrarPuntos()

Obtengo elemento donde se muestran los puntos (puntos-lista)

Elimino los puntos del elemento (.innerHTML)

Creo msg tipo punto_web para mandar

orden: ["subir_db"],

coordenadas: []

Publico en “pub_orden, orden_web ”

Informo en consola o cuadro de mensajes

F mostrarInformacion(puntos) 93

Creo msg tipo punto_web para mandar

orden: ['ver', punto],

coordenadas: []

Publico en “pub_orden, orden_web ”

Informo en consola o cuadro de mensajes

Desactivo inputs de ángulos (para que no se puedan cambiar valores) ▲

Desactivo inputs de cartesianos (para que no se puedan cambiar valores) ▲

F nuevo_punto_cart()

Habilita los inputs cartesianos ▲

Habilita el input del nombre del punto ▲

Vacía input del nombre del punto ▼

Pone a “0” lo inputs de los angulos ▲

Informo en consola o cuadro de mensajes

F nuevo_punto_ang()

Habilita los inputs de ángulos ▲

Habilita el input del nombre del punto ▲

Vacía input del nombre del punto ▼

Habilitar inputs de velocidad y aceleración ▲▲

Informo en consola o cuadro de mensajes

F agregarPunto()

Obtener valor del input del nombre del punto ▲

Obtener valor de los inputs de ángulos ▲

Verificar si el nombre se conoce (si es asi informar y salir de la función F)

Creo msg tipo punto_web para mandar

orden: ['agregar', nombre],

coordenadas: [coordenadas angulares obtenidas]

Publico en “pub_orden, orden_web ”

Correr ” F mostrarPuntos() “

F enviarCoordenadasCart()

Obtener valor de los inputs cartesianos▲

Creo msg tipo std_msgs/Float64MultiArray para mandar

data: [coordenadas cartesianas obtenidas]

Publico en “ pub, cord_ros “

Informo en consola o cuadro de mensajes

F enviarCoordenadasAng()

Obtener valor de los inputs angulares ▼

Creo msg tipo std_msgs/Float64MultiArray para mandar

data: [coordenadas angulares obtenidas]

Publico en “ pub, cord_ros “

Informo en consola o cuadro de mensajes

F eliminarPunto()

Obtener valor del input del nombre del punto ▼

Creo msg tipo punto_web para mandar

orden: ['eliminar', nombre],

coordenadas: [ ]

Publicar en “pub_orden, orden_web ”

Correr ” F mostrarPuntos() “

F modificarPunto() 267 ◄

F modificarInformacion() 328 ◄

F correr() 322 ◄

F guardar() 336 ◄

F que corre al recibir msg y Subscripción de “Informe, Informe_web ” ▲

Definir elemento con el “cuadro de información”

Cambiar mensaje mostrado en el cuado por lo resibido en el mensaje

Informo en consola

F que corre al recibir msg y Subscripción de “Punto_ros, Puntodb “ ▲

Definir elemento donde se muestran los puntos (puntos-lista)

SI el componente “coordenadas” del msj esta vacio

| Muestra los nombres de los puntos que se encuentra en el componente “orden” del msj 359 a 364 ◄

SINO

SI “ modo_funcionamiento ” === 0

| Rellena los inputs de los ángulos con los valores del componente “coordenadas” del msj

SINO

Rellena los inputs de los cartesianos con los valores del componente “coordenadas” del msj

Informo en consola o cuadro de mensajes

////////// Inicio de botones PTP,LIN,CIRC //////////

let id_plan = ""; ◄ agregar esto en codigo

F actualizarBotonSeleccionado(nuevoIdPlan) /sirve para cambiar los colores de los botones “PTP-LIN-CIRC” /

Guardar el valor recibido en “ id_plan “ /El valor resibido es un string con PTP,LINK o CIRC /

SI “ modo_funcionamiento ” === 0

| Cambiar fondo y texto de los 3 botones (acorde al modo) ▲

SINO

Cambiar fondo de los 3 botones (acorde al modo) ▲

Cambiar texto y fondo del botón seleccionado (según " id_plan ")

F PTP()

Correr " F actualizarBotonSeleccionado("PTP") "

Correr " F ocultarRad() "

F LIN()

Correr " F actualizarBotonSeleccionado("LIN") "

Correr " F ocultarRad() "

F CIRC()

Correr " F actualizarBotonSeleccionado("CIRC") "

Correr " F mostrarRad () "

F que corre cuando se abre carga la pagina (pone el botón "PTP" como seleccionado) ▲

Correr " F actualizarBotonSeleccionado("PTP") "

F mostrarRad ()

Hace visual el input y el laber " rad "

F ocultarRad()

Oculta visual el input y el laber " rad "

////////// Fin de botones PTP,LIN,CIRC //////////

let rutina_actual = []; ◄

let trayectoria_actual= []; ◄

////////// Inicio de agregar trayectoria //////////

F agregarInformacion() 481 (crea fila en el cuadro de rutina)

Obtener valores de imputs de “velocidad” y “aceleración”

Crear variables para almacenar “nombre” , “coordenadas” , “nuevo punto”,

Asignar a variables par la tabla de la rutina “serie “ (489), para la nueva fila (490) y posición de las finas (492)

SI “ modo_funcionamiento ” === 0

| aumentar contador “ contadorRutina ”

| crear nombre temporal (rut” contadorRutina “ (rut1-rut2,…)

| asigno este nombre temporal a la variable “ nombre ”

| guardo los valores de los inputs angulares a la variable “ coordenadas ”

| asigno los valores a “ nuevo punto “ (“ nombre ”, “ coordenadas ”, “ velocidad “, “ aceleración “, “ id_plan “)

SINO

Guardo en la variable “ nombre ” el valor del input nombre

Switch en base a “ id_plan “

SI “ id_plan “ = ‘PTP’

guardo los valores de los inputs angulares a la variable “ coordenadas ”

Asigno los valores a “ nuevo punto “ (“ nombre ”, …….,)

Salgo del switch

SI “ id_plan “ = ‘LIN’

guardo los valores de los inputs cartesianos a la variable “ coordenadas ”

Asigno los valores a “ nuevo punto “ (“ nombre ”, … , … , … , …)

Salgo del switch

SI “ id_plan “ = ‘CIRC’

Guardo los valores de los inputs cartesianos a la variable “ coordenadas ”

Obtener valores de imputs de “ rad “

Asigno los valores a “ nuevo punto “ (“ nombre ”, … , … , … ,” rad “x …)

Salgo del switch

SINO coincide

Mostrar mensaje de error

Salgo del switch

Colocar valores en la “nueva fila” (posición,nombre,coordenadas,velocida,aceleración,rad,id_plan)

F que se usa al cliquear en la fila creada (muestra los valores en los inputs)

Coloca en el input nombre el respectivo valor de la fila seleccionada

SI “ modo_funcionamiento ” === 0

| Guardo en una constante los valores de las coordenadas de la fila

| Coloco estos valores en los inpus angulares

SINO

Correr ” F mostrarInformacion (nombre del punto que esta en la fila) “

Asignar los valores de la fila a los inputs “ velocidad”, y “aceleración”

Inhabilitar inputs “ velocidad”, y “aceleración”

SI el tipo de trayectoria es CIRC

| Agregar el valor de “ rad “ de la fila al input correspondiente

| Correr “ F CIRC()”

SINO, SI el tipo de trayectoria es LIN

| Correr " F LIN "

SINO, SI el tipo de trayectoria es PTP

| Correr " F PTP "

SINO

Mostrar mensaje de error

trayectoria_actual = [fila.rowIndex - 1];

Informo en consola o cuadro de mensajes

F agregar_Punto() 647 (analizar si es importante, ver función anterior)

Obtener valores de imputs de "velocidad" y "aceleración"

Crear variables para almacenar "nombre" , "coordenadas" , "nuevo punto",

Asignar a variables par la tabla de la rutina "serie " (489), para la nueva fila (490) y posición de las finas (492)

Aumentar contador " contadorRutina "

Crear nombre temporal (rut" contadorRutina " (rut1-rut2,…)

Asignar este nombre temporal a la variable " nombre "

Guardar los valores de los inputs angulares a la variable " coordenadas "

Asignar los valores a " nuevo punto " (" nombre ", " coordenadas ", " velocidad ", " aceleración ", " id_plan ")

Colocar valores en la "nueva fila" (posición, nombre, coordenadas, velocida, aceleración, rad,id_plan)

F que se usa al cliquear en la fila creada (muestra los valores en los inputs)

Coloca en el input nombre el respectivo valor de la fila seleccionada

SI " modo_funcionamiento " === 0

| Guardo en una constante los valores de las coordenadas de la fila

| Coloco estos valores en los inpus angulares

SINO

Correr ” F mostrarInformacion (nombre del punto que esta en la fila) “

Asignar los valores de la fila a los inputs “ velocidad”, y “aceleración”

Inhabilitar inputs “ velocidad”, y “aceleración”

SI el tipo de trayectoria es CIRC

| Agregar el valor de “ rad “ de la fila al input correspondiente

| Correr “ F CIRC()”

SINO, SI el tipo de trayectoria es LIN

| Correr “ F LIN “

SINO, SI el tipo de trayectoria es PTP

| Correr “ F PTP “

SINO

Mostrar mensaje de error

trayectoria_actual = [fila.rowIndex - 1]; ▲

Informo en consola o cuadro de mensajes

F eliminarInformacion()

Verifica si hay una fila seleccionada. Si no hay, termina la función y muestra un mensaje en consola (796 a 799)

Coloco en una variable el índice de la fila seleccionada

Coloco en una variable la tabla de la rutina “ serie “

ELIMINO la fila seleccionada (usando el índice)

Eliminar el valor correspondiente en " rutina_actual "

Vacío la variable “ trayectoria_actual “

CORRER “ F actualizarPosiciones() “

F actualizarPosiciones()

Colocar en una variable la tabla de la rutina “ serie “

Colocar en una variable la las filas de la tabla

FOR para modificar en cada fila su posición en la tabla mostrada “ serie “ y el “data-pos” de cada uno

Informo en consola o cuadro de mensajes

////////// FIN de Eliminar trayectoria //////////

////////// Inicio de Modificar trayectoria //////////

////////// FIN de Modificar trayectoria //////////

////////// Inicio de Correr trayectoria //////////

F correr() altero el msg “msj_punto_web” al poner “orden” en ves de “acción” y mando directamente este msj ◄

Colocar en una variable la tabla de la rutina “ serie “

Colocar en una variable la las filas de la tabla

Crear variable con cantidad de filas(sacando el encabezado)

FOR en base a la cantidad de filas

Colocar en una variable la fila de la iteración

Crear msj del tipo punto_web con componentes vacíos ▼

Colocar en una variable la las coordenadas de filas de la iteración

Transformar estas coordenadas (que son string) a un array de float (873 a 877)

Colocar en una variable la velocidad de filas de la iteración

Agregar la velocidad al array de float

Colocar este array en el msj (en el componente “coordenada”)

IF-ELSE para definir componente “orden” del msj, indicando principio, fin e intermedio de la rutina

Publicar en “pub_orden, orden_web ”

F publicarMensaje(mensaje) ◄

////////// FIN de Correr trayectoria //////////

F correr_launch() *(puedo hacer un solo launch on/off)* ◄

Conectarse al tópico “ topic, control_launch , std_msgs/String “ ▲

Crear msj tipo std_msgs/String (data: ‘prender’)

Publicar en “ topic, control_launch “

Informo en consola o cuadro de mensajes

F finalizar_launch() (puedo hacer un solo launch on/off) ◄

Conectarse al tópico “ topic, control_launch , std_msgs/String “ ▼

Crear msj tipo std_msgs/String (data: ‘prender’)

Publicar en “ topic, control_launch “

Informo en consola o cuadro de mensajes

////////// FIN LAUNCH ///////

////////// Inicio cambio de modo ///////

let nombre_temporal_rutina; ◄

let contadorRutina = 1; ◄

F activarModoEscritura() ▲ (tal vez se reemplace con una página) (revisar porque hay subscripciones)

F activarModoLectura() ▲ (tal vez se reemplace con una página) (revisar porque hay subscripciones)

let enModoActuar = false; // Flag para controlar el envío de mensajes ◄

F que corre al recibir msg y Subscripción de “Modo_actual, modoActuarTopic” ▲

SI enModoActual(true)

enModoActuar = false;

Salir de función

SI componente data del msg recibido = true

| SI “ modo_funcionamiento “ diferente de 1

| Correr “F activarModoEscritura() “ (o pasar a la otra página) ◄

SONO

SI “ modo_funcionamiento “ diferente de 0

Correr “F activarModoLectura() “ (o pasar a la otra página) ◄

Informo en consola o cuadro de mensajes

F botón_modoEscritura() (puedo hacer una sola función Escritura/Lectura) ◄

SI “ modo_funcionamiento “ diferente de 1

enModoActuar = false

Crear msj tipo std_msgs/Bool (data: ‘true’)

Publicar en “Modo_actual, modoActuarTopic “

Correr “F activarModoEscritura() “ (o pasar a la otra página) ◀

F botón_modoEscritura() (puedo hacer una sola función Escritura/Lectura) ◀

SI “ modo_funcionamiento “ diferente de 0

enModoActuar = true

Crear msj tipo std_msgs/Bool (data: ‘false’)

Publicar en “Modo_actual, modoActuarTopic “

Correr “F activarModoLectura() “ (o pasar a la otra página) ◀

let enTipoModoLectura = false; // Flag para controlar el envío de mensajes ◀

let tipo_modo_lectura=1; ◀

F que corre al recibir msg y Subscripción de “Tipo_modo_lectura, tipoModoLectura” ▲◀

F modo_Punto () (puedo hacer una sola función punto/transmision) ◀

Es una función para cambiar la forma del modo lectura, si lee por punto o va leyendo en toda la trayectoria

SI “tipo_modo_lectura “ diferente de 1

enTipoModoLectura = true

Crear msj tipo std_msgs/Bool (data: ‘true’)

Publicar en “Tipo_modo_lectura, tipoModoLectura “

tipo_modo_lectura=1;

F modo_transmision () (puedo hacer una sola función punto/transmision) ◄

Es una función para cambiar la forma del modo lectura, si lee por punto o va leyendo en toda la trayectoria

SI “tipo_modo_lectura “ diferente de 0

enTipoModoLectura = true ◄

Crear msj tipo std_msgs/Bool (data: ‘false’)

Publicar en “Tipo_modo_lectura, tipoModoLectura “

tipo_modo_lectura=0

Dentro de un try/carch para manejar errores

Regresa la entrada (en bit) pasado a grados

F actualizarAngulos_M_lectura(message)

FOR en base a las 6 coordenadas angulares(inputs)

Pasar los valores de la entrada de bit a grados (“F;bit_grados(bits) “)

Colocar estos valores pasados en el input angular de la iteración

F agregar_punto_M_lectura(message) (1223) revisar si es necesaria ◄

F inicializarInputs (1261) revisar si es necesaria ◄