

---

# GAAR-I

Generic Arm Assist Robot - Intelligent

---

PROJECT SPRINT #3.

DATE: 05<sup>th</sup> May 2021

ROGER REY MESA #1

JAVIER ALEGRE REVUELTA #2

DANIEL LÓPEZ LARA #3

MOHSIN RIAZ #4

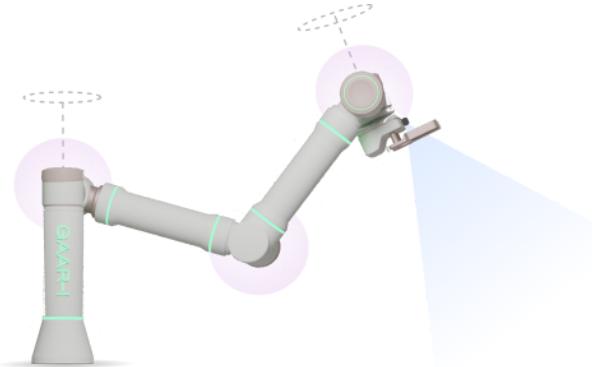
# Table of Contents

---

<b>Project description</b>	<b>1</b>
<b>Electronic components</b>	<b>3</b>
<b>Hardware Scheme</b>	<b>4</b>
Interconexiones, entradas y salidas	5
<b>Software Architecture</b>	<b>8</b>
Descripción de los módulos	9
Cinemática inversa (Parámetros Denavit-hartenberg)	13
<b>Amazing contributions</b>	<b>14</b>
<b>Extra components and 3D pieces</b>	<b>15</b>
Estructura del robot	15
Brazos y conexiones del robot	16
Instrumentos/Objetos 3d	17
<b>Simulation Strategy</b>	<b>18</b>
Diagrama de la estrategia de simulación	20
Máquina de estado: Traer <objeto>	21
Escena del simulador en Coppelia	23
<b>Foreseen risks and contingency plan</b>	<b>24</b>

# GAAR-I

Generic Arm Assist Robot - Intelligent



## Project description

GAAR-I es un brazo robótico inteligente, asistente de materiales y objetos. A través de un módulo de reconocimiento de voz se le indica un objeto a seleccionar situado en la base del robot, y gracias a la visión por computador identifica el objeto y se lo proporciona al usuario.

Nuestro robot está contextualizado en el ámbito quirúrgico, donde un médico/cirujano pedirá un instrumento que necesite como podría ser el bisturí y se lo acercará.

El funcionamiento del brazo consistirá en una primera fase donde el brazo estará situado en una posición inicial con una cámara apuntando a la base donde se encuentran los instrumentos implicados en la cirugía.

Al emitir una orden de forma oral el cirujano el robot procesa la petición del objeto en demandado y a través de la cámara y algorítmicas de visión por computador se llevará a cabo el reconocimiento del instrumento y dará a la orden al brazo de 5 ejes para que llegue a esa posición mediante la cinemática inversa. Una vez el

instrumento está en la pinza este irá a una posición fija donde sostendrá el objeto a una posición elevada cercana al cirujano para que este pueda coger el objeto deseado sin necesidad de mucho esfuerzo, lo que le permitirá seguir concentrado en la cirugía. En el momento en el que el cirujano coge el instrumento de la pinza este tendrá que emitir otra orden para que se abra la pinza y libere el objeto, una vez liberado el robot volverá a la posición inicial.

En el caso inverso, donde el cirujano quiere dejar un objeto, este tendrá que emitir otra orden para que el brazo vaya a la posición cerca del cirujano y este coloque el objeto en las pinzas, cuando quiera que el brazo cierre la pinza y devuelva el objeto juntos al resto deberá especificarlo.

Las órdenes disponibles que hay son:

- GAAR-I “objeto”: Con esta orden el brazo acercará el objeto al cirujano y esperará a nuevas órdenes. La palabra “objeto” deberá ser sustituida por el instrumento deseado.
- GAAR-I abre: La pinza se abrirá y después de un segundo volverá a su posición inicial.
- GAAR-I devuelve “objeto”: La pinza se cerrará y después de un segundo procederá a dejar el objeto en su posición inicial y volverá a su posición inicial. La palabra “objeto” deberá ser sustituida por el instrumento deseado.
- GAAR-I ven: Con esta orden el brazo se acercará al cirujano con la pinza abierta a la espera de nuevas órdenes.

El conjunto de objetos que va a ser capaz de reconocer serán los siguientes:

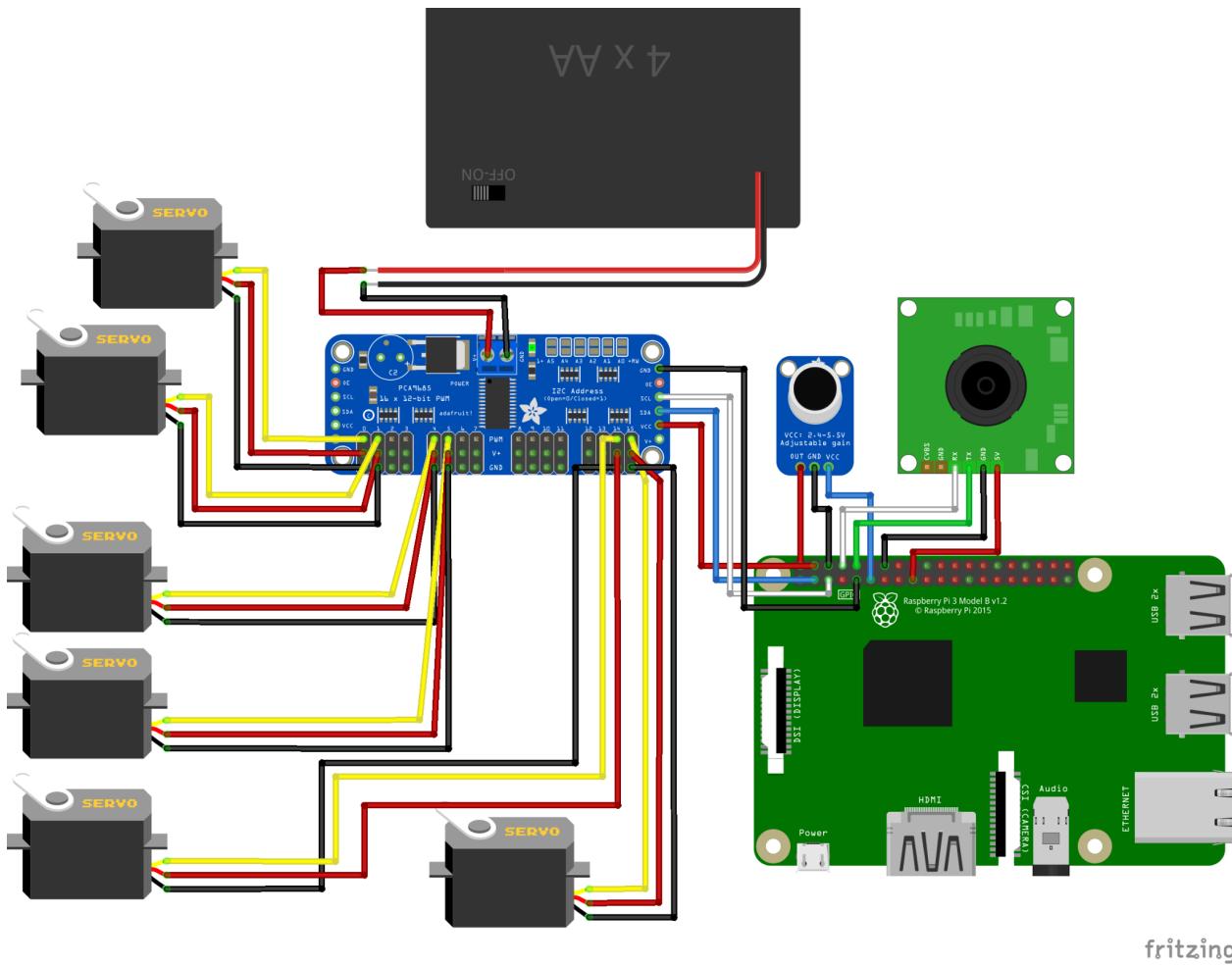
- Pinzas
- Bisturí
- Jeringuilla
- Tijeras

## Electronic components

Esta es la lista de los componentes eléctricos:

- Servo motor, 6 unidades
- Raspberry Pi 3 A+
- Electret Microphone Amplifier - MAX4466 with Adjustable Gain
- Raspberry Pi Camera Module V2
- Driver I2C PCA 9685
- Protoboard
- Cables
- Fuente de alimentación

## Hardware Scheme



## Interconexiones, entradas y salidas

Raspberry Pi Camera Module V2	Raspberry Pi 3 A+
RX	GPIO14
TX	GPIO15
GND	GND
5V	5V

Electret Microphone Amplifier	Raspberry Pi 3 A+
OUT	5V
GND	GND
VCC	GPIO17

Driver I2C PCA 9685	Raspberry Pi 3 A+
GND	GND
SCL	GPIO3
SDA	GPIO2
VCC	5V

Fuente de alimentación	Driver I2C PCA 9685
POS	PWRIN
NEG	GND
Servomotor 1	Driver I2C PCA 9685
Pulse	PWM0
VCC	5V
GND	GND
Servomotor 2	Driver I2C PCA 9685
Pulse	PWM1
VCC	5V
GND	GND
Servomotor 3	Driver I2C PCA 9685
Pulse	PWM4
VCC	5V
GND	GND

Servomotor 4

Driver I2C PCA 9685

Pulse

PWM5

VCC

5V

GND

GND

Servomotor 5

Driver I2C PCA 9685

Pulse

PWM14

VCC

5V

GND

GND

Servomotor 6

Driver I2C PCA 9685

Pulse

PWM15

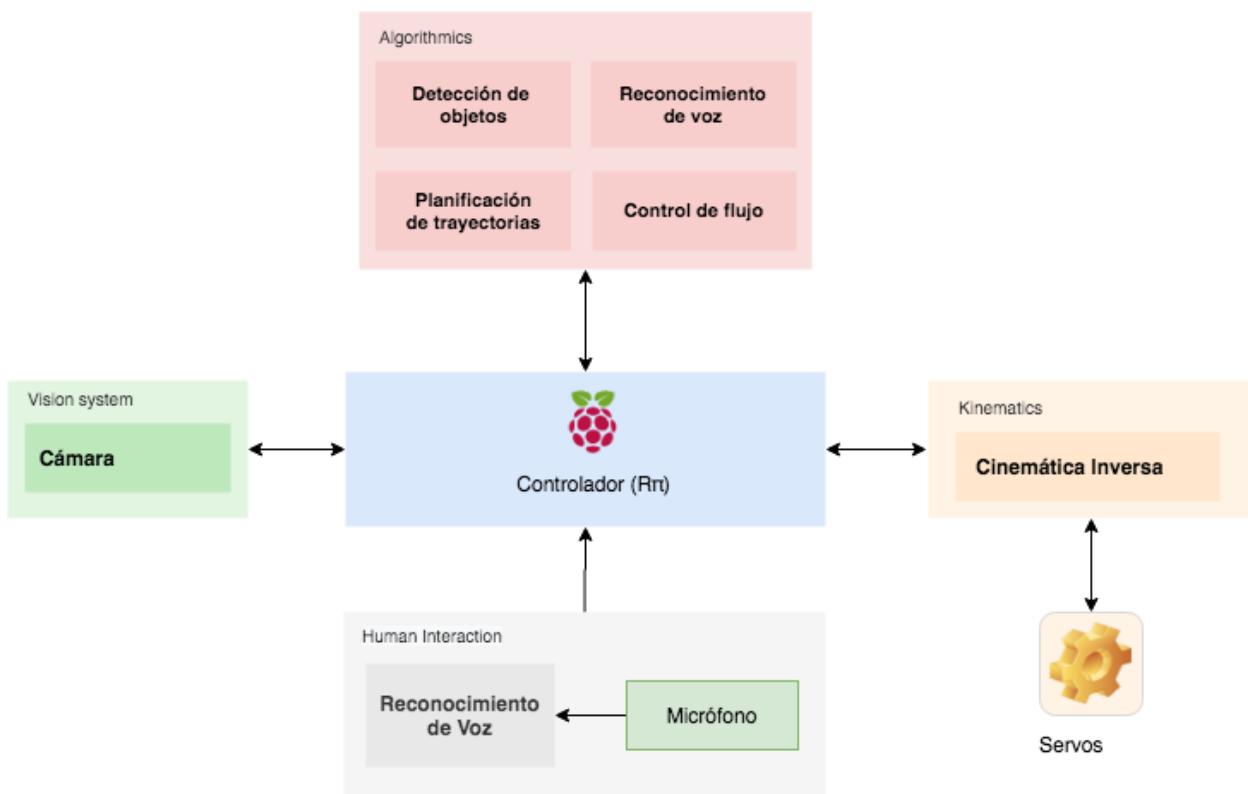
VCC

5V

GND

GND

## Software Architecture



## Descripción de los módulos

Módulo	<b>Reconocimiento de voz</b>
Descripción	<p>Este módulo se dedicará al reconocimiento de voz mediante el micrófono.</p> <p>Para poder pasar de voz a texto se utilizará una librería python. Las órdenes vendrán dadas siempre precedidas por la palabra “GAAR-I”, de esta forma al reconocer esta primera palabra se hará una búsqueda en una lista de nombres del resto de la orden.</p> <p>Las órdenes permitidas que siguen al prefijo serán las siguientes:</p> <ul style="list-style-type: none"><li>• “Nombre del objeto”</li><li>• Abre</li><li>• Devuelve “Nombre del objeto”</li><li>• Ven</li></ul> <p>Donde “Nombre del objeto” puede ser cualquiera de los objetos predefinidos que el brazo será capaz de coger.</p>
Entrada	Audio mediante el micrófono.
Salida	Identificador de la orden y el id del objeto.

Módulo	<b>Detección de objetos</b>
Descripción	<p>Este algoritmo se dedicará a captar la imagen de la zona de los objetos y posteriormente mediante filtrado, transformaciones y algoritmos de etiquetajes detectará los objetos.</p> <p>Se utilizarán técnicas de correlación y transformaciones</p>

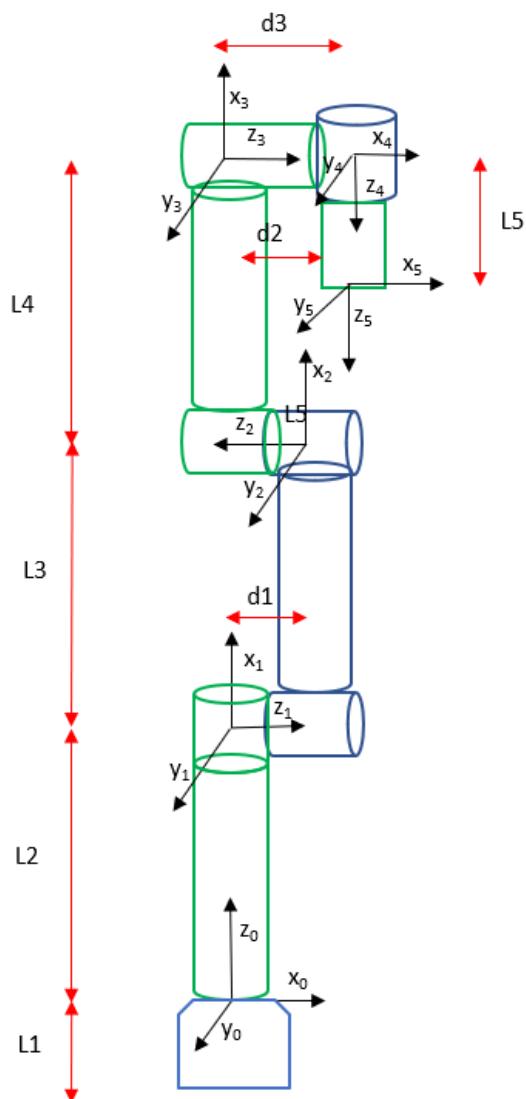
	<p>geométricas entre la imagen del objeto de referencia ya generada previamente y la imagen capturada actual de los objetos.</p> <p>Con los procedimientos anteriores, sabiendo las dimensiones de la zona y las distancias se estimará la posición concreta del objeto a recoger o para devolver.</p>
Entrada	Imagen de la zona de los objetos.
Salida	<ul style="list-style-type: none"> <li>• Indicar si se ha encontrado el objeto.</li> <li>• Devolver la posición (x,y,z) y la rotación respecto el eje Z.</li> </ul>

Módulo	<b>Cinemática Inversa</b>
Descripción	<p>A partir de los datos obtenidos del algoritmo de visión o las posiciones ya preprogramadas obtenemos los datos de entrada al algoritmo.</p> <p>Utilizando los parámetros de Denavit–Hartenberg y las dimensiones de robot, calculamos la matriz para obtener el sistema de ecuaciones de traslación para cada eje. Utilizando el <i>nsolve</i> obtenemos los ángulos para cada servo.</p> <p>Las operaciones que deberá realizar el robot serán las siguientes:</p> <ul style="list-style-type: none"> <li>• Navegar a la posición deseada.</li> <li>• Recoger o devolver el objeto dependiendo de la orden (abrir y cerrar la pinza).</li> <li>• Recoger un objeto teniendo en cuenta el ángulo de su posición respecto al eje Z.</li> </ul>
Entrada	Posición en 3D.
Salida	Navegar hasta la posición indicada.

Módulo	<b>Planificación de secuencias de movimientos</b>
Descripción	<p>Este algoritmo se dedicará a crear una secuencia de movimientos del robot para poder llevar a cabo una orden.</p> <p>Empezará desde su posición inicial (reposo), ejecutará la orden y volverá a su posición de reposo.</p> <p>Ejemplo:</p> <ul style="list-style-type: none"> <li>• El robot se encuentra en posición de reposo preprogramada.</li> <li>• Se le ordena traer un objeto en concreto.</li> <li>• El robot planificará la siguiente secuencia: <ul style="list-style-type: none"> <li>◦ Ir desde la posición actual a la de la zona de los objetos.</li> <li>◦ Navegar a la posición concreta del objeto a recoger.</li> <li>◦ Realizar las operaciones de la pinza para agarrar el objeto.</li> <li>◦ Navegar a la posición preprogramada para ofrecer el objeto al usuario.</li> <li>◦ Volver a la posición de reposo del robot.</li> </ul> </li> </ul>
Entrada	Identificador de la orden y id del objeto.
Salida	Una secuencia de movimientos del robot.

Módulo	<b>Control de flujo (controlador)</b>
Descripción	<p>Este módulo se dedicará a controlar y coordinar el resto de módulos para poder efectuar las órdenes deseadas por el usuario.</p> <p>Esta parte será la encargada de:</p> <ul style="list-style-type: none"> <li>• Poner en marcha al robot y realizar la secuencia de inicialización y pruebas.</li> <li>• Tener integrados los algoritmos de voice-to-text.</li> <li>• Tener integrado el algoritmo de planificación de la secuencia de movimientos.</li> <li>• Ejecutar la secuencia de movimiento usando la cinemática inversa.</li> <li>• Usar algoritmos de visión para ubicar los objetos.</li> <li>• Controlar el estado de la secuencia de movimientos.</li> <li>• Ver si la orden se ha completado para volver a la posición de reposo.</li> <li>• etc...</li> </ul>

## Cinemática inversa (Parámetros Denavit-hartenberg)



Articulación	$\theta$	d	a	$\alpha$
1	$\theta_1$	L1 + L2	0	90º
2	$\theta_2$	d1	L3	180º
3	$\theta_3$	d2	L4	180º
4	$\theta_4$	d3	0	90º
5	$\theta_5$	L5	0	0

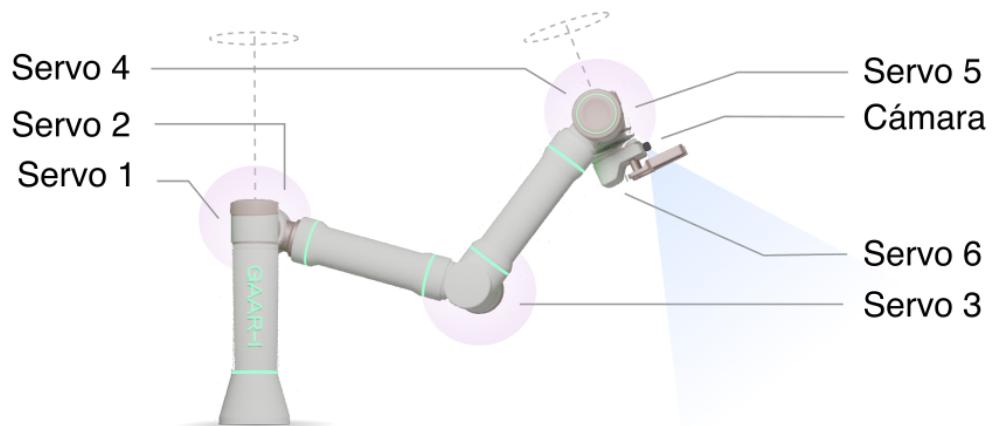
## Amazing contributions

- Eliminación de personal durante las cirugías.
- Ayuda al cirujano acercando los instrumentos que necesita.
- Control por voz permite dar órdenes al brazo y tener libres las manos al mismo tiempo.
- Visión integrada con una inteligencia artificial que le permite determinar cuál es el objeto deseado y permitiendo que la colocación de los objetos no esté totalmente predefinida y tenga cierto grado de flexibilidad.
- Evita la pérdida de tiempo ya que el brazo reconoce y encuentra el objeto rápidamente, no comete errores humanos derivados de la presión, distracción o cansancio.

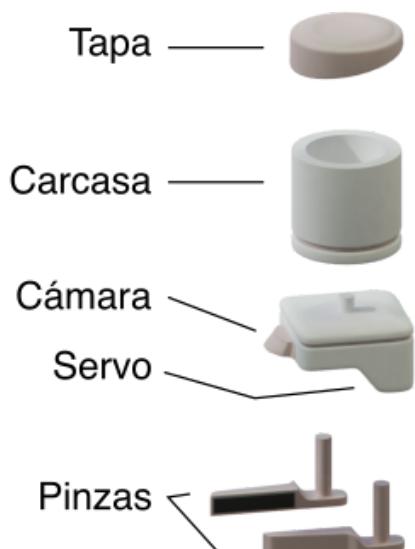
Opinamos que la realización de todos las características que hemos presentado constituyen un proyecto de brazo robótico completo y operativo y que pensar en añadir más funcionalidades solo adornaría el proyecto pero realmente no le sumaría valor ya que para el problema que queremos resolver esta solución lo satisface.

## Extra components and 3D pieces

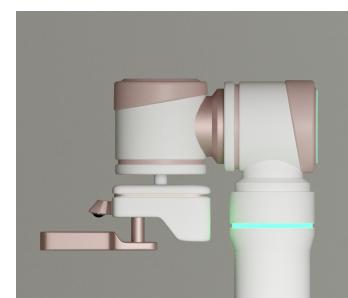
### Estructura del robot



Manipulador + Pinza



Detalle Pinza + cámara



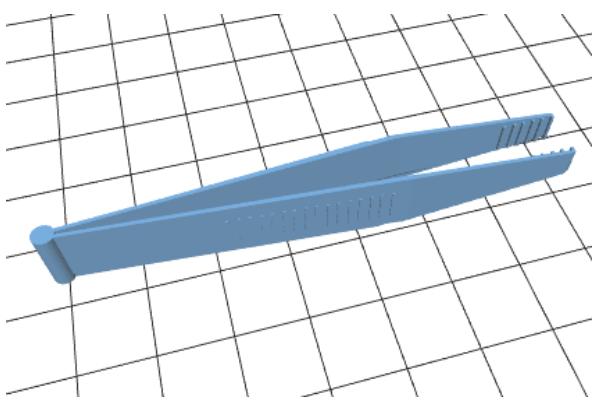
## Brazos y conexiones del robot



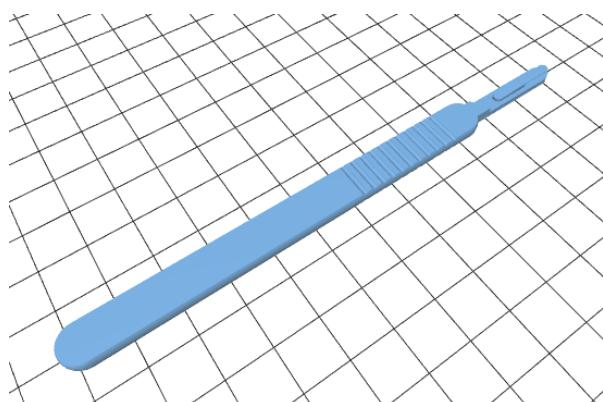
## Instrumentos/Objetos 3d

Nuestro robot está contextualizado en el ámbito quirúrgico, por ello los objetos con los que interactúa también pertenecerán a ese mismo ámbito. Por ello, se han utilizado modelos de varios instrumentos quirúrgicos tanto para la simulación como para el entrenamiento de la red neuronal encargada de identificarlos dentro de una imagen tomada por el robot.

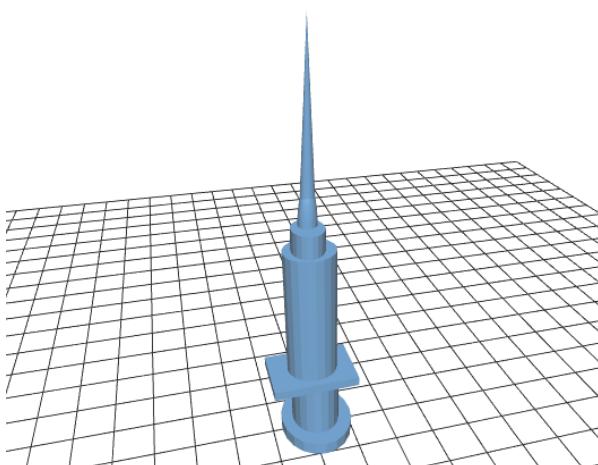
Pinzas



Bisturí



Jeringuilla



Tijeras



## Simulation Strategy

Los diseños para la escena de la simulación se realizarán mediante tinkercad. Estará ambientada en una operación, tendremos una camilla con el paciente y una mesa sobre la que estará GAAR-I. El robot estará posicionado de manera estratégica para poder asistir óptimamente al cirujano, que estará frente al paciente.

La simulación se realizará mediante la interconexión entre Coppelia y Script de Python.

- **Coppelia:** es donde se representará la simulación. Podremos observar la escena sobre el comportamiento del robot ante los comandos que se le vayan indicando.
- **Python Script:** aquí es donde todo el código que conforma la inteligencia artificial del robot se va a ejecutar. Se encargará por tanto de ejecutar el código necesario para procesar las entradas de voz, determinar cuál es el instrumento requerido, determinar la posición donde se encuentra, hacer que el robot lo coja y se lo lleve al cirujano.

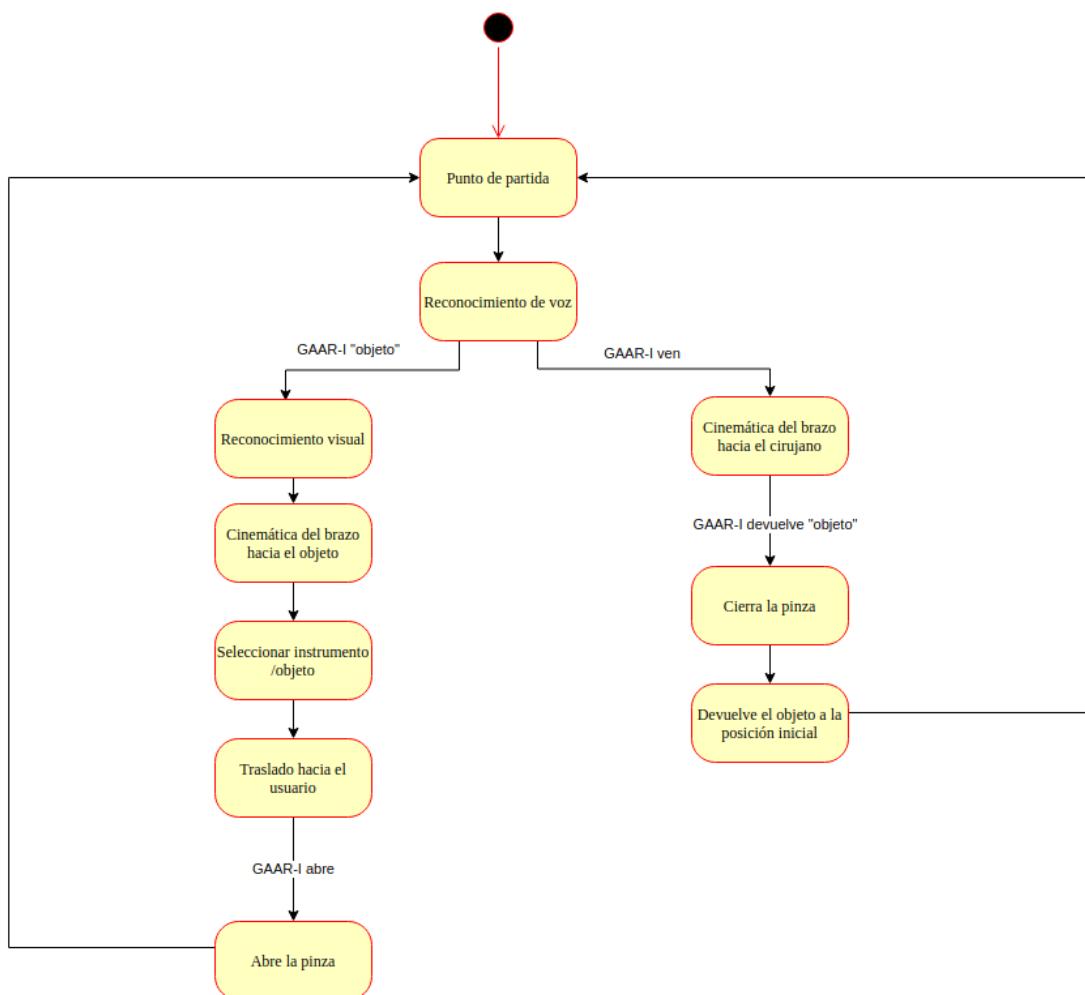
La estrategia de simulación va a ser la siguiente, dividida en 2 situaciones:

- **El cirujano quiere un instrumento:**
  - **Punto de partida:** GAAR-I, siempre está en un punto inicial fijo con la pinza abierta, esperando las órdenes del usuario.
  - **Reconocimiento de voz:** mediante un micro indicaremos cuál es el instrumento requerido mediante el comando "GAAR-I <<objeto>>". En Python, donde estará implementada la inteligencia artificial capaz de reconocer y procesar la voz, se determinará cuál es el objeto que se ha solicitado.
  - **Reconocimiento visual:** desde el Coppelia se llevará la imagen al Python, donde la inteligencia artificial procesa la imagen y determinará dónde se encuentra el objeto.

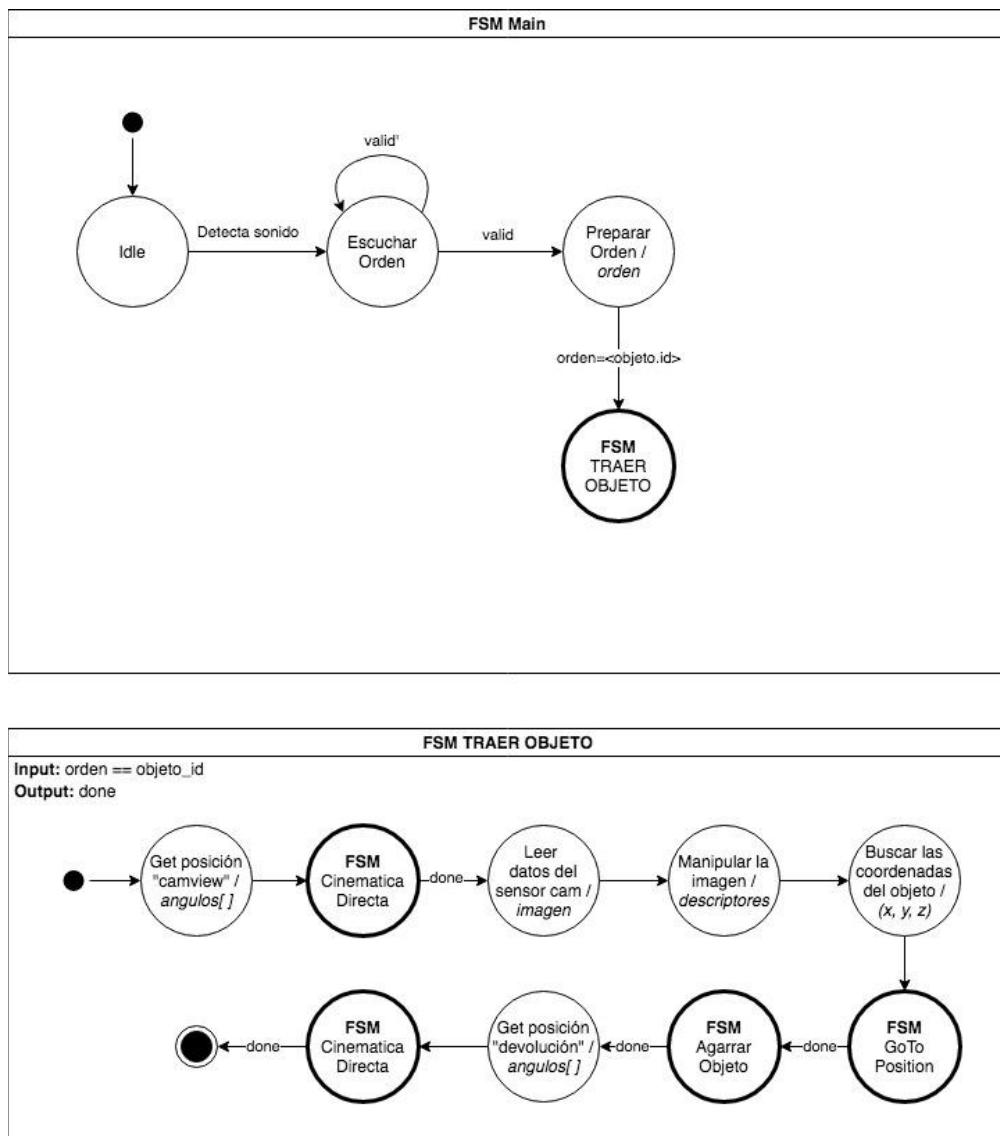
- **Cinemática del brazo:** se realizará la cinemática inversa para determinar los ángulos necesarios de cada motor de GAAR-I para llegar a la posición necesaria. Desde Python se enviarán las órdenes para que el robot se mueva a esa posición. En un diccionario, asociaremos el nombre del instrumento con la posición a la que se encuentra.
  - **Seleccionar instrumento/objeto:** el robot una vez situado encima de la pieza (a una altura fija determinada) comenzará a cerrar las pinzas hasta que el objeto quede sujetado.
  - **Traslado hacia el usuario:** el brazo una vez haya recogido la pieza se desplazará a la posición fijada determinada, donde el cirujano podría recogerlo. Una vez el cirujano esté preparado para coger esa pieza, le indicará que abra la pinza con el comando por voz pertinente.
  - **GAAR-I vuelta a origen:** posteriormente al paso anterior, GAAR-I volverá a la posición de inicio, preparado para volver a iniciar el primer punto de partida.
- 
- **El cirujano quiere dejar un instrumento:**
    - **Punto de partida:** GAAR-I estará en el punto inicial del que siempre parte con la pinza abierta.
    - **Reconocimiento de voz:** mediante el micro indicaremos que queremos devolver el objeto con el comando “GAAR-I ven” y se procesa.
    - **Movimiento brazo:** el brazo aplicará los ángulos predefinidos para realizar la acción de devolver y colocarse en la posición en la que va a recibir el objeto.

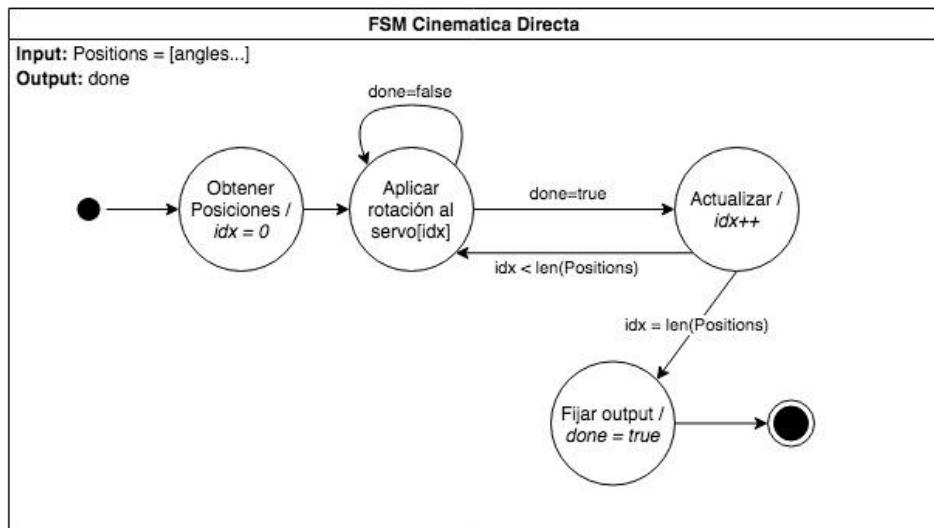
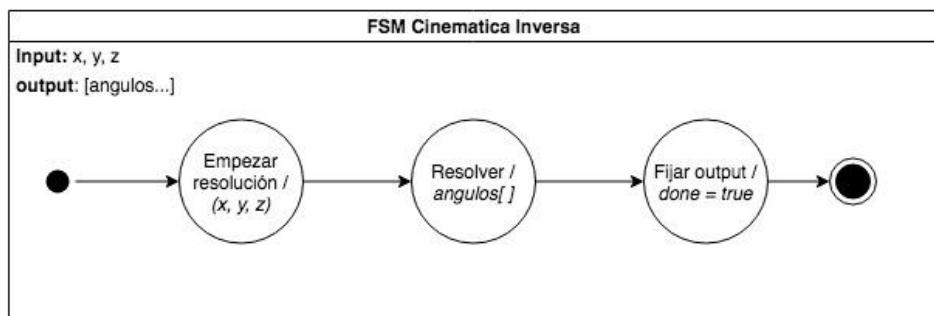
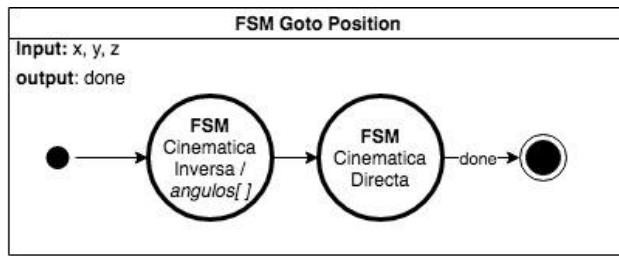
- Se colocará el objeto en la pinza del brazo y con el comando “GAAR-I devuelve <<instrumento>>”. Gracias al diccionario, se podrá determinar la posición asociada a ese instrumento.
- GAAR-I irá a esa posición y dejará el objeto en la base.
- GAAR-I vuelta al origen: GAAR-I vuelve al punto de partida.

### Diagrama de la estrategia de simulación

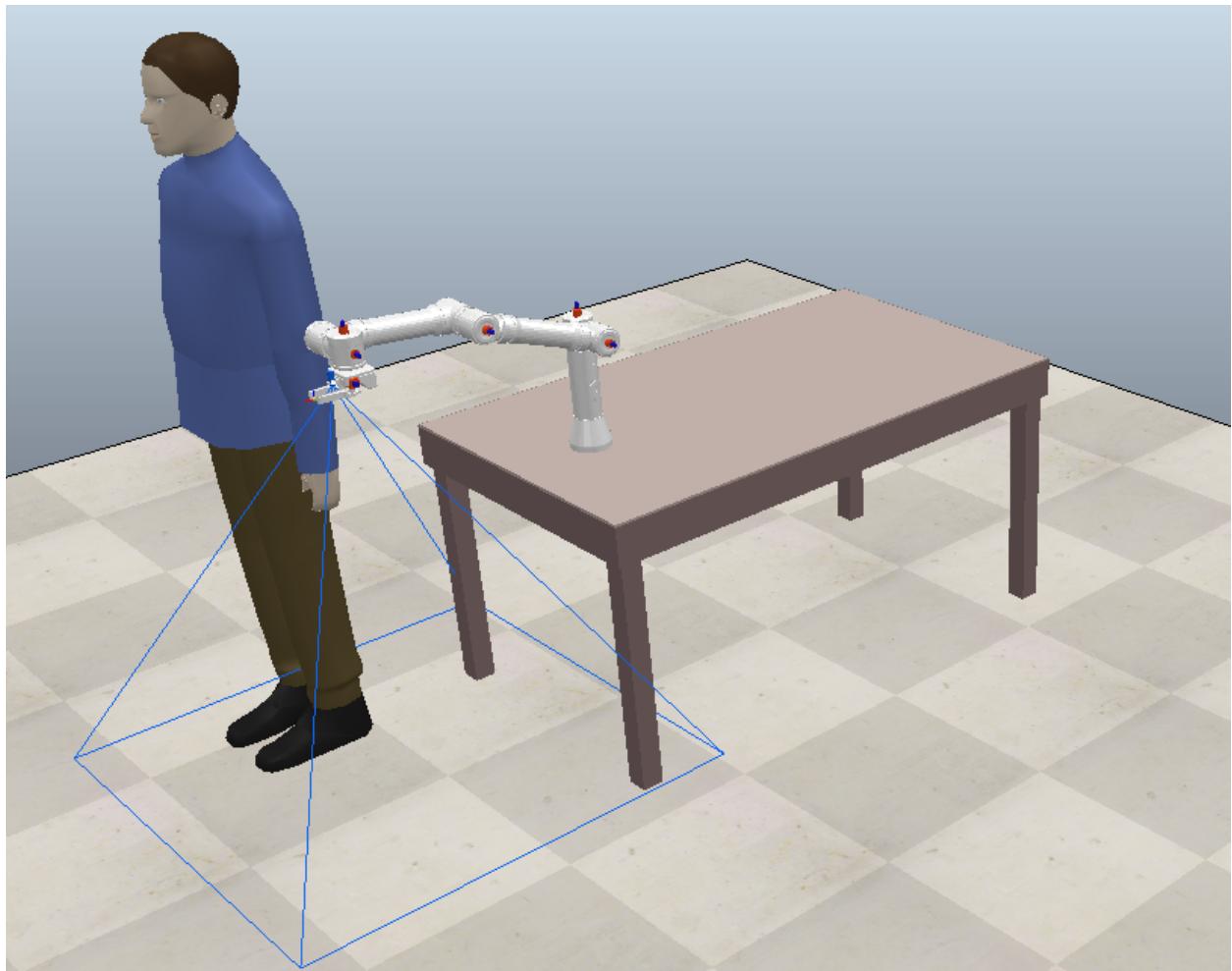


## Máquina de estado: Traer <objeto>





## Escena del simulador en Coppelia



## Foreseen risks and contingency plan

Risk #	Description	Probability (High/Medium/Low)	Impact (High/Medium/Low)	Contingency plan
1	No conseguir el reconocimiento de voz	Baja	Alto	No se puede realizar el proyecto si ocurre esto
2	No conseguir el reconocimiento visual de objeto	Baja	Alto	No se puede realizar el proyecto si ocurre esto
3	No conseguir la cinemática inversa del robot	Baja	Alto	No se puede realizar el proyecto si ocurre esto
4	Baja de un miembro del equipo	Baja	Medio	Repartir el trabajo entre el resto de integrantes del grupo.
5	Coste superior al previsto, por falta de componente.	Medio	Medio	Añadir el componente si no es muy caro, buscar una alternativa o sacrificar prestaciones según el componente en cuestión.

6	No conseguir acabar el trabajo del sprint a tiempo	Medio	Alto	Aumentar las tareas a realizar para el próximo sprint y dedicarle mayor número de horas.
7	Simulador no adecuado	Baja	Alto	Cambiar de simulador o hacer una simulación más simple, dependiendo del estado de progreso del proyecto
8	Mal diseño de los brazos del robot (no llega a las posiciones requeridas)	Baja	Alto	Rediseño (aumentamos o disminuimos la longitud de los brazos)
9	Diseño de la pinza ineficiente para coger los objetos	Medio	Medio	Rediseño de la pinza
10	Componentes eléctricos no adecuados	Baja	Alto	Cambiar de componentes
11	Componentes averiados	Baja	Alto	Habría que volverlo a comprar si es imprescindible o adaptar el robot para suplir su ausencia.

# References

---

Enlace al proyecto principal en el que nos hemos inspirado:

- <https://www.instructables.com/Voice-Controlled-Robot-Arm/>

Enlaces a otros proyectos:

- <https://bit.ly/3dQqsD7>
- <https://bit.ly/3pVwgo0>
- [https://marvelcinematicuniverse.fandom.com/es/wiki/Dum-E\\_y\\_U](https://marvelcinematicuniverse.fandom.com/es/wiki/Dum-E_y_U)