Universidad Zaragoza
1542

Trabajo Fin de Grado

Estudio de la viabilidad de robotización de un bar mediante modelado y simulación de sistemas dinámicos

Feasibility study of a pub robotization using modeling and dynamic systems simulation

Autor

Óscar Pérez Montón

Director

Carlos Sánchez Tapia

Escuela Universitaria Politécnica de Teruel

2024

RESUMEN

Este trabajo fin de grado tiene como objetivo valorar la viabilidad de incluir varios robots que desempeñen funciones de camarero en un bar nocturno, atendiendo a la variedad de peticiones posibles por parte de los clientes y dando servicio a un número variable de éstos.

Para ello se ha usado el programa RobotStudio para realizar el modelo y programación de los brazos robóticos, para este modelo se ha utilizado el Robot IRB 1100 E10, el cual es un modelo de brazo robótico de la compañía ABB comprado recientemente por la Escuela Politécnica De Teruel. También se ha usado el programa JaamSim para realizar un modelo de simulación de eventos discretos del bar real y del bar robótico, con el propósito de obtener una comparativa del desempeño de estos durante varios años.

Para aumentar al máximo la fidelidad de los modelos de simulación de eventos discretos se ha realizado un estudio estadístico de los tiempos empleados en servir por camareros reales.

ABSTRACT

The objective of this final degree project is to assess the feasibility of including several robots that perform bartending functions in a night bar, attending to the variety of possible requests from customers and providing service to a variable number of them.

For this purpose, the RobotStudio program has been used to make the model and programming of the robotic arms, for this model has been used the Robot IRB 1100 E10, which is a model of robotic arm of the ABB company recently purchased by the Escuela Politécnica De Teruel. The JaamSim program has also been used to perform a discrete event simulation model of the real bar and the robotic bar, with the purpose of obtaining a comparison of their performance over several years.

In order to maximize the fidelity of the discrete event simulation models, a statistical study of the serving times of real bartenders has been carried out.

AGRADECIMIENTOS

Este apartado está destinado íntegramente para expresar mis agradecimientos a todas las personas que me han facilitado la elaboración de este Trabajo Fin de Grado.

Gracias a mi director de proyecto, Carlos Sánchez Tapia, por darme la oportunidad de realizar el Trabajo Fin de Grado con él, valoro gratamente su labor como orientador, así como su esfuerzo y dedicación en las asignaturas que he cursado con él como docente.

De igual manera, me gustaría agradecer a todos aquellos profesores que, a lo largo de toda mi etapa académica, me han ayudado y orientado, motivándome a aprender y mejorar.

Finalmente, me gustaría dedicar un último agradecimiento a mis padres y abuelos, por proporcionarme siempre los mejores recursos a su disposición, así como por ofrecerme su apoyo incondicional durante tantos años de mi vida estudiantil.

ÍNDICE DE CONTENIDO

ANEXOS

ANEXO I Ficheros modelos de simulación JaamSim.

ANEXO II Resultados numéricos simulaciones JammSim.

ANEXO III Fichero bar robótico RobotStudio.

ANEXO IV Muestras estudio estadístico.

ANEXO V Video funcionamiento modelo RobotStudio.

ANEXO VI Archivos de biblioteca RobotStudio.

ÍNDICE DE FIGURAS

INTRODUCCIÓN

1) MOTIVACIÓN

La temática de este trabajo final de grado tiene su origen en un problema ha ido aumentando en España en los últimos años. Debido a diferentes factores, como la pandemia del COVID-19 y la posterior inflación que está en estos momentos azotando Europa, los gastos de los autónomos y pymes han aumentado drásticamente [1]. Al aumento de los gastos en locales y productos hay que sumarle el incremento de costes laborales. Las empresas tienen que hacer frente a unos gastos que se sitúan en los 2.996,63 € por trabajador, conllevando un 4% más respecto al tercer trimestre de 2022 [2].

De estas imposiciones nace la idea de intentar crear una nueva forma de usar la robótica en el sector hotelero, la cual reduzca los gastos laborales de estas empresas y aumente a su vez el interés de los clientes por la misma.

1) OBJETIVOS

A continuación, se detallan los objetivos principales de este Trabajo Fin de Grado:

-Estudiar de manera exhaustiva la robótica en el ámbito de la hostelería y de qué manera esta puede implementarse de manera realista y económicamente favorable.

-Realizar un modelo optimizado de un bar robótico en el programa de RobotStudio, en el que se intente reducir tanto como sea posible el tiempo de cada pedido, reduciendo al máximo la intervención humana.

-Realizar un modelo con la mayor fidelidad posible de un bar real y el bar robótico, con el programa JaamSim, el cual sea capaz de simular una gran cantidad de jornadas de trabajo y mostrar los resultados obtenidos.

-Obtener conclusiones sobre la viabilidad física y económica del proyecto, partiendo del análisis estadístico de los datos extraídos de las simulaciones.

2) ESTADO DE LA TÉCNICA

Ya que esta simulación tiene un componente de innovación, se ha decidido analizar los diferentes robots ya existentes utilizados en el sector hostelero, los resultados obtenidos indican que la técnica se encuentra en un estado de desarrollo temprano.

Los primeros datos registrados de la utilización de un robot en el sector hostelero se remontan a 2010 con el robot "Motoman", un prototipo robótico utilizado en un restaurante de sushi en la ciudad de Bangkok [3].

Tras los primeros prototipos, muchas han sido las empresas interesadas en desarrollar sus propias soluciones para adaptar el mundo de la robótica al sector de la hostelería. Actualmente existe una gran oferta de robots camareros, las empresas más destacadas son las siguientes:

2.1) Pudu

Pudu es una empresa con sede en Shenzhen, fundada en 2016 la empresa asiática ofrece una gran variedad de robots camareros de tipo carro, de todo su catálogo el más famoso es el Bellabot [4].

Bellabot se trata de un robot utilizado en el sector de la restauración, el cual se encarga de llevar los platos desde la cocina hasta los comensales, es capaz de comunicarse con las personas ya que dispone de una pantalla interactiva y una voz controlada por inteligencia artificial, este robot es capaz de realizar más de 400 repartos en un día, su precio es de 8000€.

Figura 1. Bellabot[5]

2.2) DAX robotics

La empresa DAXrobotics es una empresa de origen español que dispone de varios modelos de robots camareros, los robots que esta empresa fabrica son robots tipo carro, encargados de llevar los productos a los comensales de manera 100% autónoma[6].

Los modelos que actualmente comercializa esta empresa son **delibot** y **slimbot.**

Figura 2.Robot delibot[6]

Figura 3. Robot slimbot[6]

2.3) CaliExpressbyFlippy

Se trata del primer restaurante robótico del mundo, situado en california, la hamburguesería totalmente autónoma, utiliza brazos robóticos para realizar todas las tareas que típicamente realizan los cocineros y camareros.

Los pedidos se realizan a través de una pantalla, los ingredientes utilizados y las técnicas de cocina son iguales que en cualquier hamburguesería común, es decir los robots no necesitan hamburguesas u otro tipo de ingredientes especiales para cocinar la comida[7].

Figura 4. Robot restaurante CaliExpress[7]

2.4) MakrShakr

Se trata de un robot capaz de preparar cocteles de manera autónoma, el sistema cuenta con unos dispensadores especiales que se colocan en las botellas, estas se cuelgan del techo y el robot tiene que presionar el dispensador con un vaso para servir las diferentes bebidas, este proyecto empezó en 2013 para una exposición de Google, actualmente el robot está en producción y ya está operativo en diferentes lugares del mundo, concretamente 4 localizaciones en Europa, 3 en América, 1 en Asia y 9 en cruceros que viajan por diferentes partes del mundo[8].

Actualmente la empresa también oferta otra configuración de robots la cual es capaz de preparar café.

Figura 5. Robot MakrShakr[8]

2.5) Comparativa de mercado, situación actual

Tras comparar la oferta actual de robots camareros en el mercado, se puede observar que la idea de un robot camarero con brazos robóticos es una idea factible y que puede tener un hueco en el mercado. Actualmente la robótica en el ámbito de la hostelería está dominada por los robots de tipo carro, los cuales han demostrado tener un nivel de desempeño lo suficientemente elevado para ser usados en un restaurante real, con el objetivo de aumentar su producción. Estos robots no suponen competencia, ya que el robot y las tareas robóticas que se van a desarrollar en este TFG tienen como objetivo ser una alternativa en el ocio nocturno.

La solución propuesta por **CaliExpressbyFlippy** se asemeja a lo que se quiere analizar en este trabajo, centrado en la restauración, ya que la manera en la que usa los brazos robóticos invita a pensar que es posible el desarrollo de un sistema parecido, adaptando este al ocio nocturno.

Por último, evidentemente, el sistema creado por **MakrShakr** es el más parecido a lo que se estudia en este proyecto, ya que utiliza brazos robóticos comunes de la marca Kuka para realizar de manera autónoma cócteles, demostrando tanto que la idea de este TFG es posible ejecutarla como que hay locales interesados en este tipo de dispositivos.

DESARROLLO

1) SOFTWARE

En esta sección se va a explicar las características principales del software usado para realizar las simulaciones:

1.1) RobotStudio

RobotStudio es un software de simulación y programación propiedad de la empresa ABB, una de las principales características y ventajas es que este software crea una réplica virtual completa, lo que hace que se pueda simular y probar una instalación completa de robots en un entorno virtual 3D.Para este TFG se ha usado la versión de 2023 de RobotStudio [9].

Las razones por las que se ha elegido este software son:

-Los robots que se van a usar para realizar la simulación son de la marca ABB y RobotStudio es el software de la empresa ABB.

-La Escuela Universitaria Politécnica de Teruel dispone de licencias de este software.

-Se trata de un software de programación y simulación robótica muy avanzado, que se actualiza continuamente y que cuenta con multitud de posibilidades.

1.2) JaamSim

JaamSim es un software de simulación de eventos discretos, el cual se va a usar para realizar una simulación de eventos discretos, comparando un modelo de un bar real, con el modelo de bar robótico creado en RobotStudio [10].

Las razones por las que se ha elegido este software son:

-Se trata de un software gratuito de código abierto (aunque existe una versión de pago que no aporta mejoras significativas para los objetivos en los que se va a emplear en este TFG).

-Tiene una interfaz clara e intuitiva, en la que se muestra visualmente el funcionamiento en cualquier momento de una simulación.

-Aunque existen programas que podrían emplearse para la simulación, como por ejemplo OpenModelica, JaamSim es un programa que personalmente ya he utilizado en la asignatura de Simulación de Sistemas Dinámicos.

2) MODELO ROBOTSTUDIO

2.1) Elección del robot

El robot que se ha elegido para esta simulación es el Robot IRB 1100, este es un brazo robótico de 6 ejes, con una carga útil de 4 kg y un alcance de 580mm, el controlador utilizado en este robot es el OmniCore E10 [11] .

Este robot se adapta a las necesidades que se demandan en este proyecto, ya que tiene un alcance suficiente para la actividad que se va a realizar y no se va a necesitar cargar ningún objeto que supere la carga útil. Hay más robots en el catálogo de ABB que cumplan estas características, pero se ha optado por este modelo ya que la Escuela Universitaria Politécnica de Teruel ha adquirido una unidad de este robot recientemente, lo que facilitaría trasladar en el futuro los resultados de simulación de este TFG a experimentación real (aunque esto escapa al ámbito de realización y extensión del presente TFG).

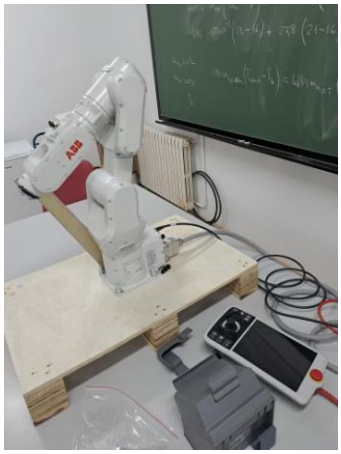

Figura 6. Robot IRB 1100de la EUPT

En el modelo de bar diseñado en este TFG, se han empleado dos robots con tareas y herramientas específicas para cada uno (en adelante: robot1 y robot2), pero capaces de interaccionar entre sí.

2.2) Modelado 3D de objetos

Con el objetivo de realizar una simulación lo más profesional y vistosa posible se han modelado en 3D los diferentes objetos que se van a necesitar dentro de la simulación, los modelos 3D se han hecho con el propio programa de RobotStudio desde la pestaña de Modelado. Si bien es cierto que las herramientas que dispone para el modelado en 3D el

propio RobotStudio son limitadas, tras buscar información parece ser la única manera de utilizar modelos 3D dentro del programa, a pesar de estas limitaciones, el objetivo de estos modelos es el de representar con la mayor fidelidad posible los objetos que manipularía el robot en la realidad.

Para conocer las medidas y geometría de los diferentes objetos se han utilizado diferentes técnicas:

-Para las botellas de bebidas alcohólicas, vasos, cajas de botellines, abrebotellas y botellines, personalmente he tomado las medidas de estos, utilizando un calibre y una cinta métrica de sastre.

-Para la máquina expendedora de hielos he tomado las medidas de la página web de la referencia [12]

-Para el resto de objetos, los cuales incluyen la barra con papelera, cristaleras, pies para los robots y el atril para las cajas de los botellines, las medidas son aproximadas ya que estos objetos se han creado expresamente para el modelo, si se implementara en la realidad estos objetos podrían variar sus alturas para adaptarse a las alturas requeridas en una barra de un bar real, siempre y cuando se mantenga la relación que existe en el modelo entre las alturas de todos estos objetos.

A continuación, se van a mostrar los diferentes modelos 3D con algunas de sus características principales:

-En la Figura 7 se observa el modelo 3D creado para la caja de botellines este tiene unas medidas de 300mm de ancho X 400mm de largo X 210mm de alto, cuenta con 30 huecos para botellines, 6 a lo largo y 5 a lo ancho, los cuales son de 57mm X 57mm.

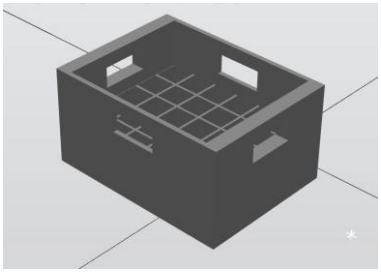

Figura 7. Modelo 3D caja de botellines

-En la Figura 8 se muestran los modelos 3D creados para los botellines. Los botellines que se han tomado como referencia son de 20cl, que son los que se usan típicamente en hostelería,

con medidas de 52mm de ancho X 198mm de alto. Como se puede observar, hay 5 tipos de botellines: Fanta de limón, Fanta de naranja, tónica, Coca-Cola y cerveza. Los 5 modelos tienen la misma forma, pero cambian las opciones gráficas de cada uno de ellos, cambiando los colores de los cuatro cuerpos de los que constan: parte inferior, parte superior, etiqueta y chapa.

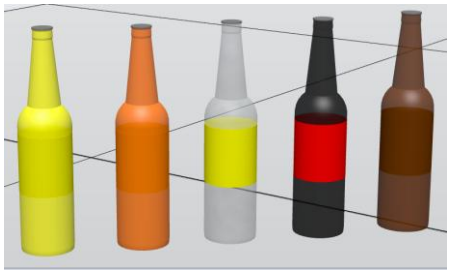

Figura 8. Modelos 3D botellines

-En la Figura 9 se muestra el modelo 3D que se ha creado para la botella de ginebra. Para este modelo se ha tomado como referencia una botella de la marca Beefeater de 75cl: una botella cuadrada con medidas 71mm de largo X 71mm de ancho X 266mm de alto.

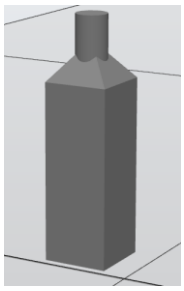

Figura 9. Modelo 3D botella ginebra

-En la Figura 10 se muestra el modelo 3D que se ha creado para la botella de whisky. Para este modelo se ha tomado como referencia una botella de la marca Ballantines de 1l: una botella rectangular con medidas 101mm de largo X 61,5mm de ancho X 265mm de alto.

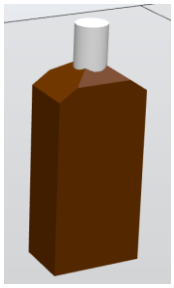

Figura 10. Modelo 3D botella whisky

-En la Figura 11 se muestra el modelo 3D que se ha creado para la botella de vodka. Para este modelo se ha tomado como referencia una botella de la marca Smirnoff de 75cl: una botella cilíndrica con unas medidas de 35,5mm de radio X 265mm de alto.

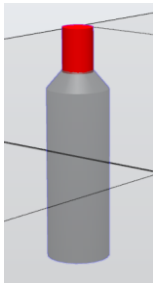

Figura 11. Modelo 3D botella vodka

-En la Figura 12 se muestra el modelo 3D que se ha creado para la botella de ron. Para este modelo se ha tomado como referencia una botella de la marca Brugal de 75cl: una botella cilíndrica con unas medidas de 35,5mm de radio X 265mm de alto.

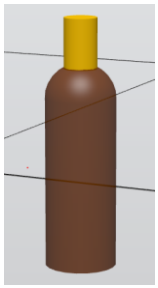

Figura 12. Modelo 3D botella de ron

-En la Figura 13 se muestra el modelo 3D que se ha creado para el vaso. Se ha tomado como referencia un vaso con capacidad de 500ml.Este tipo de vaso es el que se usa típicamente en hostelería para servir las copas. Es denominado vaso de sidra y sus medidas son 90mm de diámetro en la parte superior X 70mm en la parte inferior X 116mm de altura.

Figura 13. Modelo 3D vaso

-En la Figura 14 se muestra el modelo 3D de la máquina de hielos. Las medidas de este modelo se han obtenido de la página web de la [12].En el modelo se observan dos partes, el cuerpo de la maquina en color gris y la palanca que acciona la maquina en color negro.

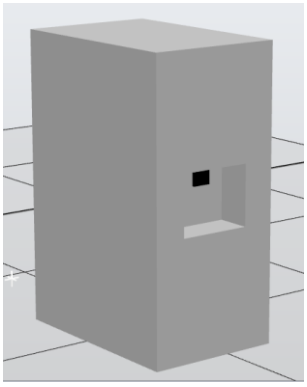

Figura 14. Modelo 3D máquina expendedora de hielos

-Los modelos de las Figuras 15, 16 y 17 muestran modelos 3D creados de manera exclusiva para la simulación. Estos modelos tienen unas medidas aproximadas para encajar en la simulación. En el caso de que esta simulación se implementara en un prototipo real, las alturas de estos objetos podrían cambiar, aunque debería mantenerse la diferencia de alturas entre los modelos de las Figuras 15, 16 y 17.

El modelo de la Figura 15 se trata de una barra en forma de U con una papelera integrada en la parte derecha de la misma. Esta barra se utilizará para apoyar la mayoría de modelos que se han mostrado con anterioridad.

El modelo de la Figura 16 se trata de una cristalera que separa la parte donde se encuentran los robots de los clientes. Esta cristalera tiene el propósito de evitar accidentes y hurtos. Para esta simulación solo se ha modelado la parte de la cristalera donde el robot deja los pedidos

terminados. No se ha rodeado por completo toda la zona del robot para no entorpecer la visión de la simulación.

El modelo de la Figura 17 es simplemente un pie que se coloca debajo del robot para elevarlo y que así este pueda aprovechar de mejor manera su envolvente de trabajo, ya que este robot puede posicionar el TCP ("Tool Center Point": punto donde se ancla la herramienta a la muñeca del robot) por debajo de su zona de anclaje.

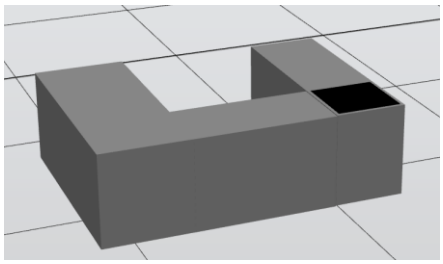

Figura 15. Barra con papelera

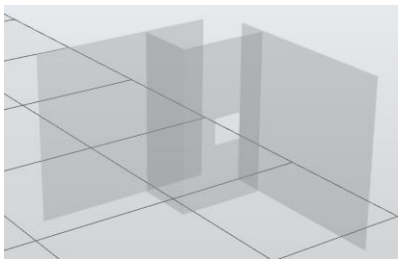

Figura 16. Cristalera

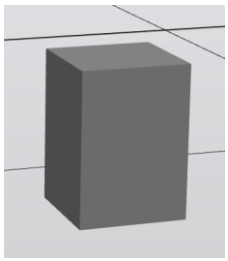

Figura 17. Pie robot

2.3) Componentes inteligentes

En este apartado se van a explicar los diferentes componentes inteligentes que se han creado para la simulación, tanto su objetivo como su funcionamiento.

2.3.1) Ventosa

El componente inteligente tiene el nombre de ''ventosa_smart'' se trata de la ventosa que se ha utilizado como herramienta para el robot 1.

Para crear la ventosa, primero se ha modelado en 3D el cuerpo de la misma, el cuerpo se ha añadido dentro del componente inteligente para que este forme parte de él y se ha posicionado el TCP en el extremo del modelo como puede verse en la Figura 19. En la pestaña "componer'' dentro del componente inteligente se han añadido varios elementos:

-**LineSensor:** se ha añadido un line sensor que sobresale un poco del TCP para que detecte cuando hay una pieza cerca, las medidas del sensor son de 15mm de largo X 0,62mm de radio.

-**Attacher:**el attacher se va a usar para conectar dos objetos cuando este se active. Dentro de su configuración se ha establecido como objeto superior la ventosa_smart y en los datos de la herramienta a conectar se ha seleccionado el TCP. Esto se utiliza para que se pueda conectar cualquier objeto a la herramienta.

-**Detacher:** el detacher se va a usar para desconectar dos objetos que están conectados cuando este se active.

-**LogicGate [NOT]:** se usa para negar una señal.

-**LogicSRLatch:** se va a usar para aplicar o restablecer una señal.

Una vez se tienen listos todos los componentes de la ventosa, en la pestaña diseño hay que unirlos para conseguir el funcionamiento deseado como se observa en la Figura 18. En primer lugar, se crea una entrada y una salida del componente: la entrada recibe el nombre de vacío y la salida el nombre de vacuostato.

La entrada vacío se conecta al "Active" del LineSensor y al "InputA" de la logicGate: cuando la entrada vacío se active, el sensor estará activo; cuando una pieza corte el sensor, la salida del LineSensor " SensorOut" se pondrá a uno. Esta se encuentra conectada a la entrada "Execute" del Attacher, por lo que lo activará y conectará las piezas. El Attacher conectará la pieza que ha detectado el sensor, gracias a que la salida "SensedPart" del LineSensor está unida a la entrada "Child" del Attacher, lo que hace que la pieza que detecte el sensor se una a la pieza que se ha indicado como objeto superior en el Attacher, que como se ha indicado antes es el cuerpo de la ventosa. Cuando se conectan las dos piezas se activa la salida "Executed" del Attacher, lo que activa la entrada "Set" de la LogicSRLatch, la cual, a su vez, activa la salida vacuostato del componente. Esta salida activa indica que el componente se ha unido correctamente.

Cuando la entrada vacío se desactiva, la LogicGate recibirá un 0 en su entrada "InputA" y su salida "Output" pasará a valer 1.Esta salida está conectada a la entrada "Execute" del Detacher, que cuando se activa desconecta los dos objetos. Nuevamente se indican los objetos que hay que desconectar porque la salida "SensedPart" del LineSensor está unida a la entrada "Child" del Detacher. Cuando se desconectan las dos piezas se activa la salida "Executed" del Detacher, lo que activa la entrada "Reset" de la LogicSRLatch, la cual desactiva la salida vacuostato del componente. Esta salida desactivada indica que el componente se ha soltado correctamente.

Por último, la salida "SimulationStopped" del SimulationEvents se conecta a la entrada "Reset" de la LogicSRLatch para asegurarme que la salida vacuostato del componente siempre se queda a cero cuando se termina la simulación.

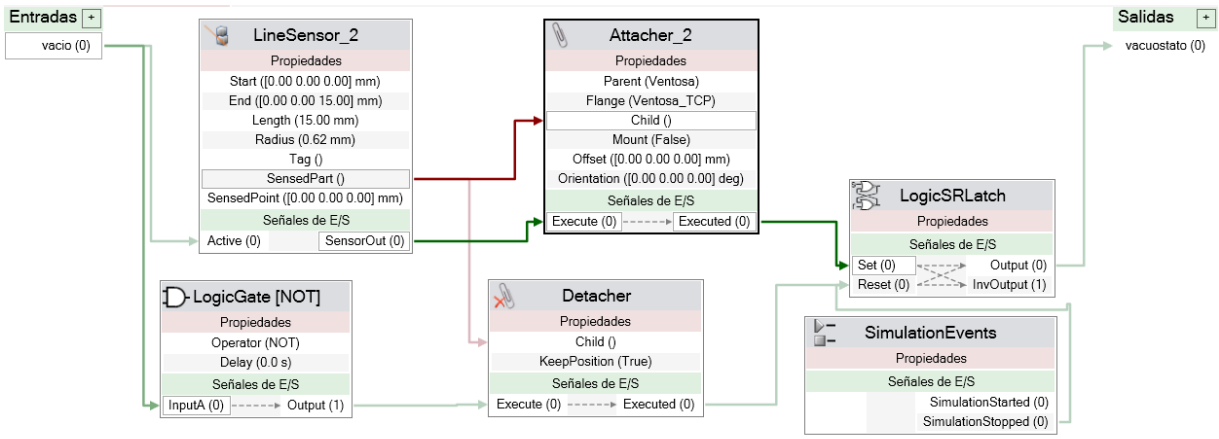

Figura 18. Diseño ventosa_smart

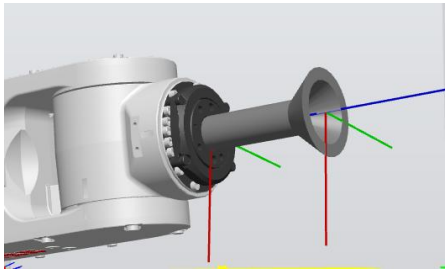

Figura 19. ventosa_smart

2.3.2) Gripper

El componente inteligente tiene el nombre de ‘’Smart_gripper_line_sensor’’. Se trata de la pinza que se ha utilizado como herramienta para el robot 2.

El modelo 3D de la herramienta se ha importado de la biblioteca de ABB que se encuentra dentro de RobotStudio. Este recibe el nombre de “ABB Smart Gripper”.Una vez se tiene el modelo 3D, se ha posicionado el TCP entre los dos dedos de la pinza como puede verse en la Figura 21. Una vez se tiene el “ABB Smart Gripper” dentro del componente inteligente, desde el desplegable en la opción “Modificar Mecanismo” se han programado dos posiciones de los dedos de la pinza, una con el nombre de “abierto” con un valor del eje de 25 y otra con el nombre de “cerrado_cuello” con un valor del eje de 10,84. Este es el valor para que los dedos de la pinza ajusten en el cuello de la botella.

En la pestaña “componer’’dentro del componente inteligente se han añadido varios elementos:

-**Attacher:** el attacher se va a usar para conectar dos objetos cuando este se active. Dentro de su configuración se ha establecido como objeto superior 'Smart_gripper_line_sensor'' y en los datos de la herramienta a conectar se ha seleccionado "servo". Esto se utiliza para que pueda conectar cualquier objeto a la herramienta.

-**Detacher:** el detacher se va a usar para desconectar dos objetos que están conectados cuando este se active.

-**LogicGate [NOT]:** se usa para negar una señal.

-**LogicSRLatch:** se va a usar para aplicar o restablecer una señal.

-**PoseMover:** se usa para mover los mecanismos hasta una posición predefinida. En este caso se van a usar dos: uno para abrir los dedos de la pinza hasta la posición abierto y otro para cerrar la pinza hasta la posición cerrado. También se ha añadido que la duración del movimiento sea de 1 segundo.

Una vez se tienen listos todos los componentes de la pinza, en la pestaña diseño hay que unirlos para conseguir el funcionamiento deseado como se observa en la Figura 20. En primer lugar, se crea una entrada y dos salidas del componente: la entrada recibe el nombre de pinza y las salidasFC_cuello_cerrado y FC_abierto.

El funcionamiento es similar al de la ventosa: al activar la señal pinza, el LineSensor se activa; cuando este detecta una pieza le pasa la pieza que ha detectado a la entrada "Child" del Attacher, indicándole así la pieza que la tiene que unir a la pinza; al mismo tiempo el PoseMover[cerrado_cuello] se ha ejecutado con la señal pinza, lo que mueve los dedos de la pinza a la posición cerrado_cuello que se ha programado anteriormente. Una vez se ha cerrado la pinza, se activa el Attacher para que conecte la pieza y activa la señal de salida del componente FC_cerrado_cuello. Esta señal indica que la acción de coger la pieza ha finalizado.

Para soltar la pieza, la entrada se pone a cero. Gracias a la LogicGate[NOT], se activa el PoseMover[abierto], el cual mueve los dedos de la pinza a la posición abierto que se ha programado anteriormente. Una vez la pinza se ha abierto, se activa el Detacher para que desconecte la pieza y activa la señal de salida del componente FC_abierto, esta señal indica que la acción de soltar la pieza ha finalizado.

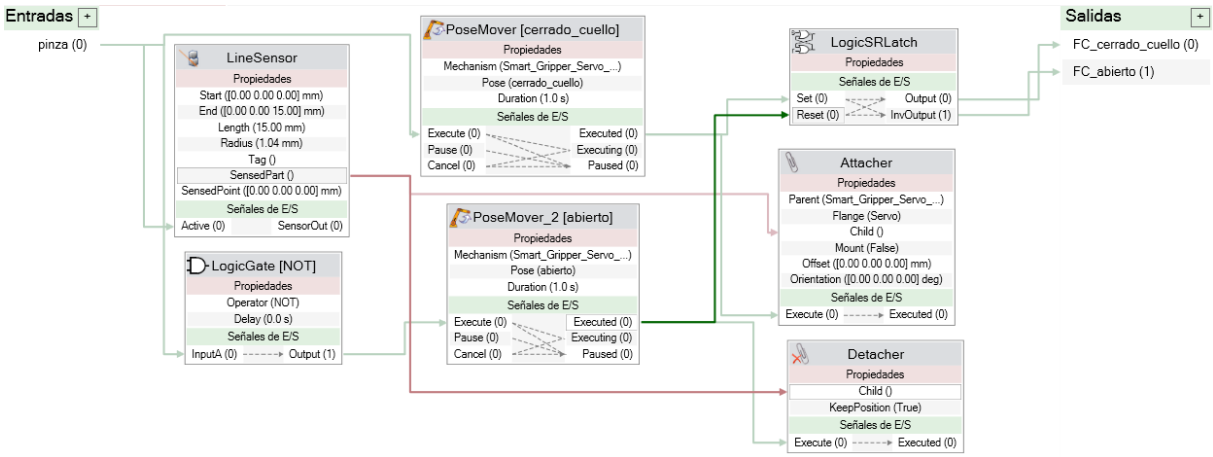

Figura 20. Diseño smart_gripper

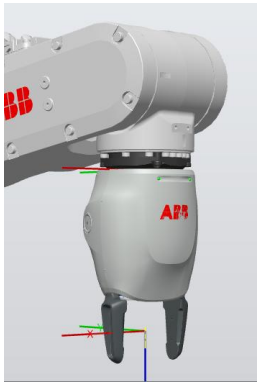

Figura 21. smart_gripper

2.3.3) Depósito

Este componente inteligente, a diferencia de la pinza y la ventosa, se ha creado con un propósito estético en la simulación. No trata de replicar ningún objeto real, como ocurre con la pinza o la ventosa. El objetivo de este componente inteligente es cambiar la posición de la copa cuando el robot la sirve, para simular que un cliente ha cogido la copa.

Los componentes utilizados en este componente inteligente son los siguientes:

PlaneSensor: sensor que detecta si algún objeto corta el plano.

Timer: activa una señal digital con el intervalo de tiempo que se le haya indicado.

Positioner: define la posición de un objeto.

En la pestaña diseño se han conectado los diferentes componentes, como puede observarse en la Figura 22. Este componente inteligente no tiene ni entradas ni salidas para activarlo. Cuando algún objeto corta el PlaneSensor (en este caso la copa terminada), este se activa y

pasa la información de la pieza que ha cortado el sensor a la entrada “Object” del Positioner, la salida del PlaneSensor activa el Timer. Cuando pasan 4 segundos se activa la salida del Timer lo cual ejecuta el Positioner que mueve el modelo de la copa dejando libre el lugar donde el robot las sirve.

Los objetos se mueven a la posición (-447mm,889mm,0mm), en la que se encuentra la máquina de hielos. El objetivo de moverlos a esta posición es ocultarlos en un lugar donde no se vean en la simulación. Se ha optado por mover los objetos y no eliminarlos para poder restablecer la simulación al estado original al acabar de simular, ya que si se borran los objetos no se puede restablecer la simulación a la situación original.

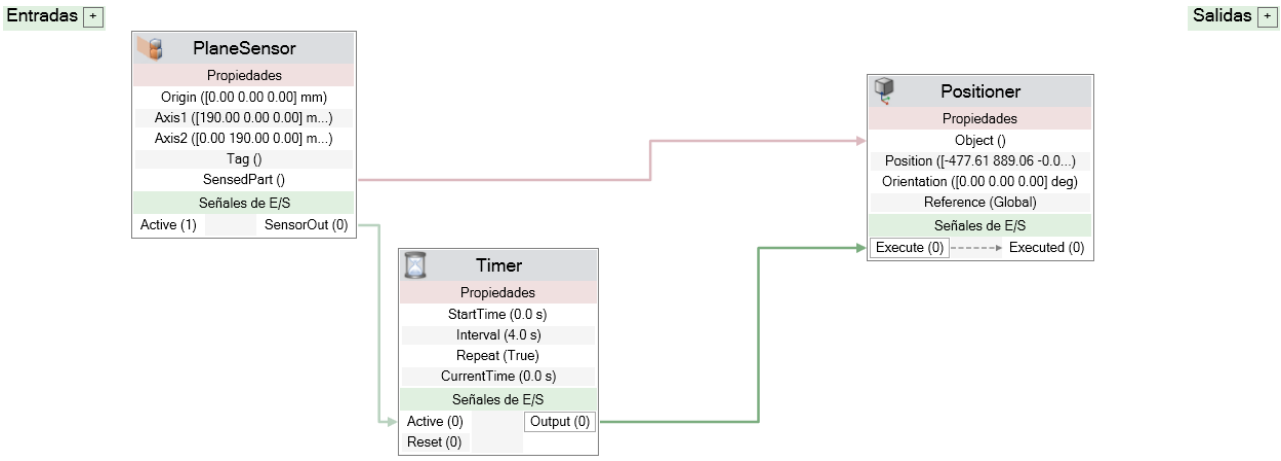

Figura 22. Diseño deposito_smart

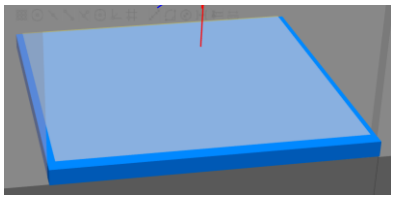

Figura 23. deposito_smart

2.3.4) Accionador

Este componente inteligente se ha creado para controlar los pedidos. Se trata de una botonera en la que activando sus salidas los robots realizarán los diferentes pedidos.

Como puede verse en la Figura 24, el funcionamiento es muy sencillo. Se han creado como entradas todos los tipos de bebida que puede servir el robot, como salidas se han creado las mismas que las entradas y se han unido las entradas con su salida correspondiente. Posteriormente, las salidas del componente inteligente se han unido con las entradas del robot

correspondiente, como veremos más tarde en el apartado de lógica de la estación. Con esto conseguimos, como se puede ver en la Figura 25, una botonera en la que podemos gestionar las entradas digitales de una manera más intuitiva que la proporcionada por RobotStudio.

Figura 24. Diseño accionador_smart

Figura 25. Botonera accionador_smart

2.4) Posición de los robots, componentes inteligentes y modelos 3D

En este apartado veremos la posición elegida para todos los elementos de la simulación.

En primer lugar, se procede a posicionar los dos robots junto con sus herramientas, la altura de los atriles que soportan los robots es de 230mm y la distancia entre los dos robots es de 1100mm. Como puede observarse en la Figura 26, las envolventes de trabajo tienen puntos en común entre los dos robots. Esto es premeditado con el objetivo de que los robots lleven a cabo tareas colaborativas, por lo que tiene que haber una zona en la que los dos robots puedan trabajar de forma simultánea.

A parte de la posición de los robots, también se han unido las herramientas a los mismos. El robot 1 es el que tiene la ventosa, el cual se encuentra en la parte izquierda en la Figura 26, y el robot 2 es el robot con la pinza, y se encuentra en la parte derecha de dicha figura.

La posición en la que se encuentran los ejes de los robots en la Figura 26 es la que se ha elegido y programado como posición de referencia de cada uno de los robots. En estas dos posiciones de referencia se evitan singularidades, las cuales son posiciones donde el robot pierde un grado de libertad, generando problemas como aceleraciones muy elevadas de sus motores, que pueden comprometer la integridad del robot. A lo largo de la programación del modelo de RobotStudio se ha prestado especial atención a evitar singularidades.

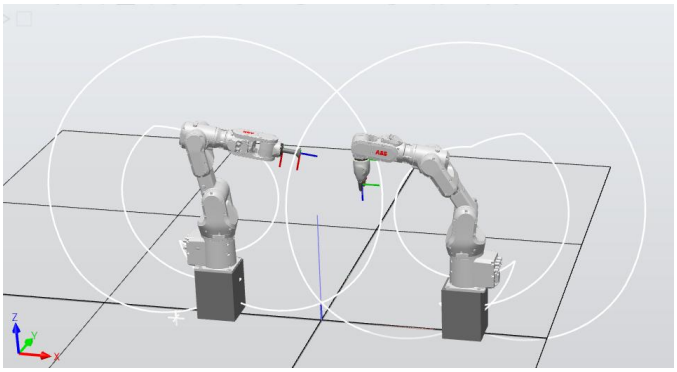

Figura 26. Posición robots

Después de posicionar los robots con sus herramientas, se han posicionado todos los modelos 3D. Primero se han generado 30 modelos de cada uno de los refrescos y se han introducido dentro del modelo de la caja de refrescos, como puede verse en la Figura 27. Los objetos se han añadido como cuerpos dentro de la pieza “caja_de_botellines”, de manera que al mover la caja todos los botellines se mueven junto con esta, pero cuando el robot coge un botellín este se desconecta de la caja y puede moverse solo.

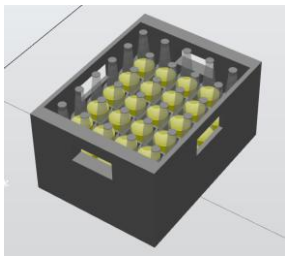

Figura 27. Caja refrescos llena

Una vez todas las cajas de refresco están llenas, se ha posicionado la barra y la máquina de hielo, como puede observarse en la Figura 28.La barra se ha puesto rodeando al robot 1 para

que así este llegue a todos los puntos de la misma. La zona de la barra que hay entre los dos robots coincide con la zona donde las dos envolventes de trabajo se cruzan, puesto que es en esta zona donde se prepararán las copas y los robots podrán coger el objeto que haya dejado el otro. Por otra parte, la máquina de hielo se ha posicionado dentro de la envolvente de trabajo del robot 1, en una posición que permita al robot 1 girar sin chocar con la máquina, ya que, si esta estuviera muy cerca el robot, no sería capaz de girar a la zona exterior de la barra, y si estuviera muy lejos, este no llegaría para servir los hielos.

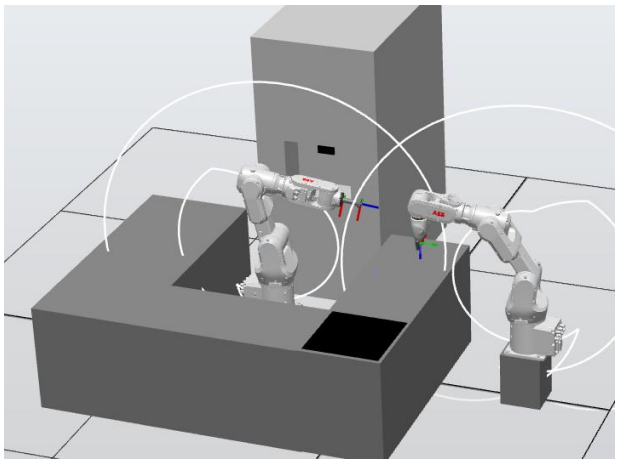

Figura 28. Posición barra y máquina de hielo

En la Figura 29 se puede observar la posición que se ha elegido para todos los modelos 3D. Dentro del volumen de trabajo del robot 1 se han colocado los vasos, la caja de cervezas, las botellas de bebidas alcohólicas y el deposito_smart. Los vasos se han colocado en la zona exterior de la barra, cerca de la máquina de hielo, con el objetivo de reducir los movimientos del robot y por lo tanto los tiempos de servicio, ya que siempre que el robot coja un vaso le pondrá hielo en la máquina. Las botellas de bebidas alcohólicas se han colocado en la parte frontal de la barra, la caja de cervezas se ha colocado entre el robot y la barra en una posición donde el robot sea capaz de alcanzar todos los botellines, y en la esquina exterior de la barra se ha colocado el depósito donde el robot servirá la copa al cliente.

Alrededor del robot 2 se han colocado las cajas de refresco. Las cuatro cajas se encuentran en una posición dentro de su volumen de trabajo, el cual da la posibilidad al robot 2 de alcanzar todos los botellines con la herramienta en posición vertical. Debido al reducido tamaño del robot IRB 1100, su alcance es bastante limitado, por lo que se han tenido que hacer numerosas pruebas hasta que se ha encontrado esta posición de las cajas. Aunque en un primer momento se intentó poner la caja de cervezas en la parte trasera del robot 2, no fue posible, ya que,

como puede verse en la representación gráfica del volumen de trabajo de la Figura 28, el robot en la parte trasera tiene menor libertad de movimiento, por lo que no se ha conseguido encontrar ninguna posición de la caja de cervezas donde el robot alcanzase todos los botellines. Por ello, se ha optado finalmente por posicionarla cerca del robot 1.

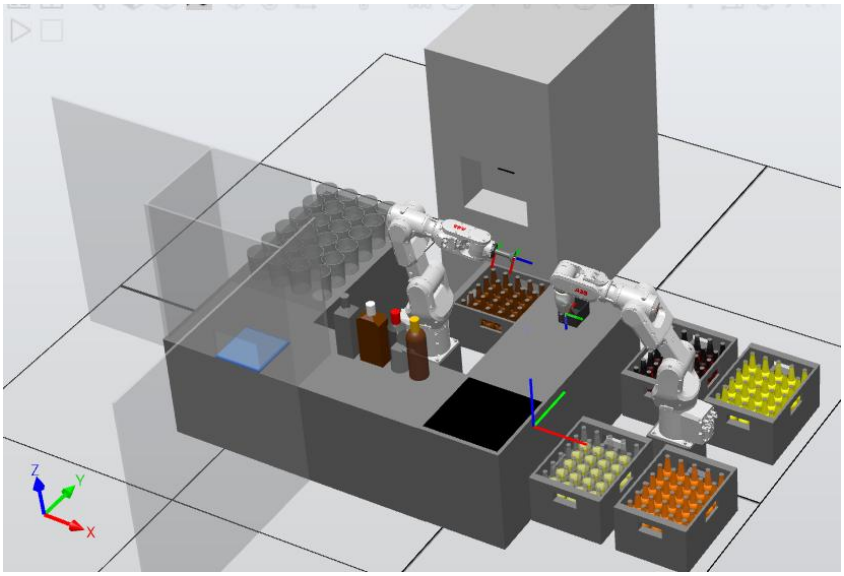

Figura 29. Posición modelos 3D

2.5) Controladores

Una vez se tienen los robots y todos los objetos colocados, se procede a introducir los controladores digitales. Los controladores son los encargados de procesar la información captada por las señales y regular el movimiento de los motores. En este caso se han utilizado dos controladores digitales OmniCore E10, los cuales disponen de 16 entradas y 8 salidas digitales cada uno. El límite de salidas y entradas digitales hay que tenerlo en cuenta,como se puede ver en el apartado de lógica de la estación. Se han utilizado dos controladores digitales, ya que el OmniCore E10 es capaz de controlar un solo robot. Para usar este controlador se ha necesitado descargar el paquete “RobotWare versión 7.12.0“, el cual se encuentra en la galería de complementos de RobotStudio.

2.6) E/S digitales

Para gestionar los componentes inteligentes y comunicar un robot con otro es necesario crear entradas y salidas digitales, las cuales se pueden gestionar desde el código en lenguaje RAPID (lenguaje de programación específico que se emplea en robots de la marca ABB). Para cada uno de los controladores se han generado las siguientes señales:

2.6.1) Controlador1:

En el controlador 1 se han creado las entradas vacuostato, refresco_c1, lista_servir_C1, solo_refresco_C1, cerveza, ron, whisky, ginebra y vodka. También se han creado las salidas vacio, vaso_bebida_C1, cogiendo_cerveza_c1, cerveza_lista_abrir_C1, Robot_1_libre_C1.

2.6.2) Controlador 2:

En el controlador 2 se han creado las entradas vaso_bebida_C2, cogiendo_cerveza_C2, cerveza_lista_abrir_C2, robot_1_libre_C2, tónica, Fanta_limon, Fanta_naranja, CocaCola, Solo_Fanta_naranja, Solo_tonica, Solo_Fanta_Limon, Solo_CocaCola. También se han creado las salidas pinza, refresco_c2, listo_servir_c2, Solo_refresco_C2.

2.7) Lógica de la estación

Las entradas y salidas digitales que se han nombrado en el apartado anterior hay que conectarlas para que desempeñen la función que nos interesa. Esto se hace desde la lógica de la estación. La unión con los componentes inteligentes es la siguiente:

2.7.1) Ventosa

Como puede verse en la Figura 30, la ventosa se conecta al controlador 1 ya que este es el controlador del robot 1. La salida vacío del controlador 1 se conecta a la entrada vacio de la ventosa, de manera que la salida ventosa del controlador 1 puede activarse o desactivarse desde el código RAPID, lo que activará el componente inteligente. La salida vacuostato de la ventosa se conecta a la entrada vacuostato del controlador 1, de manera que cuando el componente inteligente coja o suelte una pieza, la entrada vacuostato del controlador cambiará de valor indicando que ha terminado la acción.

2.7.2) Pinza

Como puede verse en la Figura 30, la salida pinza del controlador 2 se conecta con la entrada pinza, de manera que desde el código RAPID se puede activar y desactivar la pinza para coger y soltar las piezas.

2.7.3) Accionador

También en la Figura 30 se puede ver que el Accionador_smart no tiene nada conectado a las entradas, ya que el propósito de este es cambiar las entradas de manera manual para pedir las

diferentes copas. Las salidas del accionador están conectadas a las diferentes entradas de los dos controladores. Estas entradas se gestionarán en el código RAPID para saber cuándo se pide una copa. Como puede observarse, las bebidas alcohólicas están conectadas al controlador 1 ya que el robot 1 gestionará este tipo de bebidas, mientras que al controlador 2 están conectadas las salidas de los refrescos.

2.7.4) Depósito

Como se puede observar nuevamente en la Figura 30, el deposito_smart no tiene nada conectado, ya que este se activa cuando un objeto corta su sensor, y no cuenta con entradas ni salidas digitales.

2.7.5) E/S Robots colaborativos

Aparte de las señales que se usan para los componentes inteligentes, hay otras 7 entradas y salidas digitales que se usan para sincronizar los dos robots. Estas 7 señales son vaso_bebida, cogiendo_cerveza, cerveza_lista_abrir, robot_1_libre, refresco, listo_servir, solo_refresco. En la Figura 30 se ve que existen esas 7 señales en el controlador 1 (con C1 al final de su nombre) y otras 7 en el controlador 2 (con C2 al final de su nombre). Estas señales funcionan de la siguiente manera: cuando un robot hace una acción en la que el otro robot tiene que esperar a que termine, el robot que hace la acción tiene una salida digital que se va a usar para indicar el final de esta acción; en el código RAPID se indicará que la señal cambie de valor cuando el robot termine la acción, mientras que el otro robot tendrá indicado en el código RAPID que espere hasta que la entrada que corresponde a esa acción cambie de valor, indicándole así que el otro robot ya ha terminado su tarea.

El funcionamiento de cada una de estas señales se explica más a fondo en el apartado del código RAPID.

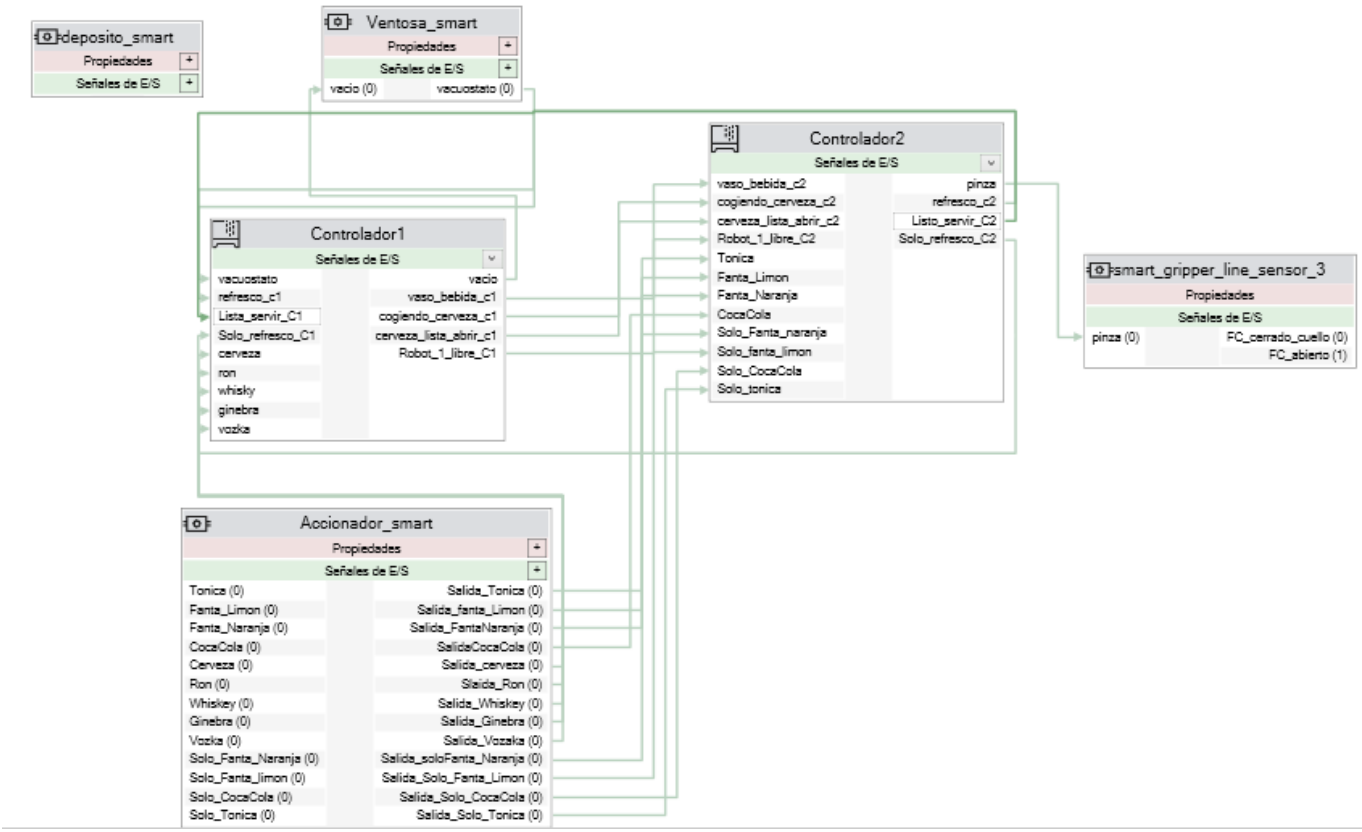

Figura 30.Lógica de la estación

2.8) Objetos de trabajo y puntos

Para crear las trayectorias por donde se van a mover los brazos robóticos primero hay que crear los puntos y los objetos de trabajo. Los objetos de trabajo son bases (o sistemas) de referencia que engloban todos puntos que asocien a ellos. De esta manera, si se quiere cambiar la posición de un modelo 3D con el que se esté trabajando, si se cambia la posición del objeto de trabajo, se cambiarán todos los puntos asociados a este y no habrá que volver a programarlos. Los objetos de trabajo que se han creado en este proyecto son los que se pueden ver en la Figura 31.

Dentro de cada uno de estos objetos de trabajo se encuentran los puntos que se han programado para crear las diferentes trayectorias. Las consideraciones que se han tenido en cuenta a la hora de crear los puntos son, en primer lugar y más importante, que hay que evitar puntos en los que el robot se encuentre en una posición de singularidad o cerca de ella, lo que se puede comprobar fácilmente ya que RobotStudio muestra errores al ejecutar la simulación si hay singularidades. En segundo lugar, hay que prestar especial atención a la orientación de la herramienta en cada uno de los puntos, ya que, si la orientación no es la correcta, a la hora

de formar las trayectorias se producirán movimientos no deseados o simplemente movimientos que el robot no puede ejecutar. Las posiciones de referencia del robot son dos puntos con una gran importancia donde el robot vuelve al acabar una acción. Otros puntos a destacar son los que comienzan su nombre con "ayuda_girar", ya que se usan para hacer una trayectoria redondeada cuando el robot tiene que hacer un giro pronunciado con su eje 1, ya que si no se pusieran estos puntos la trayectoria que crea el robot es una línea recta entre los dos puntos, que en muchas ocasiones hace que la trayectoria pase por el eje del robot y por lo tanto se salga de su volumen de trabajo.

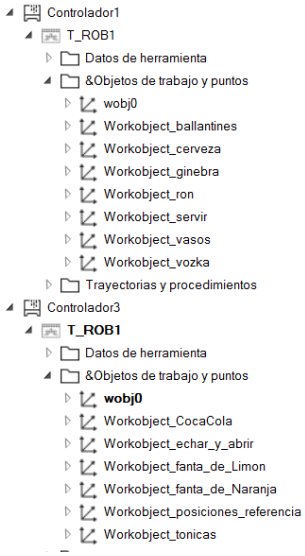

Figura 31. Objetos de trabajo

2.9) Trayectorias y código RAPID

Las trayectorias y el código RAPID van de la mano, ya que, al crear cada una de estas trayectorias, hay que hacer cambios en el código RAPID para que todo funcione de manera adecuada.

A continuación, se va a explicar cada una de las trayectorias y el código RAPID asociado a las mismas:

2.9.1) Trayectoria vaso

La trayectoria trayectoria_vaso se va a utilizar para coger los vasos, llenarlos de hielo y llevarlos hasta la zona entre los dos robots donde se van a preparar las copas. En la Figura 32 se muestran las instrucciones de la trayectoria. Las instrucciones MoveL se utilizan para mover el robot al punto que se indica en la instrucción, indicando la velocidad y precisión de cada uno

de los movimientos. En cada una de las trayectorias se ha reducido la velocidad y se ha aumentado la precisión cuando el robot se encuentra cerca de un punto conflictivo, es decir, puntos donde el robot tiene objetos cerca ya que tiene que acercarse para manipularlos, por lo que ajustar estos parámetros ayuda a evitar colisiones no deseadas.

En la Figura 33 se puede observar la trayectoria dentro del modelo. Esta se inicia en el punto de referencia del robot 1 y desde ahí se mueve hasta el punto de aproximación para coger el vaso. En ese momento el robot coge un vaso u otro dependiendo los que se hayan gastado y, utilizando una serie de condicionales (IF),se consigue guardar en las variables numéricas coger_vaso_x y coger_vaso_y las coordenadas del vaso que el robot ha de coger. Estas coordenadas tienen como origen el punto coger_vaso, el cual se encuentra en el primer vaso que se utiliza, y con la instrucción MoveLOffs se añaden las coordenadas de las variables numéricas al punto coger_vaso. El robot 1 coge el vaso que le corresponde y lo mueve hasta la máquina expendedora de hielos. Gracias a la instrucción WaitTime, se consigue que el robot 1 espere 2 segundos en el dispensador de la máquina expendedora de hielos, simulando que la maquina está sirviendo los hielos. Una vez ha terminado de servir los hielos lleva el vaso hasta el punto dejar_vaso y vuelve hasta la posición de referencia.

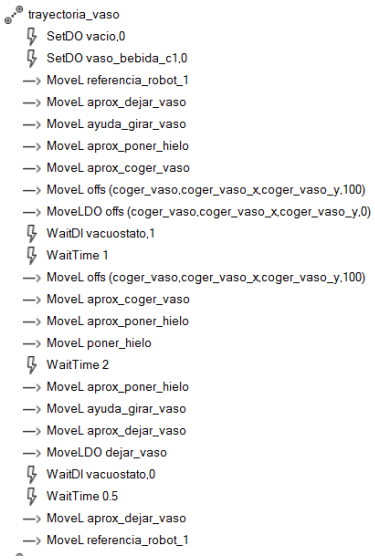

Figura 32. Trayectoria vaso

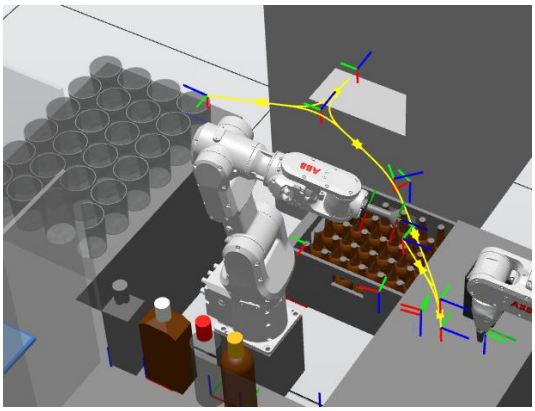

Figura 33. Trayectoria vaso modelo

2.9.2) Trayectoria bebidas alcohólicas

Para servir todos los tipos de bebidas alcohólicas, todas las trayectorias siguen el mismo procedimiento, por lo que solo se va a explicar detalladamente una de ellas. Las otras tres

trayectorias de bebidas alcohólicas son exactamente iguales, cambiando los puntos de referencia para coger la botella.

En la Figura 34 se muestran las instrucciones de la trayectoria utilizada para servir ginebra. La trayectoria comienza en el punto de referencia del robot 1 y, seguidamente, con la instrucción MoveL se mueve el robot a través de los diferentes puntos que se muestran en la Figura 34 llegando hasta la posición *coger_ginebra* donde la ventosa se conecta a la botella. El robot se vuelve a mover hasta la posición *aprox_poner_ginebra* y de ahí pasa a la posición *poner_ginebra*, momento en el que el robot 1 está sirviendo la bebida alcohólica. Con la instrucción WaitTime se hace que el robot permanezca 4 segundos en esa posición, tras lo cual, con la instrucción SetDo se modifica el valor de la entrada digital *vaso_bebida_c1* a 1, indicando al robot 2 que puede comenzar a servir el refresco. El robot 1 vuelve a mover la botella de ginebra a su posición original, se desconecta la ventosa de la botella y se mueve el robot a la posición de referencia del robot 1.

Por último, con la instrucción WaitDi se espera a que la entrada digital refresco_c1 tome el valor1, lo que indica al robot 1 que el robot 2 ha terminado de servir el refresco.

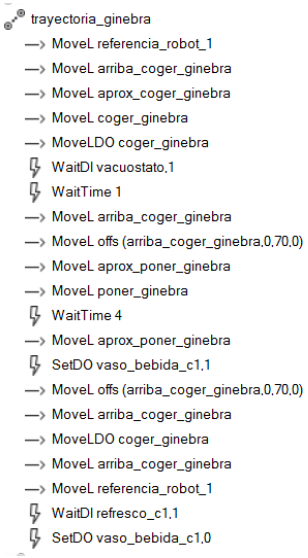

Figura 34. Trayectoria ginebra

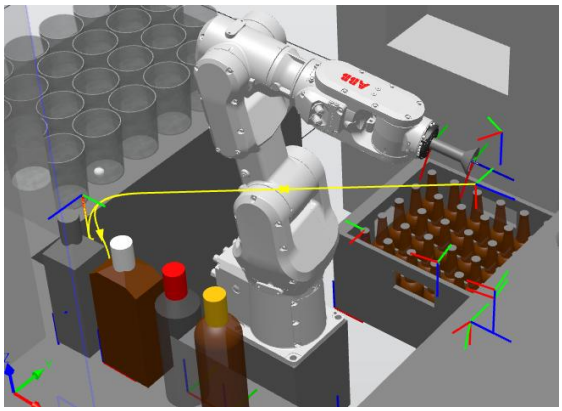

Figura 35. Trayectoria ginebra modelo

2.9.3) Trayectoria servir

Esta trayectoria se va a utilizar para llevar todas las bebidas terminadas a la zona del componente inteligente depósito, donde en la realidad se encontraría el cliente final. En esta trayectoria, aparte de reducir la velocidad en los puntos conflictivos como en otras

trayectorias, también se ha reducido la velocidad en toda la parte de la trayectoria donde el robot está moviendo la copa, ya que en ese momento el robot lleva la copa llena y cambios bruscos de velocidad y dirección pueden provocar que la copa derrame parte de su contenido.

La trayectoria comienza en la posición de referencia del robot 1, como puede verse en la Figura 36, y, después, con la instrucción MoveL se mueve el robot hasta la posición *dejar_vaso*. En ese momento se conecta la ventosa con la copa, el robot 1 se mueve a una velocidad reducida hasta la posición *servir_copa,* se desconecta la ventosa de la copa terminada y se mueve el robot hasta la posición de referencia del robot 1.

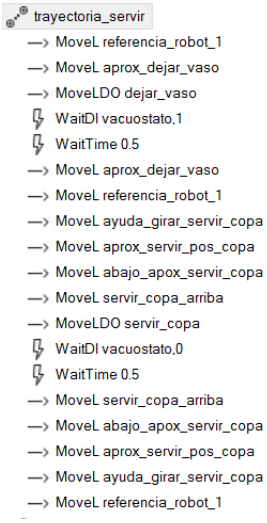

Figura 36. Trayectoria servir

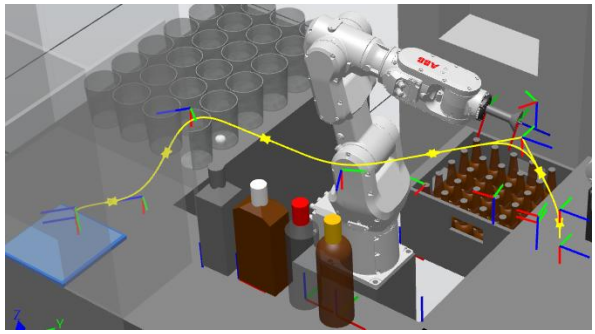

Figura 37. Trayectoria servir modelo

2.9.4) Trayectoria cerveza

Esta trayectoria se va a usar para que el robot 1 coja los botellines de cerveza de la caja y los lleve a una posición donde el robot 2 los abra. En primer lugar, con la instrucción SetDo se cambia el valor a la salida cogiendo_cerveza_c1 a 1, para indicar al robot 2 que se ha pedido una cerveza y que el robot 1 está cogiéndola. La trayectoria comienza en la posición de referencia del robot 1. Con la instrucción MoveL se mueve a la posición referencia_cerveza, donde se tiene que elegir a qué punto concreto irá el robot. Esto depende del número de cervezas que se hayan gastado, y para ello se usan las variables numéricas coger_cerveza_x y coger_cerveza_y. Con una serie de condicionales, se guardan en estas variables las coordenadas del botellín que el robot debe coger, con el origen de esas coordenadas en el punto original coger_cerveza. De esta manera con la función MoveLOffs se puede recorrer toda la matriz que forman los botellines de cerveza. Una vez el robot 1 se encuentra en la

posición del botellín que debe coger, conecta la ventosa al botellín y con la instrucción MoveL lo mueve hasta el punto dejar_cerveza. En los últimos puntos de esta trayectoria se utiliza un offset para modificar los puntos ligeramente y evitar singularidades, ya que en los puntos que se había creado la trayectoria, el robot pasaba por una singularidad. Modificando ligeramente las posiciones evitamos ese punto conflictivo. Una vez en esa posición se desconecta la ventosa del botellín y se regresa al punto de referencia del robot 1.

Utilizando la instrucción SetDo se cambia el valor a la salida digital *cerveza_lista_abrir_c1* a 1, indicando al robot 2 que el botellín de cerveza se encuentra en la posición donde el robot 1 lo tiene que dejar.

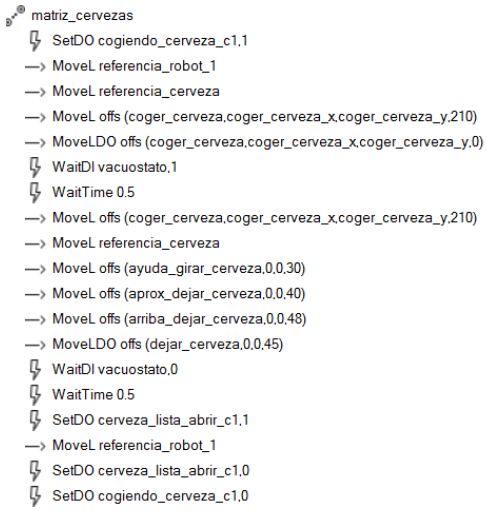

Figura 38. Trayectoria cerveza

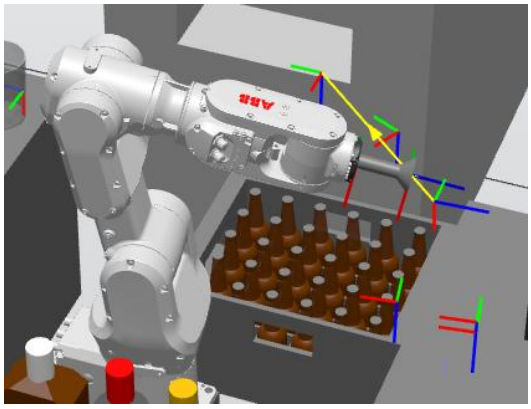

Figura 39. Trayectoria cerveza modelo

2.9.5) Trayectoria refresco

Para coger los cuatro tipos de refresco se han creado cuatro trayectorias. Como el funcionamiento de estas es muy similar entre las cuatro, ya que solamente cambian los puntos a los que se mueve el robot, solo se va a explicar una de ellas: la trayectoria del refresco Coca Cola, como puede verse en la Figura 40 y Figura 41.

En primer lugar, con la instrucción SetDo se pone el valor de la salida digital pinza a 0, asegurándonos que la pinza está abierta. Después, se mueve el robot con la instrucción MoveL a la posición referencia_refrescos_derecha (en el caso de la tónica y fanta de naranja, esta instrucción es referencia_refrescos_izquierda). Una vez el robot se encuentra en la referencia, de la misma manera que en los vasos y los botellines de cerveza, se guardan en unas variables numéricas ‘x’ e ‘y’ las coordenadas del refresco que tiene que coger el robot en esta ocasión.

Con la instrucción MoveLOffs y las variables numéricas, el robot se mueve a la posición del refresco que ha de coger, primero a un punto de aproximación con una altura de 210mm y luego al punto coger_refresco. Cuando el robot se encuentra en esa posición, se conecta la pinza al botellín de refresco. Una vez el botellín está conectado, el robot se mueve con la instrucción MoveLOffs a la posición de aproximación y, por último, vuelve a la posición de referencia.

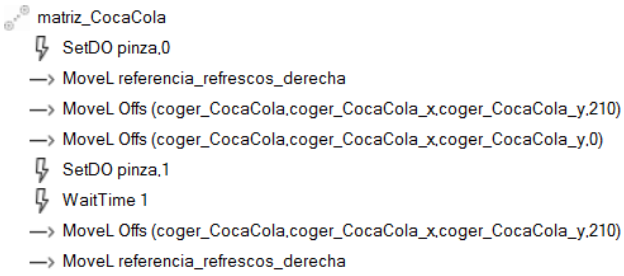

Figura 40. Trayectoria Coca Cola

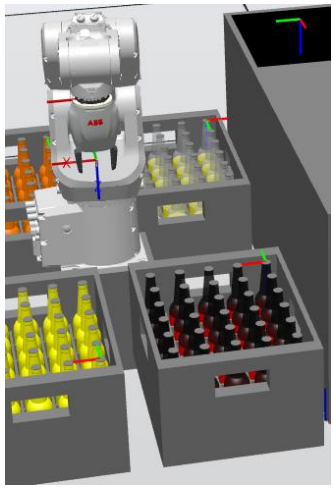

Figura 41. Trayectoria Coca Cola modelo

2.9.6) Trayectoria cambiar agarre y abrir

En esta simulación de RobotStudio existen tres trayectorias para abrir botellines: *abrir_cerveza*, *abrir_solo_refresco* y *cambiar_agarre_y_abrir*. Se va a explicar únicamente la última de estas, ya que las tres trayectorias son muy similares, aunque se indicarán las diferencias que existen entre las tres.

En el comienzo de la trayectoria nos encontramos con el botellín cogido por la parte superior, con la herramienta en posición vertical. Con la instrucción MoveL se le indica al robot que se mueva a la posición *dejar_botellin* y, una vez en esta posición, se desconecta la pinza del botellín, y se mueve el robot a la posición *coger_botellin*. La trayectoria *abrir_cerveza* empezaría en este punto ya que el botellín lo deja el robot 1 en esa posición.

Una vez el robot se encuentra en la posición *coger_botellin*, se conecta el botellín a la pinza en la zona del cuello, con la herramienta en posición horizontal. En ese momento se mueve el botellín hasta el abrebotellas donde se abre y se lleva hasta la posición *aprox_echar*. En el caso de la trayectoria *abrir_cerveza*, la trayectoria termina aquí, llevando la cerveza a la posición *dejar_vaso,* donde el robot 1 la cogerá para servirla.

En el caso de la trayectoria *abrir_solo_refresco* y *cambiar_agarre_y_abrir*, la trayectoria continúa sirviendo el refresco. Para ello se mueve a la posición *echando_refresco*, momento en el que el robot se encuentra echando refresco en el vaso. Gracias a la instrucción WaitTime se esperan 2 segundos y se vuelve a la posición *echar_refresco*. Se esperan otros 2,5 segundos y se vuelve a la posición *echando_refresco*, donde se vuelven a esperar 3,5 segundos. Esto se hace de esta manera ya que cuando se echa una bebida carbonatada a un vaso esta tiende a hacer mucha espuma. De esta manera, el robot empieza a echar el refresco y, cuando las burbujas suben, para momentáneamente de servir para después volver a servir la bebida hasta que esta se acaba.

Una vez se ha terminado de echar el refresco, se le indica al robot que se mueva a la posición *tirar_refresco_papelera*, en ese momento se desconecta la pinza del botellín para que caiga en la papelera, por último, se mueve el robot a la posición de referencia del robot 2.

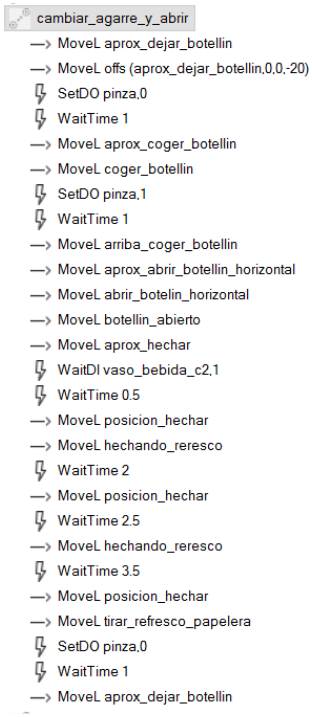

Figura 42. Trayectoria cambiar agarre y abrir

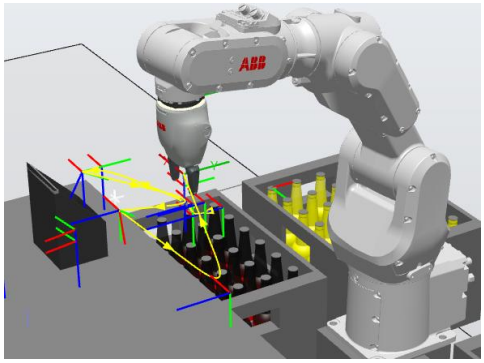

Figura 43. Trayectoria cambiar agarre y abrir modelo

2.9.7) Función main código RAPID

La función main es la función principal del programa, la cual se ejecuta de manera continua. La función main, en el caso de esta simulación, se ha usado para llamar a las diferentes trayectorias dependiendo del tipo de bebida que pida el cliente. En esta simulación se cuenta con dos funciones main, ya que cada controlador tiene su código independiente.

En este apartado se va a ver como se ha implementado la función main para cada tipo de bebida que el sistema robótico puede servir.

2.9.7.1) Copas

En el caso de la copa, se va a mostrar el funcionamiento con la mezcla de ginebra y tónica, pero funciona de forma análoga para todo tipo de combinaciones de bebida con refresco. En este caso, cuando desde el accionador_smart se activen las señales ginebra y tónica, estas tomarán un valor de 1, por los que los dos controladores entrarán dentro del condicional IF. En los dos códigos me aseguro de poner a 0 las variables que se van a utilizar, y en el controlador se cambia a 1 la variable refresco_c2, para que el robot 1 compruebe que se ha pedido un refresco.

En el controlador 1 se ejecuta la trayectoria vaso, el robot va a por un vaso, lo llena de hielo y lo coloca en la barra, como se ha mostrado en uno de los apartados anteriores. Al terminar esa trayectoria, el robot inicia la trayectoria ginebra, momento en el que va a por la botella de ginebra, la sirve en el vaso y vuelve a dejarla. Entonces, el robot 1 queda a la espera de que el robot 2 active la entrada listo_servir_C1.

Al mismo tiempo, en el controlador 2 se ejecutan las trayectorias *girar_a_ref_izquierda*, con la que el robot se posiciona en el punto de referencia izquierdo, *matriz_tonicas*, con la que el robot coge la tónica correspondiente, y, por último, *cambiar_agarre_y_abrir*, en la que el robot tiene que esperar a que la variable vaso_bebida_c2 sea igual a 1, como se ha explicado en el apartado correspondiente a esta trayectoria. Esta entrada es modificada por el robot 1 cuando este acaba la *trayectoria_ginebra*, y en ese momento el robot termina la trayectoria *cambiar_agarre_y_abrir*. Cuando el robot 2 termina de servir el refresco, cambia el valor de la salida listo_servir_c2 a 1 y, en ese momento, el robot 1, comienza a ejecutar la trayectoria *trayectoria_servir*. Por último, la variable robot_1_libre se pone a 1, indicando que este queda libre, y la variable listo_para_servir_c2 se pone a 0,ya que se necesita que tenga ese valor la próxima vez que se ejecute.

```
IF ginebra=1 THEN
    !lo uso para indicar que el robot se esta usando y no deje echar solo refesco
    setDo Robot_1_libre_C1,0;
    trayectoria_vaso;
    trayectoria_ginebra;
    waitDi Lista_servir_C1,1;
    trayectoria_servir;
    !vuelvo a indicar que el robot esta libre
    SetDo Robot_1_libre_C1,1;
ENDIF
```

Figura 44. Código bebida alcohólica controlador 1

```
IF Tonica=1 THEN
SetDo refresco_c2,0;
!me aseguro que la variable este a 0
SetDO Listo_servir_C2,0;
!activo la variable para que el otro robot compruebe si se ha pedido un refresco
SetDO refresco_c2,1;
girar_a_ref_izquierda;
matriz_tonicas;
cambiar_agarre_y_abrir;
!reinicio la variable cuando ya he acabado de hechar el refresco
SetDO refresco_c2,0;
!activo la variable para que el otro robot sirva la copa
SetDO Listo_servir_C2,1;
WaitTime 1;
SetDo Listo_servir_C2,0;
ENDIF
```

Figura 45. Código refresco controlador 2

2.9.7.2) Cerveza

A diferencia de la copa y el refresco, el procedimiento de servir la cerveza el lineal, es decir, no hay ningún momento donde los dos robots estén trabajando en paralelo. El procedimiento es el siguiente: el robot1 coge el botellín de cerveza de la caja, el robot 2 lo abre y el robot 1 lo sirve al cliente.

Cuando el cliente activa la entrada cerveza con el accionador_smart, en el controlador 1 se entra dentro del condicional IF. En ese momento se ejecuta la trayectoria *matriz_cervezas* y, cuando esta se inicia, tal y como se ha visto en el apartado correspondiente a la trayectoria matriz cervezas, se activa la salida cogiendo_cerveza_C1, por lo que el controlador 2 entra dentro del condicional IF. Cuando el robot 1 termina de ejecutar la trayectoria *matriz_cervezas*, el robot 2 inicia la trayectoria *abrir_cerveza*, el robot 1 queda a la espera de la señal Lista_servir_C1, la cuales activada por el robot 2 cuando termina la trayectoria *abrir_cerveza*, momento en el que se ejecuta la *trayectoria_servir*.

```
IF cerveza=1 THEN
    matriz_cervezas;
    WaitDI Lista_servir_C1,1;
    trayectoria_servir;

ENDIF
```

Figura 46.Código cerveza controlador 1

```
IF cogiendo_cerveza_c2=1 THEN
    abrir_cerveza;
ENDIF
```

Figura 47.Código cerveza controlador 2

2.9.7.3) Refresco

En este caso se ha tomado como ejemplo el procedimiento para la tónica, pero funciona de forma análoga para el resto de refrescos. Cuando se activa la salida solo_tonica desde el accionador_smart y el robot 1 está libre, se inicia el proceso para servir refresco. En primer

lugar, se activa la salida solo_refresco_C2, lo que hace que en el controlador 1 se entre dentro del condicional IF y se ejecute la trayectoria vaso. Mientras el robot 1 va a por un vaso y lo llena de hielo, el robot 2 ejecuta las trayectorias girar_a_ref_izquierda, con la que el robot se posiciona en el punto de referencia izquierdo, matriz_tonicas con la que el robot coge la tónica correspondiente y, por último, Abrir_solo_refresco, con la que el robot abre y sirve el refresco en el vaso. Cuando el robot 2 ha terminado, activa la salida Listo_servir_c2, señal para que el robot 1 se active y ejecute la trayectoria trayectoria_servir. Por último, en el controlador 2 se desactivan las salidas listo_servir_C2 y Solo_refresco_C2 para volverlas a utilizar cuando sea necesario.

```
IF Solo_refresco_C1=1 THEN
    trayectoria_vaso;
    waitDi Lista_servir_C1,1;
    trayectoria_servir;

ENDIF
```

Figura 48.Código solo refresco controlador **1**

```
IF Solo_tonica=1 THEN
    IF Robot_1_libre_C2=1 THEN
    !me aseguro que la variable este a 0
    SetDO Listo_servir_C2,0;
    !Solo_refresco la uso para que el tro robot vaya a por el vaso
    setDo Solo_refresco_C2,1;
    girar_a_ref_izquierda;
    matriz_tonicas;
    Abrir_solo_refresco;
    SetDO Listo_servir_C2,1;
    WaitTime 1;
    !reinicio las variables para poder volver a usarlas luego
    SetDo Listo_servir_C2,0;
    SetDO Solo_refresco_C2,0;
    ENDIF
ENDIF
```

Figura 49. Código solo refresco controlador 2

3) Modelo JaamSim

3.1) Estudio estadístico

Con el objetivo de aumentar la fidelidad del modelo de simulación de sistemas dinámicos, se ha realizado un estudio estadístico de los tiempos que emplea un trabajador en servir las copas en un bar real. Este estudio estadístico se ha llevado a cabo a lo largo de los meses de septiembre a diciembre del 2023.En él se han recopilado de manera manual los tiempos que tarda un camarero en hacer frente a los diferentes pedidos.

3.1.1) Locales

Con el objetivo de aumentar la aleatoriedad de las muestras, se han tomado en varios locales y en diferentes días, aumentando así el número de camareros que han intervenido en la toma de muestras.

Los locales donde se han recopilado muestras son: Pub Scream- Teruel, Pub Wateque- Teruel, Pub Davidue- Teruel, Bar Escalón- Mora de Rubielos, Barra comisión de fiestas- Cedrillas y barra comisión de fiestas- Gúdar - Javalambre .

3.1.2) Toma de las muestras

En primer lugar, se tiene que especificar el tipo de muestras que se van a tomar, en este caso se va a tener en cuenta cuando un camarero sirva cervezas, refrescos y copas, ya que como se ha visto en la parte del modelo de RobotStudio, son los tres tipos de bebidas que puede servir el modelo robótico.

A la hora de pedir los clientes no siempre lo hacen de uno en uno, hay ocasiones en la que un cliente pide dos o más tipos de bebidas para varias personas, en el caso de este estudio se han tomado los tiempos de las personas que piden solas y cuando dos personas piden juntas. Los principales motivos por lo que no se han tomado muestras cuando un cliente pide tres o más bebidas son:

- La dificultad que entraña cronometrar tantos tipos de pedidos a la vez, hace que la precisión de la medición se reduzca sustancialmente debido al error humano, la dificultad de una medición precisa aumenta ya que cada camarero sigue un procedimiento distinto a la hora de atender varios pedidos a la vez.

-Se ha observado que el número de pedidos en los que un cliente pide tres o más bebidas son reducidos si lo comparamos con la totalidad de pedidos, por lo cual, aunque no se tengan en cuenta, el modelo seguirá conservando su precisión.

3.1.2.1) Copas

Para de cronometrar los tiempos que tardan en servir las copas, se ha tomado el tiempo que tarda el cliente en pedir, el tiempo que tarda el camarero en coger el vaso y servir los hielos, el tiempo en servir la bebida alcohólica, el tiempo en servir el refresco y por último el tiempo que tarda en cobrar la copa. A lo largo de estos meses se han cronometrado 57 muestras.

3.1.2.2) Refrescos

Para cronometrar los tiempos que tardan en servir los refrescos, se ha tomado el tiempo que tarda el cliente en pedir, el tiempo que tarda el camarero en coger el vaso y servir los hielos, el tiempo en servir el refresco y por último el tiempo que tarda en cobrar el refresco. A lo largo de estos meses se han cronometrado 10 muestras.

3.1.2.3) Cervezas

Para cronometrar los tiempos que tardan en servir las cervezas, se ha tomado el tiempo que tarda el cliente en pedir, el tiempo que tarda el camarero en coger y abrir la cerveza y por último el tiempo que tarda en cobrar la cerveza. A lo largo de estos meses se han cronometrado 35 muestras.

3.1.2.4) Dos copas

Para cronometrar los tiempos que tardan en servir las copas, se ha tomado el tiempo que tarda el cliente en pedir, el tiempo que tarda el camarero en coger dos vasos y servir los hielos, el tiempo en servir las bebidas alcohólicas, el tiempo en servir los refrescos y por último el tiempo que tarda en cobrar las copas. A lo largo de estos meses se han cronometrado 68 muestras.

3.1.2.5) Copa y cerveza

Para cronometrar los tiempos que tardan en servir una copa y una cerveza, se ha tomado el tiempo que tarda el cliente en pedir, el tiempo que tarda el camarero en coger el vaso y servir los hielos, el tiempo en servir la bebida alcohólica, el tiempo en servir el refresco, el tiempo que tarda para coger y abrir una cerveza y por último el tiempo que tarda en cobrar la copa. A lo largo de estos meses se han cronometrado 31 muestras.

3.1.2.6) Dos cervezas

Para cronometrar los tiempos que tardan en servir las cervezas, se ha tomado el tiempo que tarda el cliente en pedir, el tiempo que tarda el camarero en coger y abrir las cervezas y por último el tiempo que tarda en cobrar las cervezas. A lo largo de estos meses se han cronometrado 38 muestras.

3.1.2.7) Dos refrescos

A lo largo de estos meses no ha ocurrido en ninguna ocasión que un cliente pida dos refrescos, por lo que esa posibilidad no se ha modelado.

3.1.3) Toma de las muestras RobotStudio

Para tomar las muestras en el modelo de RobotStudio se han cronometrado los tiempos que tardan los robots en atender los diferentes pedidos, en este caso solo se han tomado 20 muestras de cada uno de los tipos de pedidos que los robots son capaces de preparar, ya que

la variación de estos tiempos se debe en mayor medida al error humano, aunque bien es cierto que los tiempos varían ligeramente de unos pedidos a otros, ya que los vasos y botellines se encuentran en posiciones diferentes a medida que se van gastando, lo que hace al robot cambiar sus trayectorias ligeramente.

3.2) Modelo JaamSim 1 camarero

3.2.1) Distribuciones de probabilidad

Para aumentar la aleatoriedad en el modelo se van a usar distribuciones de probabilidad. Como se puede ver en la Figura 50 yFigura 51, se van a utilizar dos tipos de distribuciones de probabilidad: las distribuciones normales se usan para que el tiempo de servicio de cada uno de los servidores varíen cada uno de los clientes, indicando el tiempo mínimo, máximo y medio de cada una de las acciones que se han visto en el apartado del estudio estadístico; y las distribuciones discretas se utilizan para indicar el porcentaje de clientes que demanda cada tipo de bebida, el cual se ha obtenido calculando la cantidad de clientes que se han cronometrado pidiendo cada tipo de bebida con respecto al total de las muestras que se han tomado. De esta manera se ha obtenido que, en el bar real, el 42,67% de las veces pide un solo cliente y el 57,33% de las veces piden dos clientes juntos. A su vez, de las veces que pide un solo cliente, el 58,88% de las veces se pide una copa, el 34,31% de las veces se pide una cerveza y el 9,81% se pide un refresco. De las veces que piden dos clientes juntos, el 49,63% de las veces piden dos copas, el 27,74% de las veces piden dos cervezas, el 22,63% de las veces piden una copa y una cerveza, y el 0% de las veces piden dos refrescos.

Para el bar robótico se ha tenido en cuenta la cantidad de cada tipo de bebida que se ha pedido en el bar real, pero en este caso el robot no puede realizar dos pedidos a la vez, por lo que se han dividido los pedidos de dos personas juntas en pedidos individuales, por ejemplo cada uno de los pedidos dobles de cerveza y copa se ha dividido y se ha sumado un pedido a las cervezas pedidas por una persona y otro a las copas, con esta división se obtiene que el 59,57% de los clientes pide una copa, el 2,66% pide refrescos y el 37,77% pide cerveza.

Figura 50. Distribuciones de probabilidad modelo bar real

Figura 51. Distribuciones de probabilidad modelo bar robótico

3.2.2) Generador de entidades

Para generar todas las entidades se utiliza un generador de entidades, para que funcione como se desea se necesitan varios bloques más, tal y como se puede observar en la Figura 52. En primer lugar, se necesita una entidad, en este caso su nombre es "*pedido*", cada una de las entidades generadas representan un pedido diferente. Se quiere que el generador solo genere entidades unos días y horas en específico, tal y como sucede en un bar real. Para simular los días se utilizan dos bloques "ExpressionEntity", en uno de ellos se divide el tiempo de la simulación, el cual está en horas, entre 24 para obtener el número de días de la simulación, con el operador "%" se obtiene el resto de una división, en este caso se divide entre 7 y el resto es el día de la semana en la que nos encontramos, en el otro bloque se utiliza el operador "%" para obtener el resto de una división entre 24, de esta manera se obtiene la hora del día en la que se encuentra la simulación.

En un "ExpressionThreshold" se indican los días y horas que se deben generar pedidos, de manera que se pueden simular los horarios de apertura de un bar. En el caso del ocio nocturno en Teruel, los horarios de apertura son jueves, viernes y sábado de 23:00h a 4:00.

El bloque "Time series" se utiliza para generar el número de pedidos. El número de pedidos que atiende cada camarero se ha supuesto observando varios factores:

-El tiempo medio que tardan los camareros en realizar un pedido ronda 1 minuto, por lo que un camarero no podrá atender más de 60 pedidos en una hora.

-Según mi experiencia personal, el número de pedidos es bajo al principio de la noche, este va subiendo y alcanza el máximo a mitad de la noche, de ahí en adelante disminuyen los pedidos hasta que llega la hora de cierre.

En el caso del bar real, los pedidos se han distribuido de la siguiente manera: de 23:00 a 24:00 25 pedidos, de 24:00 a 01:00 40 pedidos, de 01:00 a 02:00 52 pedidos, de 02:00 A 03:00 45 pedidos y de 03:00 a 04:00 40 pedidos. Esto hace un total de 202 pedidos a lo largo de la noche.

En el caso del bar robótico debido a que el robot no puede atender dos pedidos a la vez, para servir el mismo volumen de bebidas se necesitan más pedidos, la manera de calcular esos pedidos es multiplicar por dos el porcentaje de bebidas que las piden 2 personas, tras este cálculo el número de pedidos del robot queda distribuido de la siguiente manera: de 23:00 a 24:00 39 pedidos, de 24:00 a 01:00 63 pedidos, de 01:00 a 02:00 82 pedidos, de 02:00 A 03:00 70 pedidos y de 03:00 a 04:00 63 pedidos. Esto hace un total de 317 pedidos a lo largo de la noche.

Para que estos pedidos no lleguen de manera equiespaciada en el tiempo se utiliza una distribución de probabilidad "NonStatExponentialDist", la cual permite generar instantes aleatorios de tiempo en intervalos prefijados con un número de entidades (pedidos, en este caso) a generar dado, lo que aumenta el realismo de la simulación.

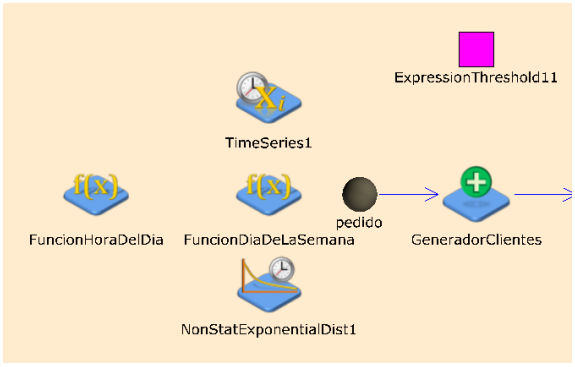

Figura 52. Generador de entidades bar real

3.2.3) Modelo bar real

Como se puede observar en la Figura 53, cada parte del proceso de pedido de una bebida se modela como un servidor con un tiempo de servicio asociado que depende de la tarea a realizar (camareros, máquinas y robots haciendo una tarea pueden modelarse como servidores). En este caso, primero, el generador de entidades genera los pedidos, los cuales van al servidor "Esperar turno", donde esperan en la cola del servidor a que el camarero esté libre y pueda atender el próximo pedido. Para simular esto, cada servidor tiene asociado un "ExpressionThreshold"(expresión condicional de permiso de acceso a un servidor), que

comprueba que la longitud de la cola del servidor sea igual a cero y que el servidor no se encuentre trabajando, de esta manera cuando el camarero este ocupado el bloque "ExpressionThreshold" correspondiente al servidor activo se activara, indicando que el camarero no está libre. Todos los bloques "ExpressionThreshold" se añaden al servidor "Esperar turno", por lo que, de esta manera, si alguno se activa, detendrá el servidor y los demás pedidos tienen que esperar en la cola del servidor "Esperar turno". Esta parte del modelo funciona de la misma manera en el bar robótico.

Cuando termina la espera en el servidor esperar turno, la entidad pasa por el servidor "Pedir bebida". Al avanzar se encuentra con un bloque "Branch", el cual sirve de forma general para tomar uno de los posibles caminos de salida en función del valor de una condición, donde se aplica la probabilidad de ser una o dos personas.Sea cual sea la dirección que tome el pedido, se encuentra con otro bloque "Branch" donde se aplica la probabilidad de pedir cada tipo de bebida. Cuando la entidad se encuentra en el tipo de bebida elegido, pasa por una serie de servidores, los cuales tienen asociado como tiempo de servicio las distribuciones de probabilidad normales que se han explicado con anterioridad. Una vez pasa por todos los servidores correspondientes a cada tipo de bebida, pasa al servidor "Cobro de la copa", el cual es un servidor común para todos los tipos de bebida. Cuando pasa dicho servidor, todos los bloques "ExpressionThreshold" quedan abiertos y el servidor "Esperar turno" deja que pase un nuevo pedido.

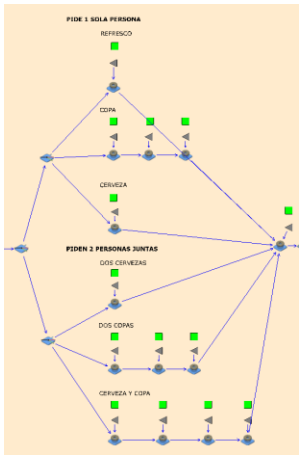

Figura 53. Servidores bebidas modelo bar real

3.2.4) Modelo bar robótico

El funcionamiento de los bloques "ExpressionThreshold" es el mismo al explicado en el modelo de bar real. Cuando los robots quedan libres, la entidad avanza a un bloque "Branch" donde se

aplica la probabilidad que hay para elegir cada una de las copas, en este momento existe una diferencia con el modelo de bar real, como el modelo de RobotStudio utiliza dos brazos robóticos en los pedidos refresco y copa existen tareas en paralelo, donde el robot 1 y el robot 2 están trabajando al mismo tiempo. Esto se modela gracias a un bloque "Duplicate", el cual separa la entidad entrante en dos, cada una fluye por los servidores que le corresponden y, cuando las dos han terminado, con un bloque "Combine" las dos entidades se fusionan y vuelven a formar una sola entidad. Tal y como pasaba en el modelo de bar real, los tiempos de servicio de los diferentes servidores están regidos por distribuciones normales de probabilidad.

Los tiempos de servicio de los servidores "Pedir bebida robot" y "Pagar robot" se rigen concretamente por las distribuciones normales "NDPedirBebida" y "NDCobroDeLaCopa",las cuales corresponden al bar real. Se ha decidido usar estas distribuciones normales ya que el modelo de RobotStudio no simula estos aspectos y se ha considerado que el tiempo que se tarda en pedir y cobrar la bebida depende en gran medida del cliente, por lo que, si se implementara un sistema para pedir y cobrar, los tiempos serían similares.

Otra diferencia con el modelo de bar real es que el servidor "Pagar robot" se encuentra antes de servir la copa, ya que he considerado que el robot no debe servir la copa hasta que el cliente la pague.

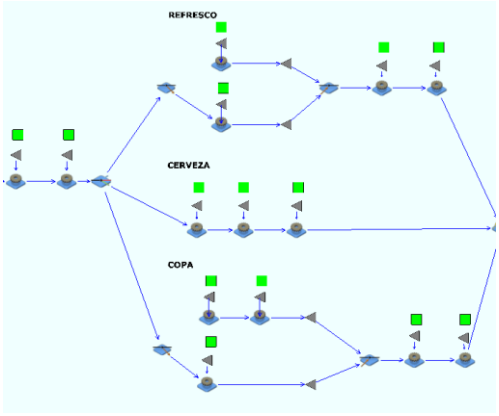

Figura 54. Servidores bebidas modelo bar robótico

3.2.5) Recopilación de datos

El objetivo principal del modelo en JaamSim es obtener datos que comparen de manera directa los dos tipos de bares. En primer lugar, se va a configurar la simulación para ajustarla a

las características que se están buscando, estableciendo el tiempo de duración de la simulación en 8760 h (equivalente a un año).

Para poder cuantificar los tiempos que tarda cada cliente en realizar cada una de las acciones se han incluido varios atributos en la entidad:

-**Inicio cola:** toma como valor el tiempo de la simulación en el momento en que la entidad llega a la cola.

-Final cola: se usa para calcular el tiempo que ha estado la entidad en la cola.

-Inicio pedir: toma como valor el tiempo de la simulación en el momento en el que la entidad inicia el pedido.

-Servido: se usa para calcular el tiempo que se ha empleado en servir la bebida.

-Tiempo total: se usa para calcular el tiempo total desde que ha llegado a la cola hasta que termina el pedido.

Para cambiar el valor de los atributos de la entidad se hace uso de los bloques "Assign". También se opera con ellos para obtener el tiempo que ha pasado en la cola restando el tiempo "inicio cola" al tiempo total de la simulación, y se almacena en el atributo "Final cola". Para obtener el tiempo que se ha empleado en servir la bebida se resta el tiempo "inicio pedir" al tiempo total de la simulación y se almacena en "servido". Por último, para guardar el tiempo total se suma "servido" y "final cola" y se almacena en "Tiempo total".

Con estos tiempos guardados en los atributos de las entidades se han usado los bloques "Statistics" para gestionar los datos. De estos bloques se ha obtenido la media de tiempo que se tarda en servir una bebida, la media de tiempo que las entidades pasan en la cola y la desviación estándar de los tiempos que tardan en servir una bebida. El atributo "Tiempo total" se ha usado como condición en un bloque "Branch" en el que se evalúa si los clientes están satisfechos o no. Para ello, se ha considerado que si el tiempo total es superior a 8 minutos los clientes no estarán satisfechos.

Para analizar y comparar las prestaciones de cada tipo de bar (real vs. robótico), los datos de interés que se han utilizado son: el porcentaje de clientes satisfechos, tiempo medio que se tarda en servir los pedidos, tiempo medio que pasan los clientes en la cola, tiempo medio total desde que el cliente llega a la cola hasta que se le ha servido la bebida, tiempo total máximo, desviación estándar del tiempo en servir y desviación estándar del tiempo total.

Como se puede observar en la Figura 55 y Figura 56, utilizando bloques de texto se muestran por pantalla en tiempo real los resultados obtenidos en la simulación.

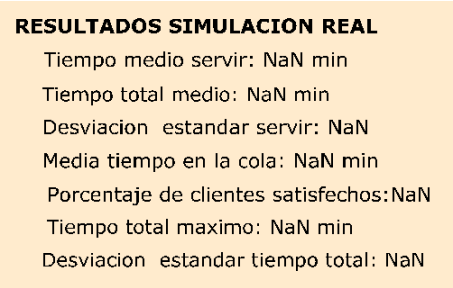

Figura 55. Resultado en tiempo real simulación bar real (aparecen NaN porque no se ha iniciado la simulación)

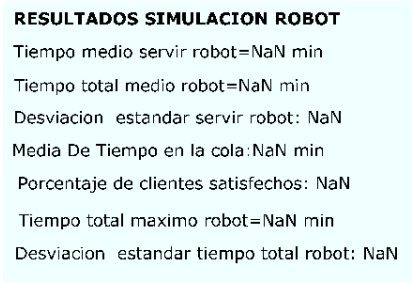

Figura 56. Resultados en tiempo real simulación bar robótico (aparecen NaN porque no se ha iniciado la simulación)

3.3) Modelo JaamSim 3 camareros

Como se verá en el apartado sobre la evaluación de los resultados obtenidos, el modelo de JaamSim con un camarero presenta varios problemas, para solucionar dichos problemas y aumentar el realismo de la simulación, se ha decidido crear un modelo con más camareros.

Para elegir el un número adecuado de camareros se va a tener en cuenta el testimonio del dueño del Pub Scream – Teruel, el cual me indico que, en un local con 155 personas de aforo máximo, cuenta con tres camareros. El nuevo modelo contará por tanto con tres camareros reales atendiendo la diversidad de pedidos, su rendimiento se comparará con tres conjuntos de camareros robóticos.

Para realizar este modelo se va a partir del modelo de un solo camarero, para simular varios camareros se va a hacer uso de los bloques de recursos compartidos. Los bloques utilizados son:

Resourceunit: Un único recurso que puede ser capturado o liberado por un grupo de recursos.

Resource Pool: Conjunto de recursos únicos.

Seize: Se apodera de una o más unidades de un recurso.

Release: Libera una o más unidades de un recurso.

En este caso cada "*Resourceunit*" representa un camarero, estas unidades están agrupadas dentro de un "*Resource Pool*", cada vez que llegue un cliente al bloque "*Seize*" este comprueba si hay alguna "*Resourceunit*" liberada, si es así la "*Resourceunit*" se une a la entidad pedido y avanza, simulando que el camarero está sirviéndole la bebida. Si en el momento que una

entidad llega no hay ninguna "*Resourceunit*" liberada significa que todos los camareros están ocupados, por lo cual, la entidad pasa a la cola de espera del "*Seize*". Cuando se termina un pedido las entidades con el recurso atraviesan el bloque "*Release*", liberando la "*Resourceunit*", lo que simula que el camarero vuelve a estar libre y puede atender a nuevos pedidos.

Para que este modelo funcione de manera correcta, todos los servidores del modelo de un camarero se han sustituido por bloques "*EntityConveyor*" (bloques que simulan una cinta transportadora) ya que un servidor no puede atender a dos entidades al mismo tiempo, mientras que por la "*EntityConveyor*" pueden pasar varias entidades a la vez, para que los bloques "*EntityConveyor*" cumplan la misma función que los servidores, las distribuciones normales que se añadían en el "*Time Service*" del servidor, se han añadido en el "*Travel Time*" de la "*EntityConveyor*".

Se han eliminado todas las colas de los servidores ya que las "*EntityConveyor*" no necesitan colas para funcionar, también se han eliminado todos los bloques "*ExpressionThreshold*" asociados a los servidores, ya que el control de paso de las entidades lo realiza el bloque "*Seize*".

Por último, el número de pedidos que se reciben a lo largo de cada hora se ha multiplicado por 3, para mantener la misma demanda en cada uno de los camareros.

3.4) Evaluación de los resultados obtenidos

Para obtener una cantidad elevada de datos se ha modificado los parámetros de la simulación, la duración de la simulación se ha establecido con una duración de 8760 horas lo que equivale a un año, en la sección "Multipleruns" se modificado el número de réplicas a 15 con lo que se en total se simularán 15 años de funcionamiento, para que todas las réplicas sean independientes y la significatividad de los resultados sea mayor, se ha modificado la semilla global en cada una de las réplicas.

Los datos obtenidos se encuentran en el Anexo II, en ellos se pueden observar las diferencias entre los dos tipos de bares:

-En primer lugar, para el tiempo medio que tarda en servir un pedido, el camarero real es de 1,06min en el modelo de 1 camarero y 1,08min en el modelo de 3 camareros, mientras que el camarero robótico invierte 0,80min en el modelo de 1 camarero y 0,80min en el modelo de 3 camareros. Como se puede observar los dos modelos arrojan tiempos casi idénticos. Lo cual es

lógico ya que el tiempo que tarda cada camarero en servir la copa no depende del número de camareros del bar. Por otro lado, la comparativa entre el bar real y bar robótico es muy positiva, ya que se ha conseguido mejorar el tiempo medio que se tarda en servir una bebida lo que invita a pensar que es posible mejorar las prestaciones de un bar real.

-La desviación estándar en el tiempo que se tarda en servir un pedido es de 0,28min en el modelo de 1 camarero y 0,28min en el modelo de 3 camareros, mientras que el camarero robótico la desviación es de 0,14min en el modelo de 1 camarero y 0,14min en el modelo de 3 camareros. De la misma manera que en tiempo medio que se tarda en servir los tiempos entre modelos son idénticos. Comparando el bar real con el robótico, este último vuelve a obtener mejores resultados, como era de esperar la dispersión de los tiempos que tarda el robot en servir las bebidas es menor, ya que el robot siempre realizada los mismos movimientos, mientras que los camareros reales son menos constantes.

-La media de tiempo que pasa la gente esperando en la cola es de 2,16min en el modelo de 1 camarero y 0,82min en el modelo de 3 camareros, mientras que el camarero robótico invierte 4.96min en el modelo de 1 camarero y 2,80min en el modelo de 3 camareros. En este caso los tiempos sufren una gran variación de un modelo a otro, eso se deba a que, aun manteniendo la misma carga individual de trabajo de cada camarero, el modelo de 3 camareros es capaz de afrontar de una mejor manera momentos puntuales de cargas altas de trabajo, reduciendo así los tiempos de espera en la cola, es por este motivo que considero que el modelo de 3 camareros se asemeja más a la realidad. En cuanto a la comparativa entre el bar real y el robótico se puede observar que el bar real a pesar de tener un tiempo de servicio más alto, tiene unos tiempos de espera mucho más cortos, esto se debe a que el camarero real puede servir más de una copa a la vez, lo que hace que el número de pedidos total sea inferior, esto se traduce en menos tiempo empleado en pedir la bebida y cobrarla, lo que provoca que menos clientes se acumulen en la cola.

-El tiempo total medio en el modelo de 3 camareros es de 1,90min mientras que en el bar robótico es de 3.60min. A pesar de que el tiempo de servicio del bar robótico es inferior al bar real, esto no compensa el número de pedidos extra que el robot tiene que afrontar, el tiempo máximo total del bar robótico llega hasta 35,81min en el modelo de 1 camarero y 20,47 en el modelo de 3 camareros, mientras que el bar real mantiene unos tiempos máximos de 26.22min en el modelo de 1 camarero y 13,73 en el modelo de 3 camareros, estos tiempos aunque no llegan a ser desmesurados, sí que indican que existe un mayor cuello de botella en el modelo de bar robótico.

-Por último, el porcentaje de clientes satisfechos que se ha obtenido es del 92,90% en el modelo de 1 camarero y 99,36% en el modelo de 3 camareros, mientras que el camarero robótico ha obtenido un 73,70% en el modelo de 1 camarero y un 87,20% en el modelo de 3 camareros. Como era de esperar, a la vista de los datos anteriores, el bar real obtiene un mayor porcentaje de clientes satisfechos, consiguiendo también una menor desviación estándar en el tiempo total.

Tras analizar todos los datos se puede observar que la carga de trabajo es demasiado elevada para el robot, la solución más rápida y sencilla en este caso, para evitar esta carga tan elevada de trabajo seria reducir el número de pedidos o en su defecto, aumentar el número de estaciones de trabajo

3.5) Conclusiones

En este Trabajo Fin de Grado, he logrado desarrollar un modelo de bar robótico que ha cumplido con los objetivos establecidos.

Durante el desarrollo de este proyecto, he aplicado diversos conocimientos adquiridos durante el grado, como los conocimientos aprendidos en las asignaturas de robótica industrial y simulación de sistemas dinámicos. Además, he aumentado mi capacidad y autonomía para resolver problemas y mi pensamiento crítico.

Aunque se ha logrado obtener un modelo funcional con un desempeño aceptable, los datos extraídos del modelo de JaamSim reflejan un fallo en mi estrategia de optimización del robot. A la hora de optimizar el robot, se ha tratado de disminuir el tiempo de servicio lo máximo posible, reduciendo las distancias entre los objetos, evitando movimientos innecesarios y aumentando la velocidad lo máximo que las circunstancias lo permitían. Aunque esta optimización ha ayudado a que aumente el rendimiento del robot, el camino no era el correcto. Los resultados finales muestran que la manera de evitar ese cuello de botella que sufre el robot es implementando pedidos múltiples.

Como conclusión final, se puede extraer que la robótica en el ámbito de la hostelería representa una alternativa viable desde el punto de vista práctico. A pesar de que este tipo de tecnología se encuentra en una fase temprana de desarrollo, como hemos podido ver en este Trabajo Fin de Grado, el rendimiento se puede llegar a asemejar al de un camarero real, lo que invita a pensar que cada vez sean más los negocios que adopten este tipo de tecnología.

3.6) Líneas de trabajo futuro

Debido a la extensión limitada de un Trabajo Fin de Grado, no se ha podido testear ni optimizar al máximo el modelo de bar robótico, aunque los resultados obtenidos sí que pueden dar una idea orientativa sobre la adecuación de robotizar un bar o no.

A continuación, propongo algunas líneas de trabajo futuro relacionadas con este TFG:

En primer lugar, probar el modelo de RobotStudio en el brazo IRB 1100 de la EUPT, comprobando que las velocidades y precisiones elegidas en la programación del modelo son las adecuadas para un trabajo de estas características.

Como segunda línea de trabajo futuro, propongo crear un prototipo capaz de comunicarse con el controlador del robot, el cual se pueda usar para realizar los pedidos, activando o desactivando las entradas digitales del controlador. El dispositivo debería intentar implementar algún método para realizar el pago de los pedidos.

Como última línea de trabajo futuro, a la vista de los resultados en el modelo de JaamSim, se puede intentar optimizar los puntos más débiles del modelo de RobotStudio, intentando mejorar la multitarea del robot, aumentando así la cantidad de pedidos que este es capaz de realizar.

REFERENCIAS

[1] D. Ghamlouche, «Aumento de gastos en autónomos y pymes». Accedido: 23 de enero de 2024. [En línea]. Disponible en: https://www.autonomosyemprendedor.es/articulo/gastos-autonomos/autonomos-pymes-pagan-33-mas-gastos-negocio-que-hace-anos-cepyme/20230324171911029795.html

[2] S. Delgado, «Incremento de costes laborales para las empresas». Accedido: 23 de enero de 2024. [En línea]. Disponible en: https://www.elblogsalmon.com/economia/empresas-estan-viviendo-fuerte-incremento-costes-laborales-eso-problema-para-ellas-trabajadores-economia-general

[3] C. Santamaría, «El primer ”camarero-robot” del mundo en Bangkok». Accedido: 23 de enero de 2024. [En línea]. Disponible en: https://www.lainformacion.com/asuntos-sociales/el-primer-camarero-robot-del-mundo-sirve-sushi-en-bangkok_MUhatCHpRzQzC8dfxWrIB5/

[4] «Products - PUDU». Accedido: 23 de enero de 2024. [En línea]. Disponible en: https://www.pudurobotics.com/products

[5] «▷ Bellabot | Robot Camarero ◁ Hostelería». Accedido: 23 de enero de 2024. [En línea]. Disponible en: https://bellabot.es/

[6] «Robots Camareros - Dax Robotics - Soluciones Robóticas». Accedido: 23 de enero de 2024. [En línea]. Disponible en: https://dax-robotics.com/robots-camareros/

[7] B. Robdríguez, «CaliExpress: el primer restaurante de comida rápida autónomo». Accedido: 23 de enero de 2024. [En línea]. Disponible en: https://elchapuzasinformatico.com/2023/12/caliexpress-restaurante-de-comida-rapida-autonomo/

[8] «Home | Makr Shakr». Accedido: 23 de enero de 2024. [En línea]. Disponible en: https://www.makrshakr.com/

[9] «RobotStudio® | ABB». Accedido: 23 de enero de 2024. [En línea]. Disponible en: https://new.abb.com/products/robotics/es/robotstudio

[10] «JaamSim | Free Discrete Event Simulation Software». Accedido: 23 de enero de 2024. [En línea]. Disponible en: https://jaamsim.com/

[11] A. Robotics, «Especificaciones del producto IRB 1100».

[12] «Dispensador de hielo |ITV Sirion 130 SLIM». Accedido: 23 de enero de 2024. [En línea]. Disponible en: https://www.equipoh.com/maquinas-de-hielo/dispensador-de-hielo-itv-sirion-dhd-130-slim-3467.html