

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería



Diseño e implementación de un paquete de herramientas de software para controlar inalámbricamente un manipulador serial R17 dentro de un ecosistema basado en captura de movimiento

Trabajo de graduación presentado por José David Pellecer Orellana para optar al grado académico de Licenciado en Ingeniería Mecatrónica

Guatemala,

2022

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería



Diseño e implementación de un paquete de herramientas de software para controlar inalámbricamente un manipulador serial R17 dentro de un ecosistema basado en captura de movimiento

Trabajo de graduación presentado por José David Pellecer Orellana para optar al grado académico de Licenciado en Ingeniería Mecatrónica

Guatemala,


2022


Vo.Bo.:

(f) 
MSc. Miguel Enrique Zea Arenales

Tribunal Examinador:

(f) 
MSc. Miguel Enrique Zea Arenales

(f) 
Ing. José Eduardo Morales Espinoza

(f) 
Ing. Otto Armando Girón González

Fecha de aprobación: Guatemala, 5 de enero de 2022.

La aplicación de la robótica ha sido para mí un tema de interés desde que me encuentro en bachillerato. En este ámbito he podido integrar y darle sentido a conceptos matemáticos y físicos, lo cual es algo que generó en mí bastante interés y satisfacción. Por otra parte, el amplia área de oportunidad de poder aplicar dicha rama de la ciencia en función del bien del desarrollo del país, fue un agregado que impulsó mi decisión en este campo para la elaboración de mi trabajo de graduación.

Este trabajo representa la culminación de 5 años de aprendizaje, esfuerzo, aspiración y retos. Llegar hasta este punto ha sido posible gracias al apoyo de ciertas personas a quienes me gustaría agradecer. En primer lugar, le agradezco a mi padres, Betzaly Orellana y Ramiro Pellecer, quienes me han motivado y apoyado a siempre seguir mis sueños. También por ser modelos ejemplares en mi vida y por su amor incondicional. En segundo lugar, le agradezco a mi profesor y asesor, Miguel Zea, por su tiempo, consejos y apoyo a lo largo del proyecto.

Agradezco también a mis hermanos, quienes me apoyaron y se alegraron por todas las metas cumplidas a lo largo de este trayecto. Le agradezco en especial a mi hermano Ricardo, futuro ingeniero electrónico, quien siempre me ayudó en temas relacionados a la carrera. Un especial agradecimiento a mi abuelita, mi *mamita*, a quien quiero mucho y siempre me motivó a esforzarme y seguir mis sueños.

Es para mi importante dar las gracias a todos aquellos excelentes catedráticos, porque promovieron mi interés por la carrera de ingeniería mecatrónica y me motivaron a seguir avanzando. Hago mención especial de mis profesores Magda Moscoso, Hugo Pallais y Mario Castillo, en quienes vi siempre la vocación de enseñar y que se interesaron por mi aprendizaje.

Por último, le deseo lo mejor a todos mis compañeros de carrera, con quienes compartí alegrías y preocupaciones, pero sobre todo buenos momentos, los cuáles recuerdo con gran cariño.

Prefacio	v
Lista de figuras	xi
Lista de cuadros	xiv
Resumen	xv
Abstract	xvii
1. Introducción	1
2. Antecedentes	3
3. Justificación	5
4. Objetivos	7
4.1. Objetivo general	7
4.2. Objetivos específicos	7
5. Alcance	9
6. Marco teórico	11
6.1. Sistema de captura de movimiento	11
6.1.1. Marker-based	12
6.1.2. Markerless	13
6.1.3. OptiTrack	14
6.2. Robótica	16
6.2.1. Cinemática de cuerpos rígidos	19
6.2.2. Cinemática directa de manipuladores seriales	28
6.2.3. Cinemática diferencial de manipuladores seriales	30
6.2.4. Cinemática inversa de manipuladores seriales	32
6.3. Brazo robótico R17	35
6.3.1. ROBOFORTH II	36

6.4. Comunicación inalámbrica	37
6.4.1. WiFi	38
6.4.2. Protocolo MQTT	39
7. Metodología	41
8. Montaje, configuración, pruebas y obtención de datos en el sistema de captura de movimiento OptiTrack	45
8.1. Montaje y configuración	45
8.2. Pruebas y obtención de datos	46
8.2.1. Posición 1	48
8.2.2. Posición 2	49
8.2.3. Posición 3	50
8.2.4. Posición 4	51
9. Modelado cinemático y desarrollo de software para controlar al manipu- lador serial R17	53
9.1. Modelo cinemático y representación DH	53
9.2. Implementación de la cinemática directa	57
9.3. Implementación de la cinemática inversa	59
9.4. Implementación de ejecución de trayectorias	63
9.5. Interfaz gráfica	70
10. Integración del manipulador serial R17 con el ecosistema Robotat	75
10.1. Sistema embebido: Robotat - R17 Module	75
10.1.1. Selección de componentes	75
10.1.2. Diseño electrónico y fabricación	76
10.1.3. Validación del módulo	78
10.2. Librería R17	79
10.2.1. Interacción dentro del ecosistema Robotat	82
10.3. Integración con OptiTrack	82
10.3.1. Cinemática inversa	83
10.3.2. Ejecución de trayectorias	85
11. Conclusiones	87
12. Recomendaciones	89
13. Bibliografía	91
14. Anexos	95
14.1. Pruebas de cinemática directa	95
14.2. Pruebas de cinemática inversa	96
14.3. Diseño y fabricación de sistema embebido	98
14.4. Repositorio de trabajo	99
14.5. Videos explicativos y demostrativos	99

Lista de figuras

1. Cámaras ofrecidas por el sistema Optitrack [14].	14
2. Motive Tracker [16].	16
3. Motive Body [16].	16
4. Diagrama cinemático de un manipulador serial.	17
5. Juntas comúnmente utilizadas en robots [17].	18
6. Información sobre grados de libertad en las juntas típicas de un robot [17].	18
7. Representación de un mismo punto p con respecto a dos marcos de referencia diferentes $\{a\}$ y $\{b\}$ [17].	20
8. Configuración del cuerpo rígido con body frame $\{b\}$ con respecto a el sistema fijo $\{s\}$ [17].	21
9. Traslación pura en 2D [20].	21
10. Rotación pura en 2D [20].	22
11. Alineación de ejes causantes de <i>gimbal lock</i> [22].	26
12. Representación eje-ángulo [17].	27
13. Unión entre eslabones según la representación de Denavit-Hartenberg [21].	29
14. Manipulador RRRP [17].	30
15. Diagrama de bloques del sistema de control que implementa el algoritmo para la cinemática inversa utilizando la inversa o pseudo-inversa del jacobiano [19].	33
16. Brazo robótico R17 [25].	35
17. Articulaciones del R17 [5].	36
18. Comandos en ROBOFORTH para R17 [5].	37
19. Arquitectura lógica funcional de IEEE 802.11 [29].	38
20. Arquitectura publish/subscribe de MQTT [30].	39
21. Montaje del sistema OptiTrack.	46
22. Cuerpo rígido en efector final del R17.	46
23. Selección de marcadores para creación de cuerpo rígido en el efector final del R17.	47
24. Cuerpo rígido en el efector final del R17.	47
25. Posición 1 - R17 real.	48
26. Posición 1 - Motive.	48
27. Posición 2 - R17 real.	49
28. Posición 2 - Motive.	49

29. Posición 3 - R17 real.	50
30. Posición 3 - Motive.	50
31. Posición 4 - R17 real.	51
32. Posición 4 - Motive.	51
33. Planos del manipulador R17 [25].	54
34. Comparación entre la simulación de la geometría del R17 en MATLAB y el sistema real para una configuración $q = [0, 0, 0, 0, 0, 0]^T$.	55
35. Comparación entre la simulación de la geometría del R17 en MATLAB y el sistema real para una configuración $q = [0, -68, 59, -70, 0, 0]^T$.	56
36. Comparación entre la simulación de la geometría del R17 en MATLAB y el sistema real para una configuración $q = [0, -85, 3, -80, 0, 0]^T$.	56
37. Diagrama de flujo de la función R17fKine.	57
38. Diagrama de flujo de la función R17iKine.	60
39. Ejecución de trayectoria en modo continuo [34].	63
40. Diagrama de flujo de la función R17CreateRoute.	64
41. Análisis en Tracker de trayectoria corrida en modo segmentado.	66
42. Análisis en Tracker de trayectoria corrida en modo continuo.	66
43. Coordenadas y y z de la trayectoria de prueba del efector final corrida en modo segmentado.	67
44. Velocidad lineal de la trayectoria de prueba del efector final corrida en modo segmentado.	68
45. Coordenadas y y z de la trayectoria de prueba del efector final corrida en modo continuo.	69
46. Velocidad lineal de la trayectoria de prueba del efector final corrida en modo continuo.	69
47. Pestaña principal.	70
48. Pestaña principal: conexión e inicialización.	71
49. Interfaz gráfica para modalidad de cinemática directa.	72
50. Interfaz gráfica para modalidad de cinemática inversa.	73
51. Interfaz gráfica para modalidad de ejecución de trayectorias.	74
52. Esquemático de Robotat -R17 Module.	77
53. Señales de salida del ESP32 y el Robotat - R17 Module.	78
54. Diagrama UML.	82
55. Diagrama de transformaciones.	83
56. Coordenadas y y z de la trayectoria de prueba del efector final corrida en modo segmentado dentro del OptiTrack.	85
57. Coordenadas y y z de la trayectoria de prueba del efector final corrida en modo continuo dentro del OptiTrack.	86
58. Posición del efector final dada por el comando WHERE del R17 para una configuración q_1 del Cuadro 4.	95
59. Posición del efector final dada por el comando WHERE del R17 para una configuración q_2 del Cuadro 4.	95
60. Posición del efector final dada por el comando WHERE del R17 para una configuración q_3 del Cuadro 4.	96

61.	Posición del efector final dada por el comando WHERE del R17 para una configuración q_4 del Cuadro 4.	96
62.	Posición del efector final dada por el comando WHERE del R17 para una configuración q_5 del Cuadro 4.	96
63.	Posición del efector final dada por el comando WHERE del R17 para una configuración correspondiente a el vector de posición p_1 del Cuadro 9.	96
64.	Posición del efector final dada por el comando WHERE del R17 para una configuración correspondiente a el vector de posición p_2 del Cuadro 9.	97
65.	Posición del efector final dada por el comando WHERE del R17 para una configuración correspondiente a el vector de posición p_3 del Cuadro 9.	97
66.	Posición del efector final dada por el comando WHERE del R17 para una configuración correspondiente a el vector de posición p_4 del Cuadro 9.	97
67.	Posición del efector final dada por el comando WHERE del R17 para una configuración correspondiente a el vector de posición p_5 del Cuadro 9.	97
68.	Diseño electrónico de PCB del sistema embebido Robotat - R17 Module.	98
69.	Modelo 3D de Robotat - R17 Module.	98
70.	Sistema embebido Robotat - R17 Module.	99

1.	Especificaciones de imagen de Prime ^x 41 [15].	15
2.	Especificaciones de desempeño de rastreo y alcance de Prime ^x 41 [15].	15
3.	Matriz de parámetros DH de manipulador serial R17.	54
4.	Configuraciones de prueba para comparar el resultado del modelo cinemático planteado con el real del R17.	58
5.	Coordenadas x , y y z teóricas para las configuraciones mostradas en el Cuadro 4.	58
6.	Coordenadas x , y y z del modelo planteado para las configuraciones mostradas en el Cuadro 4.	58
7.	Porcentajes de error de las coordenadas x , y y z del modelo planteado (Cuadro 6) y los valores teóricos obtenidos por el comando WHERE (Cuadro 5) del R17 para las configuraciones mostradas en el Cuadro 4.	59
8.	Coordenadas x , y y z teóricas de las posiciones meta del efector final, utilizadas en la función para la cinemática inversa R17iKine.	61
9.	Configuraciones obtenidas del algoritmo de cinemática inversa al emplear las posiciones meta del efector final del Cuadro 8.	61
10.	Coordenadas x , y y z experimentales sin ajuste del efecto que tiene la junta prismática; posiciones de efector final alcanzadas al utilizar las configuraciones del Cuadro 9 en la función R17fKine.	61
11.	Coordenadas x , y y z experimentales con ajuste del efecto que tiene la junta prismática; posiciones de efector final alcanzadas al utilizar las configuraciones del Cuadro 9 en la función R17fKine.	62
12.	Porcentajes de error de las coordenadas x , y y z al usar el algoritmo de cinemática inversa del modelo planteado (Cuadro 11) y los valores teóricos ingresados en la función R17iKine (Cuadro 8).	62
13.	Tabla de posiciones cartesianas de trayectoria de prueba.	65
14.	Componentes electrónicos utilizados en el Robotat -R17 Module.	77
15.	Atributos del objeto <code>r17_t</code>	79
16.	Métodos del objeto <code>r17_t</code>	81
17.	Coordenadas x , y y z teóricas de las posiciones meta del efector final dentro del OptiTrack.	83

18. Coordenadas x , y y z de las posiciones meta del efector final obtenidas por el OptiTrack.	84
19. Errores absolutos y relativos de las distancias de las posiciones meta del efector final teóricas (Cuadro 17) y las obtenidas por el OptiTrack (Cuadro 18).	84
20. Tabla de posiciones cartesianas de trayectoria de prueba dentro del OptiTrack.	85

En este trabajo se presenta el desarrollo e implementación de un paquete de herramientas de software que permita la ejecución de trayectorias del manipulador R17 en un espacio de tarea observado por un sistema de captura de movimiento. Se abarcó desde el levantamiento del sistema hasta el desarrollo y pruebas de software del manipulador.

El sistema de captura de movimiento utilizado fue el *OptiTrack*, con el cual se contaba con 6 cámaras *Prime^x 41*. El proceso para su levantamiento abarca el posicionamiento de las cámaras, sus conexiones, el proceso de calibración y la obtención de datos (poses de cuerpos rígidos). Además se trabajó con los SDK's disponibles, exportando la información obtenida a Python.

Con respecto al manipulador R17, se estableció el modelo cinemático usando la convención de Denavit-Hartenberg, y se hizo pruebas de cinemática directa, inversa y ejecución de trayectorias. Se validó este modelo de manera cualitativa y cuantitativa, comparando posiciones entre la simulación y el robot real. Se obtuvo errores absolutos en la posición del efector final menores a 20.66 mm (para las pruebas de cinemática directa) y porcentajes de error no mayores al 4% (para las pruebas de cinemática inversa). Además se compararon los modos *segmentado* y *continuo* para ejecutar trayectorias, de lo cual se concluyó que el modo continuo tiene un mayor error de posición en comparación con el modo segmentado, sin embargo posee mayor continuidad en la velocidad lineal del efector final y puede desempeñar una trayectoria en menor tiempo.

Posteriormente, se creó un módulo de comunicación inalámbrica con la intención de integrar al brazo robótico al ecosistema Robotat. Este módulo es un sistema embebido que cuenta con una placa electrónica y un estuche de protección diseñado y fabricado a la medida. El PCB cuenta con un microcontrolador ESP32, un par de baterías Li-ion18650 y un circuito encargado de convertir los niveles de las líneas de un puerto serie RS-232 a niveles TTL y viceversa. Se validó el correcto funcionamiento de este midiendo y analizando la señal de salida con un osciloscopio. Finalmente, se demostró que efectivamente el módulo permite integrar al R17 dentro del ecosistema Robotat, mediante a pruebas de cinemática inversa y ejecución de trayectorias, siendo el robot observado por el OptiTrack.

This work presents the development and implementation of a software tool package that allows the execution of trajectories of the R17 manipulator in a task space observed by a motion capture system. This work covered from the set up of the system to the development and testing of the manipulator software.

The motion capture system used was the *OptiTrack*, with which there were 6 *Prime^x 41* cameras. The process for its set up includes the positioning of the cameras and their connections, the calibration process, and the data gathering (rigid bodies' poses). In addition, available SDKs were used in order to export information about the pose of its end effector to Python.

Regarding the R17 manipulator, the kinematic model was established using the Denavit-Hartenberg convention. Moreover, tests of forward and inverse kinematics, and trajectory execution were made. This model was validated qualitatively and quantitatively, comparing positions between the simulation and the real robot. Absolute errors were obtained in the position of the end effector, which were less than 20.66 mm (for the forward kinematics) and error percentages not greater than 4% (for the inverse kinematics). In addition, the *segmented* and *continuous* modes to execute trajectories were compared, from which it was concluded that the continuous mode has a greater position error compared to the segmented mode, however it has greater end-effector's linear speed continuity and performs trajectories more effectively timewise.

Subsequently, a wireless communication module was created, with the intention of integrating the robotic arm into the Robotat ecosystem. This module is an embedded system that has a printed circuit board and a custom designed and manufactured protection case. The PCB has an ESP32 microcontroller, a pair of Li-ion 18650 batteries and a circuit that converts the lines' levels of a series RS-232 port to TTL levels and vice versa. The correct operation of this embedded system was validated by measuring and analyzing the output signal with an oscilloscope. Finally, it was demonstrated that the module indeed allows the R17 to be integrated into the Robotat ecosystem, through inverse kinematics and trajectory execution tests, being the robot observed by the OptiTrack.

En la Universidad del Valle de Guatemala han habido avances en el ámbito de la robótica. En años pasados se ha trabajado con el manipulador serial R17 para dotarlo de movimiento, como lo fue en el caso del trabajo de graduación *MEGAPROYECTO GC-R17: Sistema háptico de control a distancia*, en el que fue posible controlar el brazo robótico mediante a un guante utilizado por un usuario. Hace no más de cinco años se viene trabajando también con lo que se conoce como *robótica de enjambre*, el cual es un concepto donde se busca la cooperación de robots entre ellos, con el objetivo de desempeñar una tarea; un ejemplo de ello es el trabajo de graduación del Ing. Mecatrónico José Andrés Castañeda Forno, *Diseño e implementación de una red de comunicación wifi e interfaz gráfica para una mesa de pruebas de robótica de enjambre*. El proyecto presentado en este trabajo de graduación no es más que la culminación de integrar estas dos ideas, con el agregado de otorgar al manipulador serial cierto nivel de automatización, el cual se logra gracias a el sistema de captura de movimiento OptiTrack.

En este trabajo se presenta el proceso llevado a cabo para el diseño e implementación de software, capaz de controlar un manipulador serial R17 dentro de un ecosistema que utiliza al OptiTrack. El desarrollo de este trabajo se divide en tres bloques principales: familiarizarse con el sistema de captura de movimiento OptiTrack para obtener información acerca de la configuración del R17 y la pose de su efector final; obtener el modelo cinemático del R17 e implementar un paquete de herramientas de software que permitan simularlo y controlarlo; y desarrollar un módulo de comunicación inalámbrica que permita integrar al brazo robótico dentro del ecosistema Robotat, basado en captura de movimiento.

Cada uno de estos bloques principales son evaluados por separado. Para el primero se realizan pruebas de obtención de la pose del efector final, otorgadas por el OptiTrack. Con respecto al segundo bloque, se compara de manera cualitativa y cuantitativa el comportamiento del robot para poder concluir acerca del modelo cinemático planteado. Finalmente, se verifica el funcionamiento del módulo inalámbrico desarrollado y se efectúan pruebas de cinemática inversa y ejecución de trayectorias dentro del ecosistema Robotat.

Los manipuladores seriales, más comúnmente denominados brazos robóticos, son dispositivos de gran utilidad, precisión y confiabilidad para realizar tareas repetitivas y precisas, por lo cual son muy utilizados en la industria. Requisito de poder utilizar estos robots es poder conocer a profundidad la cinemática involucrada en el posicionamiento y orientación de su efector final. Parte de la precisión que estos dispositivos pueden presentar dependen de los sistemas de control implementados. Estos pueden ser en las juntas del mismo, por ejemplo el uso de servomotores, o bien mejorándolo mediante a la implementación de un lazo de control cerrado por medio de captura de movimiento.

Existen múltiples formas de modelar la cinemática de un manipulador serial, una muy utilizada se basa en la denominada convención de Denavit-Hartenberg (DH) y otra en métodos basados en la teoría de tornillos. En [1] se describe la metodología para el modelado y cálculo de la cinemática directa de un manipulador serial de 6 grados de libertad utilizando la convención DH. Otro ejemplo es [2], donde se plantea la cinemática según métodos basados en la teoría de tornillos y se desarrolla un algoritmo genético (AG) para el cálculo de la cinemática inversa de un manipulador serial Melfa RV-2A .

En el artículo [3] se presenta la integración de un sistema de captura de movimiento para el control de un brazo robótico dentro de un área de trabajo de $40 m^2$, con una recolección de datos por el sistema a una frecuencia de muestreo de 100 Hz. En este trabajo se concluyó que el sistema de captura de movimiento se presenta como una buena alternativa para obtener la pose del manipulador y tomarlo como retroalimentación, además de que este es fácil de instalar y tiene un área de cobertura amplia, aunque esto depende mucho de la cantidad de cámaras utilizadas. Para el uso de seis cámaras se obtuvieron resultados de un error medio de 0.0140 m . En el año 2014 la Escuela Politécnica Federal de Lausana (EPFL) desarrolló un manipulador serial capaz de atrapar objetos que le eran lanzados. El entorno de trabajo donde se desempeñaba este manipulador era cubierto por el sistema de captura de movimiento OptiTrack el cual obtenía la información acerca de la pose del objeto. Con esta información, un algoritmo se encargaba de predecir la trayectoria del objeto, adicionalmente de planificar la ruta que el brazo robótico debía de seguir para atraparlo [4].

Estos trabajos indican que es posible y factible poder incorporar un brazo robótico dentro de un entorno de sistema de captura de movimiento. En este podrá interactuar con demás objetos adicionalmente a poder obtener retroalimentación para un lazo de control referente a la pose de su efector final, todo esto mediante al previo modelado cinemático del mismo.

Por otra parte, en el año 2009 se realizó el trabajo de graduación *MEGAPROYECTO GC-R17: Sistema háptico de control a distancia*, desarrollado por un grupo de estudiantes de la facultad de ingeniería de la Universidad del Valle de Guatemala, donde se trabaja directamente con el manipulador serial R17 de ST Robotics. El objetivo final del megaproyecto era para poder controlar el brazo robótico mediante a un guante utilizado por un usuario. En este trabajo se desarrolló una interfaz que trabaja con los comandos del lenguaje ROBOFORTH II, donde la comunicación era alámbrica entre la computadora y el robot. En este también se detalló el proceso adecuado para poder establecer la comunicación entre la computadora y el R17 por medio del puerto serial de la PC y se describen algunos comandos útiles en ROBOFORTH II para el posicionamiento del efector final en coordenadas cartesianas deseadas [5].

Parte del presente trabajo de graduación implica la integración del robot R17 dentro del denominado ecosistema Robotat, el cual emplea el *mocap* OptiTrack. Este entorno comenzó como una inspiración del denominado Robotarium, del Georgia Institute of Technology. El Robotarium es un proyecto que permite el acceso remoto de cualquier persona a una plataforma de pruebas de robótica de enjambre. En este entorno se emplea robots colaborando mutuamente entre ellos para desempeñar una tarea, todo esto monitoreado mediante a un sistema de captura de movimiento. Parte del objetivo final de el Robotarium es incentivar el aprendizaje de la robótica, por lo cual se creó este espacio accesible a cualquier persona, ya que este campo resulta ser bastante costoso [6].

El ecosistema Robotat se basa en la implementación de una red de comunicación inalámbrica, la cual se comenzó en el trabajo de graduación del Ing. Mecatrónico José Andrés Castañeda Forno, *Diseño e implementación de una red de comunicación wifi e interfaz gráfica para una mesa de pruebas de robótica de enjambre*. En esta tesis se desarrolla una red de comunicación por medio de protocolo MQTT, utilizando el módulo ESP8266-12e y Micropython. Se concluyó que la implementación de multithreading mejoró el rendimiento y escalabilidad, además que el protocolo MQTT proporciona bastante flexibilidad en el diseño de la red de comunicación [7]. Este trabajo sienta las bases para poder integrar el manipulador R17 en el ecosistema basado en captura de movimiento mediante la comunicación inalámbrica.

Los manipuladores seriales son herramientas importantes en el mundo moderno. Estos se caracterizan por su precisión, confiabilidad, seguridad y resistencia, lo cual los han hecho ideales para todo tipo de trabajos. Un ejemplo claro se ve en la industria, donde algunas de las ventajas que estos presentan son: aumento en el rendimiento y producción, reducción de costos a largo plazo, mejora en la calidad del producto, evitar lesiones en los trabajadores por realizar actividades que involucran esfuerzos repetitivos, entre otras. Ejemplos puntuales son la industria automotriz y la de fabricación de PCB's, donde el uso de estos robots es indispensable.

En países de primer mundo, el uso de robots manipuladores no es algo nuevo, por lo que es común encontrarlos en casi cualquier industria. Países desarrollados ven como siguiente paso el trabajo con robots más avanzados, que logren trabajar en conjunto con demás robots. Lastimosamente, países en vías de desarrollo como Guatemala, jamás se han adentrado siquiera en la introducción de los manipuladores. Mientras el mundo se prepara para el siguiente gran paso en la revolución de la robótica en la industria, Guatemala no se ha adaptado a la tecnología de los manipuladores seriales. Esto que anteriormente se veía como sugerencia, comienza a tornarse más como una necesidad que debe de adquirir el país para poder competir en la economía a nivel internacional.

Todo lo mencionado anteriormente da una pequeña idea de la importancia y del papel que juegan los brazos robóticos en la actualidad, por lo cual vale la pena estudiarlos, modelarlos y experimentar con ellos. Además, tal como se mencionó, el mundo moderno no tardará en desarrollar nueva tecnología, la cual naturalmente toma como base los avances hechos en el pasado, como lo podrán ser los manipuladores. Por esta razón es importante que países como Guatemala se pongan al día en la participación de la robótica en la industria, sobre todo introduciendo y haciendo más común el uso de brazos robóticos, de tal forma que pueda estar preparada para la siguiente gran revolución tecnológica.

En el pasado, tal como se indicó en los antecedentes, se ha trabajado en la Universidad del Valle de Guatemala con el manipulador serial R17 de ST Robotics, con el objetivo de dotar al robot de movimiento. No obstante, nunca se le ha adjudicado de autonomía, es

decir, lograr que el mismo robot pueda efectuar tareas sin tener que controlarlo de manera remota, sino únicamente con la información que adquiere de su entorno, como lo puede ser mediante a un sistema de captura de movimiento. Por lo tanto, un paquete informático que permita la generación y planificación de trayectorias en el robot serial R17 presenta el siguiente gran paso en el estudio y experimentación de manipuladores en la Universidad del Valle de Guatemala.

En años pasados la Universidad del Valle de Guatemala ha estudiado el desarrollo de algoritmos y métodos de control para grupos grandes de robots, a lo cual se le conoce como robótica de enjambre. Además, se ha enfocado en el desarrollo de una mesa de pruebas donde los robots pueden ejecutar los algoritmos. A esta mesa de pruebas se le ha llamado el *Robotat* y su uso se ha limitado a robots móviles de pequeño tamaño. Dentro del *Robotat* no se ha contemplado aún la interacción que puede haber entre varios robots de distintos tipos, como manipuladores seriales y drones. Actualmente la Universidad está trabajando en ampliar el alcance de esta mesa de pruebas, cuya idea general es crear un área donde diversos robots, no solo móviles, puedan interactuar y ejecutar tareas de manera coordinada, todo esto gracias a la introducción de sistema de captura de movimiento Optitrack. A este nuevo espacio se le denomina el *ecosistema Robotat*.

Por lo mencionado anteriormente, la realización de este trabajo de graduación presenta la integración del manipulador serial R17 al ecosistema *Robotat*. Esta integración se llevará a cabo mediante a un paquete informático que permita obtener información acerca de la pose del efector final del R17 (adquirido por el sistema de captura de movimiento Optitrack) y generar trayectorias, todo esto mediante a un control inalámbrico dentro del mismo ecosistema. De esta manera se logrará agregar el estudio y pruebas de robots tan importantes y útiles como lo son los manipuladores seriales en un entorno con diferentes tipos de robots.

4.1. Objetivo general

Crear e implementar un paquete de herramientas de software que permita ejecutar trayectorias en el manipulador serial R17 de ST Robotics dentro de un ecosistema basado en el sistema de captura de movimiento OptiTrack.

4.2. Objetivos específicos

- Utilizar el sistema de captura de movimiento OptiTrack para obtener información acerca de la configuración del manipulador serial R17 y la pose de su efector final.
- Obtener el modelo cinemático del manipulador serial R17, bajo el estándar de Denavit-Hartenberg, que permita su simulación y animación en software.
- Desarrollar e implementar una interfaz de usuario que permita planificar y visualizar trayectorias del manipulador serial R17, además de que sea capaz de comunicarse con él mediante a su lenguaje de operación ROBOFORTH II.
- Implementar un módulo de comunicación inalámbrica que permita la integración del manipulador R17 al ecosistema Robotat.

Este trabajo de graduación se centra en el desarrollo de 3 bloques principales. El primer bloque consiste en familiarizarse con el sistema de captura de movimiento OptiTrack para obtener información acerca de la configuración del R17 y la pose de su efector final. El segundo es obtener el modelo cinemático del R17 e implementar un paquete de herramientas de software que permitan simularlo y controlarlo; esto fue realizado con el software MATLAB y se limitó a diseñar y probar una interfaz gráfica de usuario que permitiera manejar al manipulador serial por cinemática directa, inversa y planificación de trayectorias. Por último, el tercer bloque, se desarrolló un módulo de comunicación inalámbrica que permite integrar al brazo robótico dentro del ecosistema Robotat, basado en captura de movimiento; esto involucró el diseño electrónico del sistema embebido y la creación de librerías de software; pruebas de cinemática inversa y ejecución de trayectorias validaron esta integración.

Cabe aclarar que el último bloque de este proyecto se enfoca en evaluar la factibilidad de utilizar el manipulador serial dentro del entorno del Robotat, no en generar una aplicación final de uso. Visto desde otro ángulo, el último bloque tiene el propósito de levantar la infraestructura que permitirá la integración entre el R17 y el ecosistema. Por otro lado, a lo largo de este trabajo se hace mención al *Gripper* del R17; este tiene la capacidad de añadirse al manipulador, no obstante el diseño e implementación de este no forma parte de este trabajo de graduación, sino que fue trabajado por el estudiante Rony Schumann [8].

Es importante resaltar que debido a la pandemia del COVID-19, el tiempo en el campus se limitó a una fracción de lo que se tenía planeado inicialmente. Por esta razón, el tiempo invertido en las pruebas realizadas fue sujeto a disposiciones gubernamentales y universitarias.

6.1. Sistema de captura de movimiento

El sistema de captura de movimiento, también conocido como *mocap*, es el proceso de grabar el movimiento de personas u objetos. Esto incluye la medición y almacenamiento de la información de tanto la posición como la orientación del objeto analizado por el sistema [9]. En la actualidad el proceso de captura de movimiento tiene muchas aplicaciones, entre las más populares están:

- Animación: el proceso implica el registro en los cambios de posiciones de ciertos puntos del objeto o persona que quiere ser animada, creando un marco de referencia y un “esqueleto” o cuerpo rígido en software. Esto es empleado tanto en los videojuegos como en el cine y tiene el objetivo final de darle un movimiento mucho más real y fluido a la animación.
- Producción virtual: es una técnica de filmación donde se combina la grabación en vivo con efectos de animación de computadora en tiempo real para lograr efectos visuales más complejos. El proceso consiste en filmar desde múltiples localidades y entregar retroalimentación digital. El mocap en esta aplicación permite efectuar seguimiento de personajes u objetos. Todo esto permite el ahorro de tiempo y dinero en post-producción y viajes a diferentes localidades para filmar.
- Ciencias de la vida y la salud: utilizado en estudio de desempeño deportivo, para poder analizar el movimiento biomecánico del cuerpo, permitiendo diagnosticar, sugerir movimientos adecuados para rehabilitación y medir el desempeño de los atletas para alcanzar objetivos. También es utilizado para el estudio del movimiento de animales, los cuales en muchos casos son de utilidad en estudios de robótica bioinspirada.
- Robótica: empleado gracias al rastreo de los 6 grados de libertad (DOF) de uno o más objetos en un espacio de trabajo. Esto permite obtener información de retroalimentación para implementación de sistemas de control en los robots.

Existen muchas formas de clasificar los tipos de sistemas de captura de movimiento. En este trabajo se hará la distinción más general entre los mocap basados en uso de marcadores (marker-based) y los que no utilizan marcadores (markerless).

6.1.1. Marker-based

Sistemas acústicos:

Consiste en un conjunto de audio transmisores que se posicionan en las articulaciones del objeto del cual se desea capturar su movimiento. Estos transmisores emiten sonidos a frecuencias específicas, las cuales son captadas por receptores. Cabe mencionar que estos transmisores son activados de forma secuencial cada cierto tiempo. La forma de calcular la posición de estos marcadores acústicos es mediante a una triangulación de posición la cual utiliza el tiempo que tarda en llegar la señal hasta los receptores. Algunas de las desventajas que este método presenta son la dificultad de obtener la posición en un momento específico y la susceptibilidad al ruido exterior. La primera se debe a la naturaleza secuencial de emitir los sonidos, lo cual genera una descripción no muy fluida de los objetos [10].

Sistemas mecánicos:

Utiliza potenciómetros y sliders en articulaciones deseadas (por lo general se utiliza un exoesqueleto de metal o plástico). Tiene ventajas como un costo relativamente bajo, no se ve afectado por campos magnéticos o reflexiones de luz y no necesita un proceso de calibración muy largo [10].

Sistemas magnéticos:

Utiliza receptores en las articulaciones que permiten medir la posición y orientación con respecto a una antena. Presenta desventajas como la necesidad de una gran cantidad de cables (dificultando la movilidad del usuario) y que es sensible a interferencia de campos magnéticos causados por estructuras u objetos metálicos en los alrededores. No obstante, este sistema presenta un costo relativamente bajo [10].

Sistemas ópticos:

Estos son los sistemas mayormente utilizados en captura de movimiento. En este tipo de sistema, el actor u objeto poseen un traje con marcadores que reflejan o emiten luz. El sistema cuenta con una serie de cámaras de alta resolución posicionadas en puntos estratégicos, las cuales permiten rastrear los movimientos de los marcadores. Las ventajas que hacen a este tipo de sistemas tan atractivos son la alta frecuencia de muestreo, lo cual permite capturar movimientos más reales y complejos, y la facilidad de movimiento en los actores, gracias a que no se necesita de mucho o incluso ningún cableado (depende de si se utilizan marcadores activos o pasivos) [10]. Sin embargo, este tipo de sistema también posee algunas desventajas.

La primera desventaja significativa es la de problemas de oclusión, es decir, que un objeto en un espacio tridimensional tape a otro objeto desde un punto de vista específico. Este problema puede ser solucionado mediante a reconstrucción 3D posterior o agregando más cámaras, no obstante, esto incrementa el costo del sistema. Otra desventaja es que el costo es relativamente más caro, en comparación con todos los sistemas mencionados anteriormente [\[11\]](#).

Existen dos enfoques en lo que al sistema óptico de captura de movimiento con marcadores se refiere, estos son el enfoque óptico-pasivo y el óptico-activo.

- **Óptico-pasivo:** utiliza marcadores que están cubiertos por un material reflectante, los cuales reflejan la luz de regreso a las cámaras infrarrojas las cuales realizan el rastreo. Este es el método más utilizado en la industria, ya que tiene la ventaja de no cargar fuentes de alimentación ni cables en los trajes de captura de movimiento.
- **Óptico-activo:** este tipo de implementación utiliza marcadores LED, los cuales emiten su propia luz, la cual es captada por las cámaras para efectuar el rastreo. La desventaja es que se necesita de algún tipo de alimentación en los marcadores, lo cual implica mayor peso en el traje y la utilización de cables. No obstante, presenta la ventaja que aumenta el espacio de captura.

6.1.2. Markerless

Este tipo de sistema no necesita de ningún tipo de marcador para grabar el movimiento, tal como lo indica su nombre. Todo el proceso de rastreo e identificación de formas se lleva a cabo dentro del software. A pesar que este tipo de sistema permite capturar y reproducir el desempeño y forma de movimiento de algún objeto o persona utilizando únicamente cámaras convencionales y sin necesidad de marcadores o trajes especiales (reduciendo los costos), a la fecha no se ha presentado ningún mocap markerless que tenga mejor capacidad que los basados en uso de marcadores [\[12\]](#).

6.1.3. OptiTrack

OptiTrack es uno de los más grandes proveedores de sistemas de captura de movimiento en todo el mundo. Ofrece equipo y software para realizar *mocap* de alto desempeño por medio de técnicas óptico-pasivas. El sistema *OptiTrack* incluye software para captura de movimiento (Motive) y cámaras de rastreo de alta velocidad. Este sistema es utilizado por grandes empresas, organizaciones y universidades, como Activision, Disney, Google, MIT, entre otros [13]. Este será el sistema con el que se estará desarrollando este trabajo de graduación.

Cámaras

OptiTrack ofrece una amplia variedad de cámaras, las cuales cumplen con la función del rastreo de los marcadores del sistema. Podemos encontrar modelos económicos y de alto desempeño. Algunos de los modelos se muestran en la Figura 1.

En este trabajo de graduación se estará utilizando la cámara de alta precisión *Prime^x 41*.

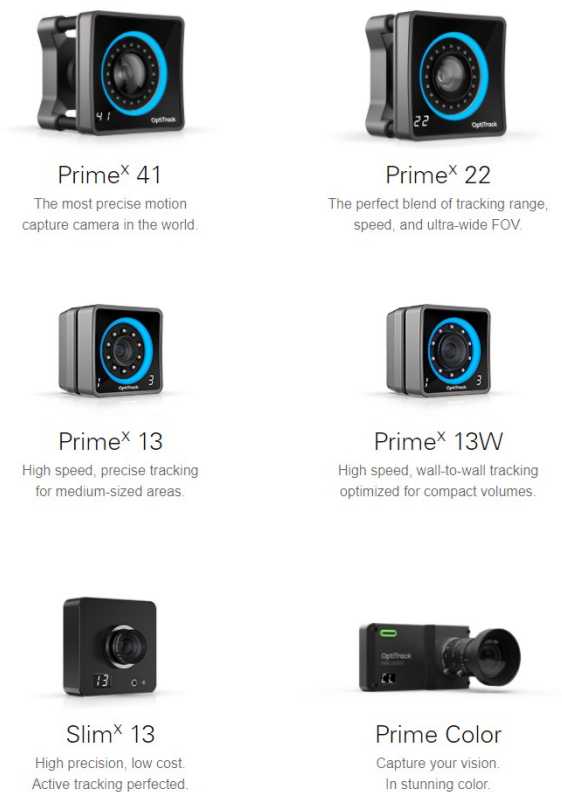


Figura 1: Cámaras ofrecidas por el sistema Optitrack [14].

Prime^x 41

Esta es la cámara más precisa y completa de toda la línea que ofrece *OptiTrack*. Dentro de las especificaciones relevantes, se encuentran las mostradas en los Cuadros 1 y 2 [15].

Imagen	
Resolución	2048 x 2048
Latencia	5.5 ms
Frame rate por defecto	180 fps
Frame rate máximo	250 fps

Cuadro 1: Especificaciones de imagen de Prime^x 41 [15].

Desempeño de rastreo y alcance	
Exactitud 3D	0.10 mm
Alcance (marcadores pasivos)	30 m

Cuadro 2: Especificaciones de desempeño de rastreo y alcance de Prime^x 41 [15].

Software *Motive*

Motive corresponde al programa que brinda *Optitrack* para poder efectuar el proceso de captura de movimiento. Este cuenta con diferentes licencias pagadas, las cuales están hechas para diferentes aplicaciones. Podemos encontrar las siguientes:

- **Motive Tracker:** está pensado para el rastreo de objetos en 6 grados de libertad (DoF). Efectúa reconstrucción 3D y resolución de cuerpo rígidos (Rigid Body Solving). Utiliza filtros de seguimiento y restricciones para poder ajustar el rendimiento del objeto que está siendo rastreado. Este software es ideal para aplicaciones exigentes donde se utiliza un número elevado de objetos y de alta velocidad. Cabe mencionar que cuenta con una arquitectura bastante flexible que le permite trabajar con otras plataformas gracias a los SDK's y API's disponibles por *Motive* [16].
- **Motive Body:** utilizado para la captura de movimiento de sujetos. Este genera y calibra lo que se conoce como el esqueleto. Permite trabajar con varios sujetos a la vez y, cuando se trabaja con cámaras de la serie Prime, permite el seguimiento preciso de los movimientos de los dedos. [16].

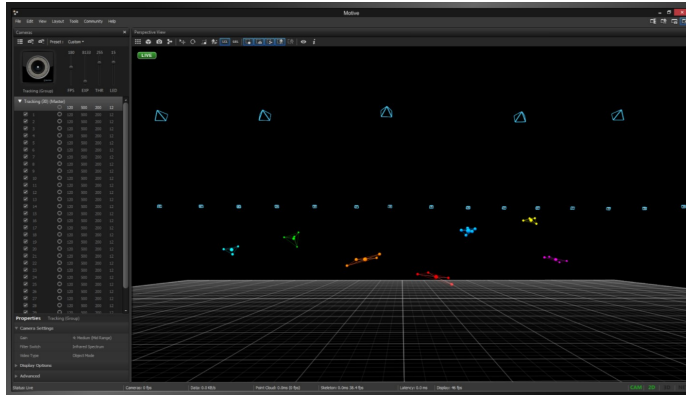


Figura 2: Motive Tracker [16](#).

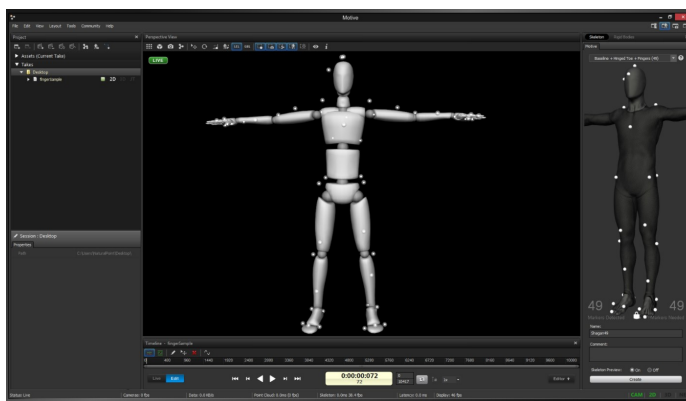


Figura 3: Motive Body [16](#).

6.2. Robótica

La robótica es una rama de la ciencia interdisciplinaria que estudia el diseño e implementación de robots, combinando conceptos de la mecánica, electrónica, informática, inteligencia artificial, control, entre otras.

Un robot en términos generales es una máquina programable capaz de percibir su entorno e imitar ciertos comportamientos de una criatura inteligente. Un robot está mecánicamente construido por eslabones, los cuales están unidos entre sí por los denominados como juntas, siendo diferente a los mecanismos porque los robots tienen más actuadores. Estos eslabones son modelados por lo general como cuerpos rígidos [17](#). Existen muchos tipos de robots, no obstante en las siguientes secciones se enfatizará en el estudio de aquellos robots denominados como *manipuladores seriales*.

Manipuladores seriales

Un manipulador serial es un robot de base fija que tiene como objetivo final manipular objetos. Estos están formados por cadenas cinemáticas abiertas, es decir, sus elementos se encuentran conectados uno a uno en serie y todas sus juntas son activas o actuadas. Los robots manipuladores seriales son herramientas muy útiles y eficaces para realizar tareas que demandan de precisión y que son repetitivas, por lo que se busca conocer siempre la posición y orientación del efector final (*pose*). En la mayoría de los casos, estos tienen la facilidad de no poseer restricciones adicionales en el *espacio de configuración*, lo cual permitirá describir su cinemática de manera más simple [18].

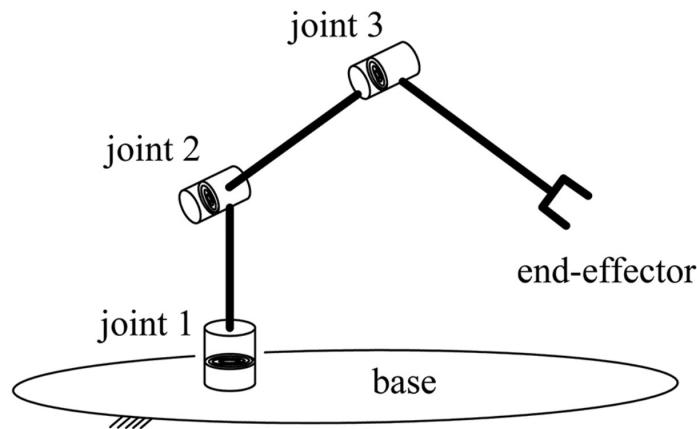


Figura 4: Diagrama cinemático de un manipulador serial.

Espacio de configuración

La pregunta más importante que se tiene que hacer cuando se habla de robots es posiblemente ¿dónde está? Esta pregunta se ve contestada por la configuración.

La configuración de un robot es la especificación de todos los puntos de un robot. Por otra parte, el espacio de configuración, también abreviado como *C-space*, es el conjunto de todas las posibles configuraciones que puede tomar el robot, es decir, corresponde al generado del vector de configuración [17].

Grados de libertad (DoF)

Los grados de libertad son el número mínimo de parámetros necesarios para poder especificar la configuración del robot. Un cuerpo rígido en un espacio planar cuenta con 3 grados de libertad y uno en un espacio tridimensional cuenta con 6 DoF. Recordando que los robots son conjuntos de cuerpos rígidos unidos entre si por juntas, es lógico pensar que los DoF del robot dependen del espacio donde se mueven estos cuerpos rígidos y del tipo de juntas que unen a los eslabones. Esta relación está dada por la fórmula de Grüebler [1].

$$DoF = m(N - 1 - J) + \sum_{i=1}^J f_i \quad (1)$$

donde $m = 3$ para mecanismos planares y $m = 6$ para mecanismos espaciales, N es el número de eslabones, J es el número de juntas y f_i es el número de DoF de cada junta i .

La fórmula de Grübler (1) indica entonces claramente que los DoF del robot dependen del tipo de junta que una los eslabones. Estas juntas por si mismas poseen grados de libertad, por ejemplo una *junta revoluta* permite únicamente girar a su alrededor, por lo tanto posee un DoF. Dentro de las juntas más utilizadas en robots se encuentran las mostradas en la Figura 5.

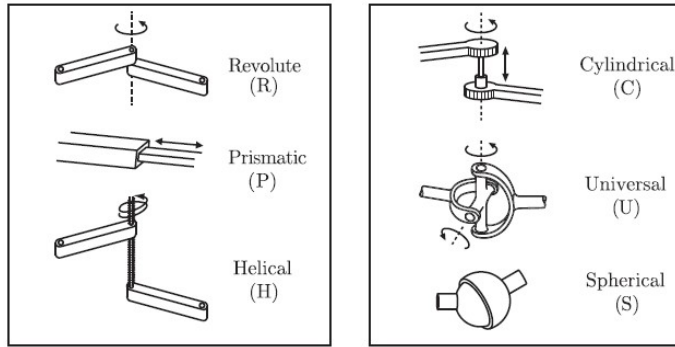


Figura 5: Juntas comúnmente utilizadas en robots [17].

Estas juntas pueden ser analizadas desde el enfoque de proveedoras de grados de libertad del movimiento de un cuerpo rígido con respecto de otro o como restricciones de movimiento de los dos cuerpos rígidos que une. La Figura 6 muestra los grados de libertad f de cada tipo de junta de la Figura 5 que se utiliza en (1).

Joint type	dof f	Constraints c between two planar rigid bodies	Constraints c between two spatial rigid bodies
Revolute (R)	1	2	5
Prismatic (P)	1	2	5
Helical (H)	1	N/A	5
Cylindrical (C)	2	N/A	4
Universal (U)	2	N/A	4
Spherical (S)	3	N/A	3

Figura 6: Información sobre grados de libertad en las juntas típicas de un robot [17].

Los grados de libertad se encuentran directamente relacionados con el espacio de configuración. Resulta ser que su dimensión corresponde al número de DoF del robot, es decir, si el C -space tiene dimensión n , entonces el robot tiene n grados de libertad.

Espacios de tarea y trabajo

Ambos se encuentran relacionados con la configuración del efector final, no de todo el robot. Las definiciones de estos conceptos se indican a continuación.

- Espacio de tarea: es el espacio en donde la tarea del robot puede expresarse de forma natural. Esto implica que para definir el espacio de tarea no es necesario conocer el robot como tal, únicamente la tarea que se desea realizar [17].
- Espacio de trabajo: es una especificación de la configuración que el robot puede alcanzar en el efector final. Contrario al espacio de tarea, este espacio si toma en cuenta las restricciones físicas que tiene el robot. De manera simple se podría decir que es el espacio de configuración del robot ya implementado [19].

6.2.1. Cinemática de cuerpos rígidos

Se necesita de un mínimo de 6 parámetros para poder representar la posición y orientación de un cuerpo rígido (*pose*) en un espacio tridimensional; para un cuerpo en un espacio planar se necesita de al menos 3. Siendo un robot un conjunto de estos cuerpos rígidos, se necesitaría de muchos parámetros para poder describir por completo su configuración, no obstante, existe una forma más simple de hacer esto. Se colocará un sistema de coordenadas inercial en las uniones de los cuerpos rígidos, los cuales son referenciados a un marco de referencia fijo. En el caso de los manipuladores seriales es de mayor interés la pose del efector final, por lo tanto, mediante al método que se describirá a continuación, se podrá conocer esta *pose* usando una o varias matrices de transformación (de 4×4 en el caso tridimensional). Esto es una gran ventaja porque permite utilizar álgebra lineal simple como una herramienta [17].

Marcos de referencia

Un marco de referencia es el origen de un sistema de coordenadas, este se denotará como una letra entre llaves. Siempre se asume que existe un marco de referencia fijo, al cual se llamará $\{s\}$. También será de utilidad en algunos casos definir un marco al cuerpo rígido, el cual se llamará *body frame* y se denotará como $\{b\}$.

Definir estos marcos es importante, ya que en la robótica los sensores obtienen información con respecto a el punto en donde se encuentran y no con respecto al origen. Esto es un problema, ya que un punto en un espacio físico puede ser representado por un vector, pero este cambiará según la referencia que se esté tomando. Es decir, un marco de referencia diferente otorga configuraciones diferentes al mismo punto en un espacio físico. Esto se muestra en la Figura 7, donde el mismo punto p se ve representado por diferentes vectores p_a y p_b . Por esta razón se quiere encontrar una manera de poder referir puntos con respecto a un marco $\{b\}$ a un marco $\{a\}$, por ejemplo. Esto se logrará gracias a la transformaciones lineales como rotaciones y traslaciones, las cuales se verán compactadas en lo que se conoce como *matriz de transformación homogénea*.

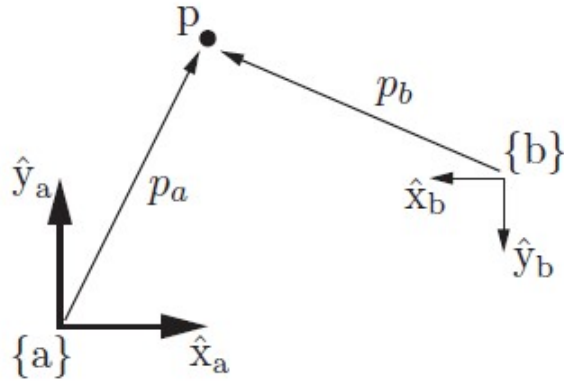


Figura 7: Representación de un mismo punto p con respecto a dos marcos de referencia diferentes $\{a\}$ y $\{b\}$ [17].

Rotación y traslación de un cuerpo rígido en un espacio 2D

Para describir la configuración de un cuerpo rígido se necesita únicamente conocer su pose, es decir, su ubicación y orientación con respecto a un marco de referencia fijo.

En un caso de dos dimensiones, la ubicación es algo trivial de obtener, pues solo se debe de conocer las coordenadas x y y del *body frame* del cuerpo rígido. El caso de la orientación no es tan simple, ya que se necesita conocer del parámetro adicional θ , el cual describe el ángulo de rotación del *body frame* con respecto a $\{s\}$.

De la Figura 8 resulta evidente que la ubicación de $\{b\}$ representado por el vector p está dado por (2) y la rotación por (3) y (4).

$$p = p_x \hat{x}_s + p_y \hat{y}_s \quad (2)$$

$$\hat{x}_b = \cos \theta \hat{x}_s + \sin \theta \hat{y}_s \quad (3)$$

$$\hat{y}_b = -\sin \theta \hat{x}_s + \cos \theta \hat{y}_s \quad (4)$$

Basado en (2) es claro entonces que si se tuviera un punto de interés medido desde el marco de referencia de $\{b\}$, su posición con respecto de $\{s\}$ no sería más que la suma vectorial entre la posición del punto respecto a $\{b\}$ y la posición del origen de $\{b\}$ con respecto de $\{s\}$. Esto corresponde a una *traslación pura*.

La rotación pura está dada por lo que se conoce como una matriz de rotación, la cual se puede extraer de (3) y (4) si se considera a el par (\hat{x}_b, \hat{y}_b) y (\hat{x}_s, \hat{y}_s) como vectores columna que denotan un cambio de coordenadas del marco $\{s\}$ al $\{b\}$.

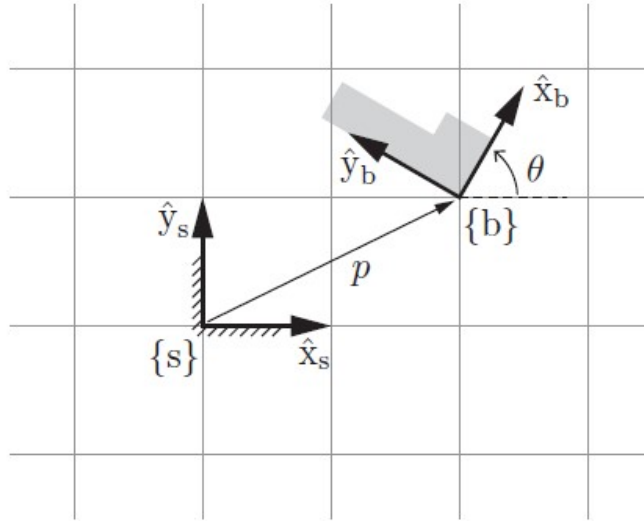


Figura 8: Configuración del cuerpo rígido con body frame $\{b\}$ con respecto a el sistema fijo $\{s\}$ [17].

Notación: a partir de este punto se utilizará la notación estándar en robótica. En ella se describe a un vector \mathbf{x} representado en un marco de referencia $\{s\}$ como ${}^s\mathbf{x}$. En matrices de rotación, se puede expresar el marco destino como superíndice del lado izquierdo y el marco fuente como subíndice derecho. Por ejemplo, la matriz de rotación que cambia las coordenadas del marco $\{a\}$ al $\{b\}$ se escribe como ${}^b\mathbf{R}_a$.

Utilizando la notación estándar y tomando el caso de traslación pura que se presenta en la Figura 9 entonces se obtiene (5).

$${}^I\mathbf{p} = {}^B\mathbf{p} + {}^I\mathbf{o}_B \quad (5)$$

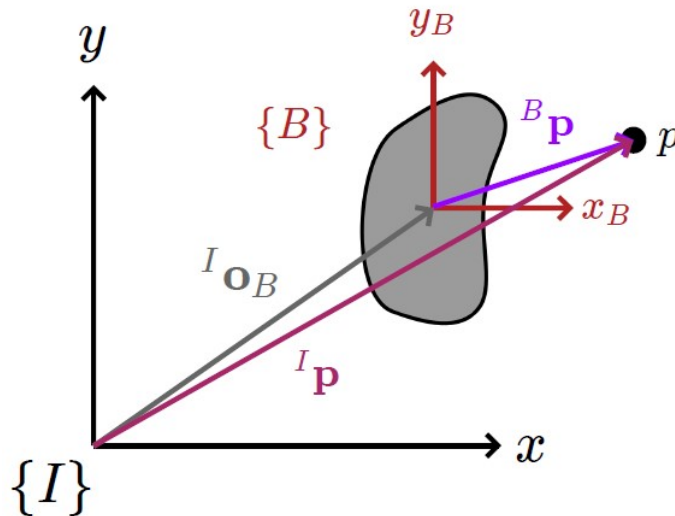


Figura 9: Traslación pura en 2D [20].

En el caso presentado en la Figura 10, se tiene la rotación pura donde se desea expresar al punto p medido en el cuerpo rotado con *body frame* $\{B\}$, en el sistema de coordenadas dado por $\{I\}$.

Si se definen los vectores columna (6) y (7), resulta evidente (8), lo cual da como resultado (9), donde ${}^I\mathbf{p}$ es el vector que expresa al punto p en coordenadas del marco inercial, ${}^B\mathbf{p}$ es vector que expresa a p en coordenadas del marco del cuerpo rígido rotado y ${}^I\mathbf{R}_B$ es la matriz de rotación que efectua esta transformación de coordenadas.

$${}^I\mathbf{p} = \begin{bmatrix} {}^I p_x \\ {}^I p_y \end{bmatrix} \quad (6)$$

$${}^B\mathbf{p} = \begin{bmatrix} {}^B p_x \\ {}^B p_y \end{bmatrix} \quad (7)$$

$$\begin{bmatrix} {}^I p_x \\ {}^I p_y \end{bmatrix} = \begin{bmatrix} {}^B p_x \cos \theta - {}^B p_y \sin \theta \\ {}^B p_x \sin \theta + {}^B p_y \cos \theta \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} {}^B p_x \\ {}^B p_y \end{bmatrix} \quad (8)$$

$${}^I\mathbf{p} = {}^I\mathbf{R}_B {}^B\mathbf{p} \quad (9)$$

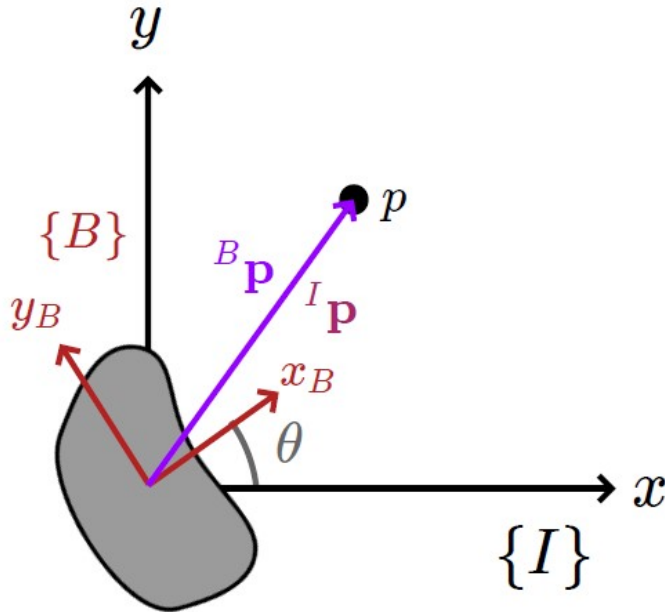


Figura 10: Rotación pura en 2D [20].

Como se mencionó anteriormente, la pose corresponde tanto a la ubicación como a la orientación, la cual se puede obtener al superponer los movimientos de traslación y rotación pura. Resulta entonces que esta combinación puede cumplir con varios propósitos:

- Representar la configuración de un cuerpo rígido en un marco de referencia fijo.
- Cambiar el marco de referencia en el que un vector u otro marco está representado.
- Desplazar un vector o marco.

Esta superposición de movimientos de rotación y traslación pura da origen a lo que se conoce como la transformación rígida, la cual está dada por (10).

$${}^I\mathbf{p} = {}^I\mathbf{R}_B {}^B\mathbf{p} + {}^I\mathbf{o}_B \quad (10)$$

Con el objetivo de poder emplear álgebra matricial, se asocia a esta transformación lo que se conoce como una matriz de transformación homogénea y se expanden los vectores agregando un 1 en la última posición. Se tienen entonces a los vectores representados en sus coordenadas homogéneas. La notación en coordenadas homogéneas para un vector ${}^I\mathbf{p}$ será ${}^I\tilde{\mathbf{p}}$. Entonces (10) se puede arreglar como un producto matricial según se muestra en (11) para simplificarlo como (12), donde ${}^I\mathbf{T}_B$ corresponde a la matriz de transformación homogénea de un cuerpo rígido en un espacio 2D.

$$\begin{bmatrix} {}^I p_x \\ {}^I p_y \\ 1 \end{bmatrix} = \begin{bmatrix} {}^I\mathbf{R}_B & {}^I\mathbf{o}_B \\ \mathbf{o}_{1 \times 2} & 1 \end{bmatrix} \begin{bmatrix} {}^B p_x \\ {}^B p_y \\ 1 \end{bmatrix} \quad (11)$$

$${}^I\tilde{\mathbf{p}} = {}^I\mathbf{T}_B {}^B\tilde{\mathbf{p}} \quad (12)$$

Es importante destacar que dentro de ${}^I\mathbf{T}_B$ no solo está lo necesario para efectuar la transformación de un vector del marco de $\{B\}$ a $\{I\}$, sino también se encuentra la pose del cuerpo rígido representado por el *body frame* $\{B\}$ con respecto de $\{I\}$.

Rotación y traslación de un cuerpo rígido en un espacio 3D

El caso de rotación y traslación de un cuerpo rígido en 3D se complica de manera significativa comparado con el de uno en 2D. En dos dimensiones, el cuerpo cuenta con 3 DoF, en cambio en un espacio tridimensional el cuerpo cuenta con 6 DoF. El caso de la traslación no es muy complicado, pues basta con las coordenadas x , y y z con respecto al marco de referencia. La dificultad radica en la orientación del objeto, donde se debe de agregar al menos 2 ángulos más en comparación con el caso planar.

Tomando intuición del problema bidimensional, resulta que en este caso también se cumple con una estructura similar para la matriz de transformación homogénea, excepto que en este caso la matriz será de 4x4 en lugar de 3x3. Esta matriz forma parte del grupo especial euclideo $SE(3)$.

$${}^I\mathbf{T}_B = \begin{bmatrix} {}^I\mathbf{R}_B & {}^I\mathbf{o}_B \\ \mathbf{0}_{1 \times 3} & 1 \end{bmatrix} \in SE(3) \subset \mathbb{R}^{4 \times 4} \quad (13)$$

El vector ${}^I\mathbf{o}_B$ que contiene las coordenadas no es muy complicado, pues basta con las coordenadas x , y y z con respecto al marco de referencia. No obstante, la obtención de la matriz de rotación ${}^I\mathbf{R}_B$ no es tan fácil. Para abordar este problema se recurrirá al *teorema de rotación de Euler*.

Teorema de rotación de Euler: *Dos marcos de coordenadas ortonormales independientes cualesquiera puede relacionarse mediante una secuencia de rotaciones (no más de tres) sobre ejes de coordenados, donde dos rotaciones sucesivas no pueden ser sobre el mismo eje* [21].

Este teorema representa una poderosa herramienta ya que permite visualizar la rotación final como una secuencia de rotaciones simples a lo largo de los ejes principales, de los cuales ya se cuentan con matrices de rotación definidas (14), (15) y (16).

$$\mathbf{R}_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta \\ 0 & \sin \theta & \cos \theta \end{bmatrix} \quad (14)$$

$$\mathbf{R}_y(\theta) = \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix} \quad (15)$$

$$\mathbf{R}_z(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (16)$$

Representaciones de tres ángulos

En tres dimensiones la rotación no es conmutativa, por lo que el orden en que estas sean aplicadas si generan un resultado diferente. En cuanto al sentido de las rotaciones, se utiliza la convención de la mano derecha.

El *teorema de rotación de Euler* requiere de rotaciones sucesivas a lo largo de tres ejes, sin repetir estos ejes de manera consecutiva. Existen dos secuencias de rotación: Euler y Cardán. En la primera se repiten ejes, pero no de manera consecutiva. En la segunda no se repiten ejes.

- Euler: $XYX, XZX, YXY, YZY, ZXZ, ZYZ$.
- Cardán: $XYZ, XZY, YZX, YXZ, ZXY, ZYX$.

Dependiendo del área de tecnología se tiene diferente convención de secuencia de rotación. Por ejemplo, la secuencia de Cardán ZYX , mejor conocida como roll-pitch-yaw, es ampliamente utilizada en el campo de la navegación [21].

Propiedades de matrices de rotación

De acuerdo con [17], las matrices de rotación cumplen con las siguientes propiedades:

1. La inversa de una matriz de rotación $R \in SO(3)$ también es una matriz de rotación y esta es igual a su transpuesta.

$$\mathbf{R}^{-1} = \mathbf{R}^T$$

2. El producto de dos matrices de rotación también es una matriz de rotación.
3. La multiplicación de matrices de rotación es asociativa pero generalmente no conmutativa. El único caso donde si son conmutativas es en matrices de rotación pertenecientes al grupo especial $SO(2)$.
4. El determinante de una matriz de rotación es 1, lo cual implica que no existe un escalamiento de los vectores que transforma. Es decir, para un vector x , el vector $y = Rx$ tiene la misma norma que x .

Propiedades de matrices de transformación homogéneas

De acuerdo con [17], las matrices de transformación homogéneas cumplen con las siguientes propiedades:

1. La inversa de una matriz de transformación $T \in SE(3)$ también es una matriz de transformación y tiene la forma:

$$T^{-1} = \begin{bmatrix} \mathbf{R} & \mathbf{o} \\ \mathbf{0}_{1 \times 3} & 1 \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{R}^T & -\mathbf{R}^T \mathbf{o} \\ \mathbf{0}_{1 \times 3} & 1 \end{bmatrix}$$

2. El producto de dos matrices de transformación homogénea también es una matriz de transformación homogénea.
3. La multiplicación de matrices de transformación homogénea es asociativa pero no conmutativa.

Singularidades (Gimbal Lock)

Un problema con la representación de tres ángulos son las singularidades. Una singularidad se refiere a una pérdida de un grado de libertad debido a la alineación de un par de ejes, específicamente el eje de en medio con alguno de los ejes exteriores. Este problema se le conoce más popularmente como *gimbal lock* término acuñado gracias a la película de Apollo 13.

La pérdida de este grado de libertad implica matemáticamente que no se puede invertir la transformación. Por tal razón se acude a otras representaciones de orientación, como la *representación de eje-ángulo* o de *cuaterniones unitarios*.

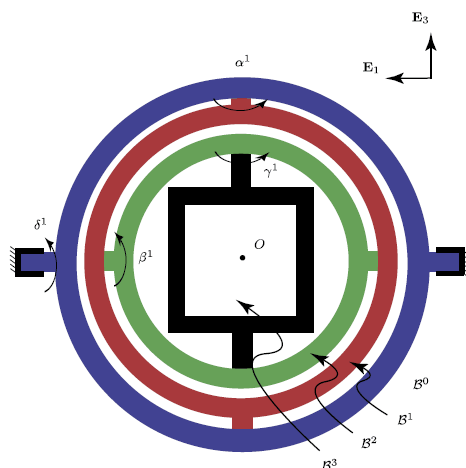


Figura 11: Alineación de ejes causantes de *gimbal lock* [22].

Representación eje-ángulo

Otra interpretación de *teorema de rotación de Euler* indica que cualquier orientación puede representarse como una rotación alrededor de un eje que pasa por un punto fijo del cuerpo rígido. Esto da como resultado el enfoque utilizado por la representación eje-ángulo y tiene las ventajas de no presentar singularidades como lo es el enfoque de tres rotaciones consecutivas alrededor de los ejes coordenados.

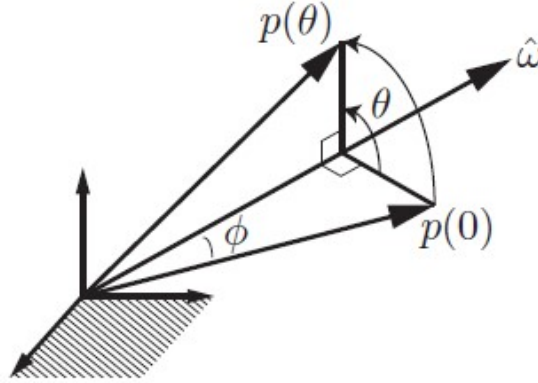


Figura 12: Representación eje-ángulo [17].

Se tiene un vector p que fue rotado θ grados alrededor de un eje de rotación unitario \hat{w} dado por la dirección de su velocidad angular, tal como se muestra en la Figura [12]. La velocidad con que rota ese vector está dada por el producto cruz entre la magnitud de la velocidad angular y el vector p [17].

$$\dot{\mathbf{p}} = \hat{\mathbf{w}} \times \mathbf{p} \quad (17)$$

Con el objetivo de poder continuar utilizando álgebra lineal simple, donde se involucren operaciones matriciales, se introduce el operador antisimétrico, también llamado *skew symmetric*.

Definición: sean $\mathbf{w}, \mathbf{p} \in \mathbb{R}^3$, entonces $\mathbf{w} \times \mathbf{p} = [\mathbf{w}]\mathbf{p}$, donde $[\mathbf{w}]$ es la matriz antisimétrica de \mathbf{w} , la cual resulta de aplicar el operador antisimétrico al vector \mathbf{w} . Esta matriz antisimétrica tiene la forma:

$$[\mathbf{w}] = \begin{bmatrix} 0 & -w_z & w_y \\ w_z & 0 & -w_x \\ -w_y & w_x & 0 \end{bmatrix} \quad (18)$$

Por lo tanto [17] puede expresarse como [19].

$$\dot{\mathbf{p}} = [\hat{\mathbf{w}}]\mathbf{p} \quad (19)$$

(19) tiene la forma de una ecuación diferencial lineal $\dot{x} = ax$, cuya solución es $x = e^{at}x_0$, excepto que ahora la constante a es una matriz *skew symmetric*. De manera análoga, la solución para la ecuación diferencial en (19) está dada por (20), donde las variables t y θ fueron intercambiadas gracias al valor unitario del vector \hat{w} (no se ahondará en la demostración, pues va más allá del objetivo de esta tesis).

$$\mathbf{p}(\theta) = e^{[\hat{w}]^{\theta}} \mathbf{p}(0) \quad (20)$$

La ecuación (20) indica claramente entonces que la matriz $e^{[\hat{w}]^{\theta}}$ rota el vector $\mathbf{p}(0)$ a la posición de $\mathbf{p}(\theta)$, por lo tanto $e^{[\hat{w}]^{\theta}}$ es una matriz de rotación.

La exponencial de la matriz, que a su vez representa la matriz de rotación \mathbf{R} , puede ser calculada mediante a la *fórmula de Rodrigues* (21).

$$\mathbf{R} = e^{[\hat{w}]^{\theta}} = \mathbf{I} + \sin \theta [\hat{w}] + (1 - \cos \theta) [\hat{w}]^2 \in SO(3) \quad (21)$$

6.2.2. Cinemática directa de manipuladores seriales

El problema de la cinemática directa implica encontrar la posición y orientación (*pose*) del efector final en un manipulador serial. Para ello se toman como parámetros los valores en las juntas del robot, lo cual compone a la configuración del mismo. En otras palabras, la cinemática directa es un mapeo de la configuración del robot a su espacio de tarea.

Una manera más sistemática de poder llevar a cabo este proceso involucra colocar un marco de referencia en cada una de las juntas del manipulador, desde la base hasta el efector final. La idea es entonces tener una secuencia de transformaciones homogéneas que permitan llegar hasta el efector final, considerando la configuración del robot, es decir, sus valores en las juntas (las cuales en un manipulador serial siempre son actuadas) este principio es el que se utiliza en la *representación de Denavit-Hartenberg* (21). Otra representación comúnmente utilizada por ciertos libros como (17) es el de producto de exponenciales (PoE), no obstante, esta requiere de un mayor número de parámetros para representar la cinemática directa, por lo que no se presentará en este trabajo de graduación.

La representación de Denavit-Hartenberg

La idea detrás de este planteamiento es derivar la cinemática directa del manipulador mediante a desplazamientos y rotaciones relativas entre los eslabones adyacentes de la cadena cinemática abierta.

Esta representación posee las siguientes restricciones y suposiciones:

1. El manipulador tiene N juntas y $N+1$ eslabones.
2. Cada junta conecta únicamente 2 eslabones.

3. Los eslabones son rígidos.
4. Las únicas juntas aceptadas son juntas revolutas o prismáticas.

Entonces, para un manipulador con N juntas y $N+1$ eslabones, la junta j conecta al eslabón $j-1$ con el eslabón $j+1$, tal como se muestra en la Figura 13. El movimiento relativo del marco de referencia en la junta $j-1$ a la j se describe completamente mediante 4 parámetros, los cuales se conocen como *parámetros de Denavit-Hartenberg* o *parámetros DH* [21]. La transformación que realiza este movimiento entre juntas se llama ${}^{j-1}\mathbf{T}_j$ y está dada por las transformaciones elementales de rotación y traslación de (22).

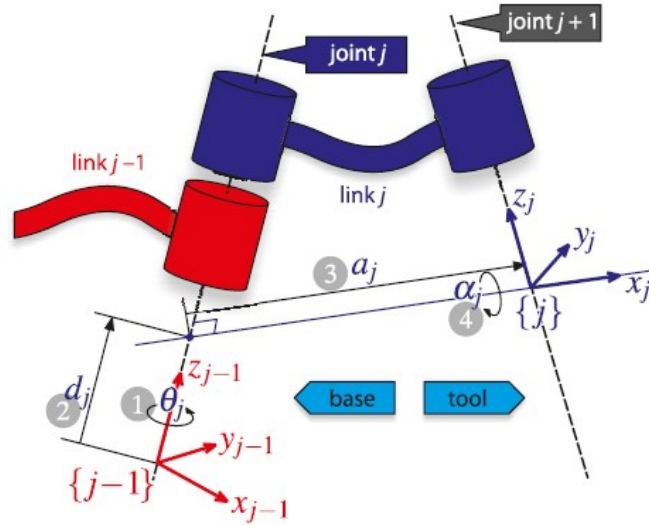


Figura 13: Unión entre eslabones según la representación de Denavit-Hartenberg [21].

$${}^{j-1}\mathbf{T}_j = Rot_z(\theta_j)Transl_z(d_j)Transl_x(a_j)Rot_x(\alpha_j) \quad (22)$$

Los argumentos de las rotaciones y traslaciones elementales de (22) corresponden a los parámetros DH. Estos se describen de manera cualitativa a continuación:

- θ_j : ángulo de rotación de x_{j-1} hasta x_j con respecto al eje z .
- d_j : distancia sobre z hasta que x_{j-1} y x_j sean colineales.
- a_j : distancia sobre el eje x que hace coincidir los orígenes de los marcos de referencia de los eslabones $j-1$ y j .
- α_j : ángulo de rotación de z_{j-1} hasta z_j con respecto al eje x .

Los parámetros a_j y α_j son constantes. θ_j y d_j pueden ser variables, dependiendo del tipo de junta. Para una junta revoluta θ_j es variable y d_j es constante, mientras que para una junta prismática θ_j es constante y d_j es variable. Los valores variables corresponden a los de la configuración del robot, los cuales se representarán mediante un vector \mathbf{q} .

En conclusión, para un manipulador de 3 juntas revolutas y una prismática (RRRP), 4 juntas en total, como el que se muestra en la Figura 14, se requiere de la multiplicatoria de 4 matrices de transformación ${}^{j-1}\mathbf{T}_j$, una para cada movimiento entre juntas. Por simplicidad, se llamará a cada ${}^{j-1}\mathbf{T}_j$ como una matriz de transformación homogénea \mathbf{A}_j que depende del vector de configuración \mathbf{q} del robot.

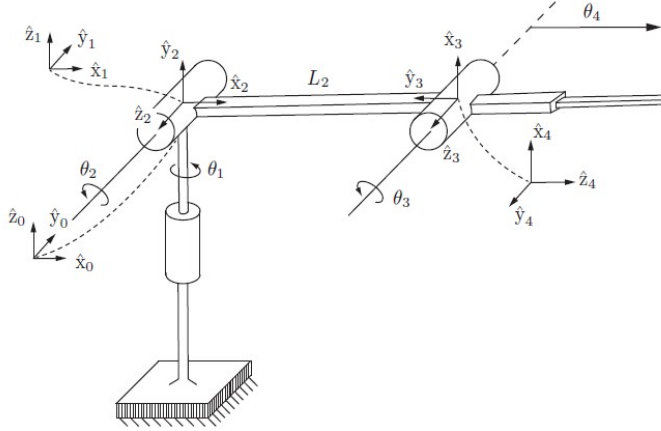


Figura 14: Manipulador RRRP [17].

Por lo tanto, la cinemática directa del efector final $\{E\}$ con respecto a la base $\{B\}$ del manipulador de la Figura 14 está dada por (23).

$$\mathcal{K}(\mathbf{q}) = {}^B\mathbf{T}_E(\mathbf{q}) = \mathbf{A}_1(\mathbf{q}_1)\mathbf{A}_2(\mathbf{q}_2)\mathbf{A}_3(\mathbf{q}_3)\mathbf{A}_4(\mathbf{q}_4) \quad (23)$$

En general, si se tienen N juntas, la cinemática directa utilizando la representación de Denavit-Hartenberg estará dada por (24).

$$\mathcal{K}(\mathbf{q}) = {}^B\mathbf{T}_E(\mathbf{q}) = \prod_{j=1}^N \mathbf{A}_j(\mathbf{q}_j) \quad (24)$$

6.2.3. Cinemática diferencial de manipuladores seriales

El problema de la cinemática diferencial en manipuladores seriales consiste en determinar la velocidad espacial, también conocida en la literatura como *twist*, del efector final. El *twist* no es más que la combinación de la velocidad lineal con la velocidad angular y se denota como un vector $\nu = [v, w]^T$ [17].

La estructura que brinda un manipulador serial debido a sus conexiones, facilita la obtención de la velocidad lineal y angular en comparación con los cuerpos rígidos libres. Recordando que la cinemática directa posee la pose del efector final, resulta que al diferenciar esta con respecto del tiempo se obtiene lo que se conoce como la cinemática diferencial.

El jacobiano

El jacobiano es una matriz que se encarga de mapear la las velocidades de las coordenadas generalizadas a la velocidades del efector final. En otras palabras, el jacobiano representa qué tan sensible es el efector final con respecto a los cambios que ocurren en la configuración del manipulador.

Como se mencionó antes, la cinemática diferencial surge de la derivación de la cinemática directa, por lo tanto:

$$\frac{d}{dt}\mathcal{K}(\mathbf{q}(t)) = \frac{d\mathcal{K}(\mathbf{q}(t))}{d\mathbf{q}} \frac{d\mathbf{q}}{ddt} \quad (25)$$

Escribiéndolo de otra manera (25), se obtiene (26), donde se ve claramente la relación que juega el jacobiano entre la velocidad de la configuración con la velocidad espacial. De esto también se concluye que el jacobiano es la derivada de la cinemática directa con respecto a la configuración. A este jacobiano se le conoce como el jacobiano analítico.

$$\frac{d}{dt}\mathcal{K}(\mathbf{q}(t)) = \mathbf{J}_A(\mathbf{q})\dot{\mathbf{q}} \quad (26)$$

El jacobiano a su vez está compuesto de 2 jacobianos más específicos, el jacobiano de posición $\mathbf{J}_v(\mathbf{q})$ y el jacobiano de orientación $\mathbf{J}_w(\mathbf{q})$, esta relación se muestra en (27).

$$\begin{bmatrix} {}^B\mathbf{v}_E \\ {}^B\mathbf{w}_E \end{bmatrix} = \begin{bmatrix} \mathbf{J}_v(\mathbf{q}) \\ \mathbf{J}_w(\mathbf{q}) \end{bmatrix} \dot{\mathbf{q}} \quad (27)$$

Este enfoque de observar al jacobiano como la derivada de la cinemática directa con respecto a la configuración es conveniente, ya que permite calcular el mismo de manera numérica utilizando métodos como el de diferencias finitas.

Es importante destacar que la orientación presenta ciertas complicaciones en este enfoque, y es que esta puede ser representada de varias formas, por lo general hemos hablado de ella como una matriz de rotación. Al ser esta una matriz, no un campo vectorial, no se puede aplicar la definición matemática del jacobiano para encontrar $\mathbf{J}_w(\mathbf{q})$. No obstante, resulta más fácil encontrar la velocidad angular directamente utilizando la definición que se hizo de la matriz antisimétrica $[\mathbf{w}]$ con la ecuación (28).

$$[{}^B\mathbf{w}_{BE}] = {}^B\dot{\mathbf{R}}_E {}^B\mathbf{R}_E^T \quad (28)$$

Se puede extraer entonces la velocidad angular del efector final con respecto de la base, expresador en coordenadas de la base de la matriz antisimétrica, pues su forma ya es conocida. Entonces de acuerdo a (27), se tiene (29).

$${}^B\mathbf{w}_{BE} = \mathbf{J}_w(\mathbf{q})\dot{\mathbf{q}} \quad (29)$$

6.2.4. Cinemática inversa de manipuladores seriales

El problema de la cinemática inversa consiste en encontrar los parámetros de la configuración del robot, dada la pose del efector final. Es decir, sea ${}^B\mathbf{T}_E$ una transformación homogénea que almacena la pose del efector final, se desea encontrar la solución \mathbf{q} que satisfaga $\mathcal{K}(\mathbf{q}) = {}^B\mathbf{T}_E(\mathbf{q})$.

Este problema es mucho más relevante en la práctica, ya que por lo general se conoce la tarea del robot, entonces lo que hace falta es conocer los valores en la configuración para poder controlar al manipulador. Adicionalmente, esto presenta una ventaja ya que lo común es que cada una de las juntas esté actuada por algún tipo de servomotor, por lo que calcular las coordenadas generalizadas permitirá controlar al robot directamente.

Existen enfoques analíticos y numéricos para poder calcular la cinemática inversa de los manipuladores seriales. El método analítico resulta ser exacto (en aquellos casos donde si existe solución analítica) pero tiene una complejidad de cálculos significativa, sobre todo al momento de crecer el número de grados de libertad del manipulador. En algunos casos es tanta la complejidad que no existe una solución analítica cerrada como tal. El método numérico, por otro lado, resulta ser una opción mucho más viable, por lo cual será el que se presente en este trabajo de graduación.

Cinemática inversa numérica

En la literatura se puede encontrar variedad de métodos numéricos para resolver la problemática de la cinemática inversa. Por ejemplo en [17] se toma un enfoque de resolución de ecuaciones no lineales por Newton-Raphson. Por otro lado, [19] aborda este problema desde una perspectiva de control. Utilizando la última mencionada se puede evitar problemas causados por el hecho de tener esta solución en lazo abierto, es decir, que no se puede corroborar que el efector final efectivamente ha llegado a la posición deseada.

Sea ${}^B\boldsymbol{\xi}_E = [{}^B\mathbf{o}_E(\mathbf{q}(t)), {}^B\boldsymbol{\phi}_E(\mathbf{q}(t))]^T$ el denominado *vector de pose* que denota la pose actual del efector final, entonces se define el error como (30) donde $\boldsymbol{\xi}_d$ es la pose del efector final deseada (la *referencia* desde un punto de vista de control).

$$\mathbf{e} = \boldsymbol{\xi}_d - {}^B\boldsymbol{\xi}_E \quad (30)$$

Si se considera la derivada con respecto del tiempo de (30) y la expresión (26), se puede llegar a (31)

$$\dot{\mathbf{e}} = \dot{\boldsymbol{\xi}}_d - \mathbf{J}_A(\mathbf{q})\dot{\mathbf{q}} \quad (31)$$

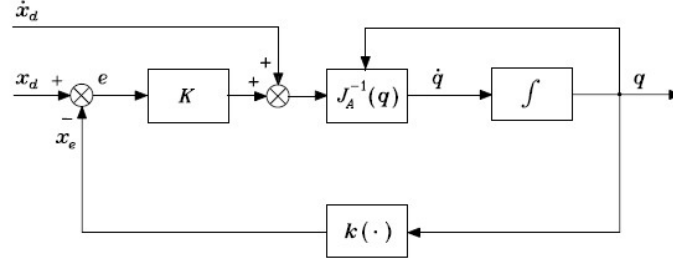


Figura 15: Diagrama de bloques del sistema de control que implementa el algoritmo para la cinemática inversa utilizando la inversa o pseudo-inversa del jacobiano [19].

En este planteamiento se considera a la derivada del vector de configuración como entrada de control y se toma el controlador mostrado en (32), donde $\mathbf{J}^{-1}(\mathbf{q})$ corresponde a la inversa del jacobiano, en caso de ser cuadrada y no singular, o a su *pseudo-inversa* en caso contrario (33) (en tal caso se utiliza el símbolo $\mathbf{J}^\dagger(\mathbf{q})$, en cual hace referencia a la *pseudo-inversa de Moore-Penrose*). \mathbf{K} es una matriz positiva definida que garantiza la convergencia del error a cero y su velocidad de convergencia depende del valor de sus eigenvalores (en cuanto más grandes, la convergencia es más rápida).

$$\dot{\mathbf{q}} = \mathbf{J}_A^{-1}(\mathbf{q})(\dot{\boldsymbol{\xi}}_d + \mathbf{K}\mathbf{e}) \quad (32)$$

$$\dot{\mathbf{q}} = \mathbf{J}_A^\dagger(\mathbf{q})(\dot{\boldsymbol{\xi}}_d + \mathbf{K}\mathbf{e}) \quad (33)$$

Sustituyendo (32) o (33) en (31), se puede obtener el sistema lineal e invariante en el tiempo (LTI) de (34).

$$\dot{\mathbf{e}} = -\mathbf{K}\mathbf{e} \quad (34)$$

El sistema de control resultante de este planteamiento se muestra en la Figura 15. Donde $\mathbf{k}(\cdot)$ es la cinemática directa y \mathbf{x} fue cambiada por $\boldsymbol{\xi}$ según la notación utilizada en [19].

Tomar la referencia $\boldsymbol{\xi}_d$ como constante implica que el término en *feedforward* $\dot{\boldsymbol{\xi}}_d$ es cero, lo cual asegura un error en estado estable de cero.

Finalmente, es de interés conocer la configuración, no su derivada con respecto del tiempo, por tanto (33) puede ser resuelta por algún método numérico para ecuaciones diferenciales tipo *forward Euler* o *Runge Kutta*. La ecuación (35) muestra la resolución por el método de *forward Euler*.

$$\mathbf{q}_{k+1} = \mathbf{q}_k + \mathbf{J}_A^\dagger(\mathbf{q}_k)(\boldsymbol{\xi}_d[k] + \mathbf{K}\mathbf{e}_k) \quad (35)$$

De manera similar a como se hizo en (27), la expresión (35) se puede separar en una parte correspondiente a la posición y otra a la orientación, separando sus jacobianos. Esto genera las ecuaciones (36) y (37).

$$\mathbf{q}_{k+1} = \mathbf{q}_k + \mathbf{J}_v^\dagger(\mathbf{q}_k)(\mathbf{o}_d)[k] + \mathbf{K}_p \mathbf{e}_p[k] \quad (36)$$

$$\mathbf{q}_{k+1} = \mathbf{q}_k + \mathbf{J}_w^\dagger(\mathbf{q}_k)(\phi_d)[k] + \mathbf{K}_o \mathbf{e}_o[k] \quad (37)$$

La pseudo-inversa de Moore-Penrose

La pseudo-inversa corresponde a la matriz \mathbf{J}^\dagger , tal que da solución a la ecuación $\mathbf{J}\mathbf{a} = \mathbf{b}$ por un enfoque de *ajuste óptimo* de mínimos cuadrados. Esto quiere decir que la pseudo-inversa busca minimizar $\|\mathbf{J}\mathbf{a} - \mathbf{b}\|_2^2$ [23].

Dependiendo de, si la matriz del jacobiano posee más filas que columnas o viceversa, se utiliza una definición diferente de la pseudo-inversa. Para el primer caso (*jacobiano delgado*) se usa [38] y para el segundo caso (*jacobiano gordo*) se usa [39] [24].

$$\mathbf{J}^\dagger = (\mathbf{J}^T \mathbf{J})^{-1} \mathbf{J}^T \quad (38)$$

$$\mathbf{J}^\dagger = \mathbf{J}^T (\mathbf{J} \mathbf{J}^T)^{-1} \quad (39)$$

6.3. Brazo robótico R17

El R17 es un manipulador serial de 5 o 6 grados de libertad de la compañía *ST Robotics* (el sexto grado de libertad se le otorga si se le incorpora la junta prismática sobre la que va montado). De acuerdo con sus creadores "el R17 es un brazo robótico de bajo costo, preciso, confiable y fácil de controlar" [25]. Este robot es ideal para aplicaciones que requieren un alto alcance, alta carga y alta velocidad.

Este modelo utiliza motores stepper con un paso de 1.8 grados, lo cual resulta en una resolución de 0.1 mm en el efector final. Su alcance es de 750 mm, desde la junta del *shoulder* hasta la junta del efector. Otra característica importante es que su sistema de monitoreo es mediante encoders, los cuales se encargan de parar el movimiento en caso de una colisión o de un error con respecto al movimiento que se le fue indicado al brazo de realizar.



Figura 16: Brazo robótico R17 [25].

El sistema del R17 está compuesto de tres partes, el robot, el controlador y la computadora. El robot tiene un formato de articulación vertical, lo cual lo hace lo más parecido a un brazo humano, donde tiene 5 juntas revolutas y estas están nombradas según partes del cuerpo como se muestra en la Figura [17]. El controlador almacena el programa desarrollado en la computadora para que el brazo pueda funcionar sin necesidad de estar conectado a ella; este controla todos los movimientos del brazo y se encuentra leyendo continuamente los sensores para poder decisiones en cuanto a su actividad. La computadora se encarga de programar al controlador y lo hace a través del lenguaje *ROBOFORTH II* [26].

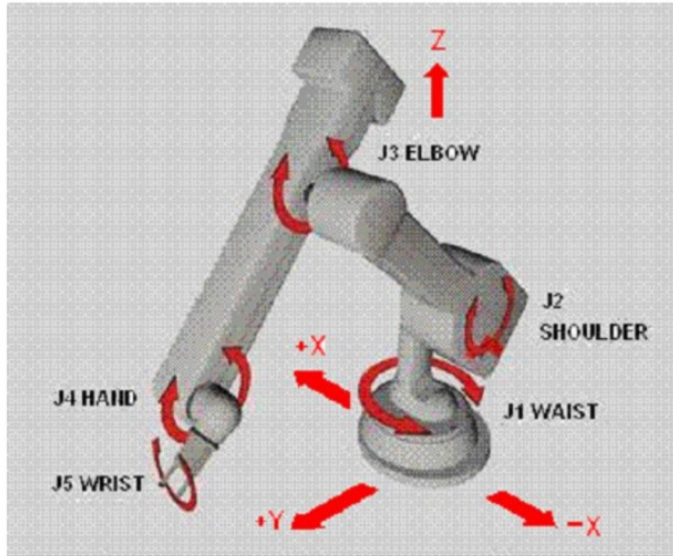


Figura 17: Articulaciones del R17 [5].

6.3.1. ROBOFORTH II

ROBOFORTH II es un lenguaje de programación de 4ta generación, esto quiere decir que está orientado a listas. Este es un lenguaje no gráfico de alto nivel de abstracción, utiliza palabras en inglés para definir comandos (*WORDS*). *ROBOFORTH* emplea una comunicación de dos vías, donde el usuario le pide al robot que haga y este le contesta al usuario.

ROBOFORTH está organizado como una lista enlazada de palabras, conocida como un *diccionario*. Junto con cada palabra en el diccionario está su definición, que generalmente se expresa en términos de palabras más simples del mismo lenguaje, como un diccionario normal. Hay más de 500 palabras en *ROBOFORTH* disponibles para el usuario, pero normalmente solo necesita un pequeño vocabulario para cualquier aplicación en particular. La programación consiste en crear nuevas palabras en el diccionario cuyas definiciones están en términos de palabras ya definidas. Las palabras nuevas pueden incluir datos, es decir, información posicional sobre el robot. Los valores (o argumentos) se pasan entre palabras en una pila (*stack*) [27].

Las posiciones del robot se manejan por coordenadas. Estas coordenadas son las del efector final y están referidas a un origen de un marco de referencia, denominado *HOME*. Estas se expresan en milímetros de manera cartesiana *XYZ*.

ROBOFORTH tiene la capacidad de almacenar rutinas y reprogramar acciones cuando ocurren errores. Estas son algunas de las características que hacen a este lenguaje de programación tan conveniente y fácil de utilizar. Cabe mencionar que todos los comandos tienen que ser escritos en letras mayúsculas (*UPERCASE*) y estos terminan con un *enter*. En la Figura [18] se muestran algunos comandos que utiliza el R17 mediante a *ROBOFORTH II*.

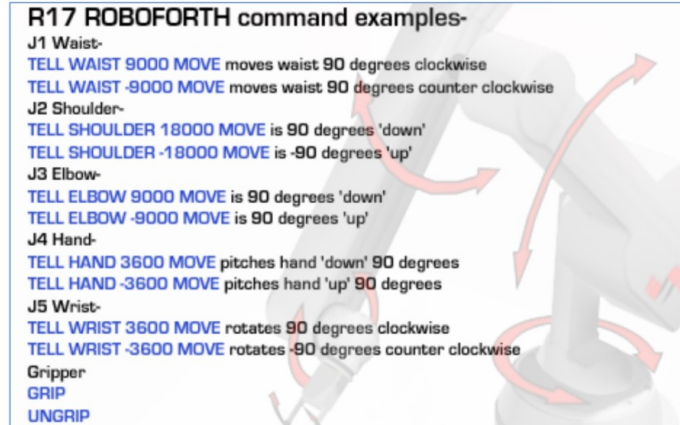


Figura 18: Comandos en ROBOFORTH para R17 [5].

6.4. Comunicación inalámbrica

De manera general, se puede definir una comunicación inalámbrica como el intercambio de información entre dos o más dispositivos (emisores o receptores) utilizando el espectro electromagnético [28]. Dependiendo de la literatura consultada, la comunicación inalámbrica se puede clasificar en varios grupos, en este trabajo se presenta la siguiente clasificación:

- Redes de área personal inalámbrica (WPAN: wireless personal area networks): presentan una limitación de alcance. Los dispositivos que buscan comunicarse debe de estar relativamente cerca. Generalmente, se acepta como límite el espacio de una habitación. Las tecnologías más utilizadas de WPAN son las siguientes: Bluetooth, DECT8, IrDa9, NFC10 y Zigbee.
- Redes de área local inalámbrica (WLAN: wireless local area networks): es una red de cobertura geográfica limitada, velocidad de transmisión relativamente alta, bajo nivel de errores y administrada de manera privada. Dentro de las ventajas de una WLAN se encuentra su fácil instalación y la comodidad del usuario. Las WLAN son una alternativa a las LAN con cables. Los usuarios de una WLAN pueden acceder a los recursos que les ofrece la LAN sin tener que depender de infraestructuras de red (cableado, conectores, etc.)
- Redes de área extendida inalámbrica (WWAN: wireless wide area networks): permiten la conexión de redes y usuarios de zonas geográficamente distantes. Esta se diferencia de la WLAN en que utilizan tecnología de red celular.

6.4.1. WiFi

En un contexto no tan técnico, WiFi (abreviatura de Wireles Fidelity) es un tipo de interconexión inalámbrica WLAN que permite a dispositivos interactuar con el internet. Estrictamente hablando, WiFi corresponde al nombre de marca utilizado para marcar productos que pertenecen a una categoría de dispositivos WLAN. Los dispositivos o hardware marcados con la marca WiFi se basan en los estándares establecidos por IEEE 802.11. En la mayoría de los casos, WiFi es considerado por la mayoría como sinónimo del propio estándar.

El IEEE 802.11 es una familia de estándares para redes WLAN. Este estándar garantiza la interoperabilidad entre diferentes fabricantes. En un principio, la expresión WiFi era utilizada únicamente para los aparatos con tecnología 802.11b, que funciona en una banda de frecuencias de 2,4 GHz y permite la transmisión de datos a una velocidad de hasta 11 Mbps. Con el fin de evitar confusiones en la compatibilidad de los aparatos y la interoperabilidad de las redes, el término WiFi se extendió a todos los aparatos provistos con tecnología de la familia IEEE 802.11: 802.11a, 802.11b y 802.11g.

La arquitectura del 802.11 se basa en una estructura celular. A cada célula se le denomina BSS (Basic Service Set) y está formado por una estación y a la vez es gobernada por un AP (Access Point). El AP es el elemento esencial de la red inalámbrica puesto que se encarga de transmitir y recibir la señal a las estaciones, también llamadas clientes inalámbricos. Las estaciones suelen ser algún tipo de computadoras, provistas de interfaces de red inalámbricos, tanto portátiles como no. Estas interfaces suelen ser tarjetas [29].



Figura 19: Arquitectura lógica funcional de IEEE 802.11 [29].

6.4.2. Protocolo MQTT

MQTT es un protocolo de comunicación de mensajería publish/suscribe (pub/sub). Este protocolo se caracteriza por ser liviano, simple, seguro y eficiente, además consume muy poco ancho de banda. Por estas razones este protocolo ha crecido mucho en popularidad en el *Internet de las Cosas* (IoT), siendo ideal para la implementación de red de sensores y microcontroladores.

Su funcionamiento se basa en el modelo pub/sub, en donde se separan a los dispositivos que envían información (publicadores) de aquellos que la reciben (suscriptores). Estos dos nunca se contactan directamente, sino que lo hacen a través de lo que se conoce como *broker*. El broker se encarga de filtrar los mensajes provenientes de algún publicador y distribuirlos correctamente a los suscriptores, esto hace posible controlar que suscriptores reciben los mensajes.

El filtrado por parte del broker puede ocurrir de acuerdo a varias opciones. En el protocolo MQTT este ocurre según el conocido *filtrado por tema*. En este tipo de filtrado el cliente que recibe se suscribe a un tema (*topic*) de interés. El broker se encarga entonces que el cliente reciba todas las publicaciones que se hicieron en ese tema. Por lo general los *topics* son strings con estructura jerárquica. Todo esto forma parte de la arquitectura pub/sub del protocolo, la cual se muestra gráficamente en la Figura 20.

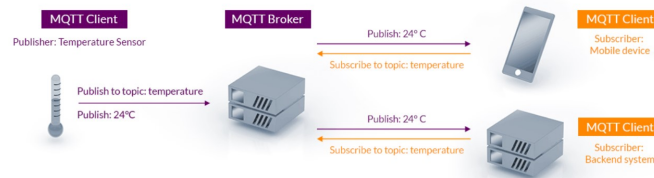


Figura 20: Arquitectura publish/subscribe de MQTT 30.

Para una mayor cantidad de clientes, la arquitectura de MQTT sigue una topología de estrella, con un nodo central que hace de servidor (el broker) con una capacidad de hasta 10,000 clientes. Los clientes mandan periódicamente un paquete (PINGREQ) y esperan la respuesta del broker (PINGRESP). La comunicación puede ser cifrada entre otras muchas opciones 31.

El desarrollo de este trabajo de graduación se puede dividir en tres bloques principales. La idea general de cada uno de estos bloques está descrita en los objetivos específicos planteados anteriormente. Estos bloques se enlistan a continuación:

1. Montaje, configuración, pruebas y obtención de datos en el sistema de captura de movimiento Optitrack.
2. Modelado cinemático y desarrollo de software para controlar al manipulador serial R17.
3. Integración del manipulador R17 con el ecosistema Robotat.

Para completar cada uno de estos bloques, se llevó a cabo una serie de pasos, estos se describen en lo que resta de esta sección.

Montaje, configuración, pruebas y obtención de datos en el sistema de captura de movimiento OptiTrack

- Instalación de software Motive: se descargó en la página oficial de Optitrack, luego se activó la licencia. Para la activación de la licencia se conectó la llave física, luego se inició el programa y se utilizó el *License Tool*; aquí se introdujo el número de serie de la licencia y el *código hash*.
- Posicionamiento de cámaras en el área de trabajo: estas se colocaron alrededor de un área de trabajo despejada, con poca iluminación y sobre un piso no muy reflectante, tal y como lo indica Optitrack. Estas se montaron en los trípodes a una altura entre 2.5 y 3 metros.

- Conexión de componentes: primero se conectó el switch a la computadora donde se encuentra el software Motive; la conexión se realiza mediante a cable ethernet. Luego se conectaron todas las cámaras al switch. Cada cámara es energizada por la misma conexión ethernet.
- Enfoque de las cámaras: se posicionó un marcador dentro del área de captura de movimiento, se colocaron las cámaras en *raw grayscale mode*, se hizo zoom en uno de los marcadores reflectivos y se verificó la claridad de la imagen. Finalmente se buscó el punto donde la imagen del marcador tiene la mejor resolución.
- Calibración: primero se realizó el proceso conocido como *masking*, el cual significa marcar aquellos objetos que detectan las cámaras pero que no forman parte de los marcadores. Luego viene el proceso de *wanding*, el cual es mover la vara de calibración (*wand*) frente a las cámaras, el software se encargará entonces de capturar posiciones y orientaciones múltiples veces. El último paso de la calibración fue colocar la escuadra de calibración, donde la vara larga es el eje z, la corta el eje x y el eje y va hacia arriba (donde se encuentra un marcador). Esta escuadra representa el origen del marco de referencia inercial y también debe de colocarse completamente horizontal con ayuda del nivel.
- Colocación de marcadores: las pruebas se pueden hacer en sujetos o en objetos. Para el sujeto de prueba se debe de colocar los marcadores en el traje, en caso de usar un objeto se debe de colocar en las bases para cuerpos rígidos. Se realizó una prueba para cada una de estas opciones.
- Creación de una sesión de captura: se creó un folder en la computadora para la sesión. Todas las capturas fueron guardadas en este como un archivo *Take* (TAK).
- Grabar una sesión: se presionó el botón para grabar y toda la información de la captura de movimiento se guardó en un archivo *Take*.
- Capturar la pose del efector final del R17: se repitieron los pasos anteriores ahora con el manipulador serial dentro del sistema de captura de movimiento, de tal manera que se pudo obtener información acerca de la pose de su efector final.

Modelado cinemático y desarrollo de software para controlar al manipulador serial R17

- Mediciones de la geometría del R17: la idea original era medir los eslabones del manipulador serial, no obstante se contaba con unos planos obtenidos desde la página oficial del proveedor del R17, ST Robotics, y se tomaron estos valores para realizar el modelo. También se identificó el tipo de juntas que posee.
- Generación de la matriz de parámetros de Denavit-Hartenberg: se traducieron las mediciones hechas en el paso anterior a los parámetros θ_j , d_j , a_j y α_j de DH. Estos se colocaron en una matriz de DH.
- Simulación de la geometría del R17: en MATLAB se generó una simulación que permitió validar la geometría del manipulador mediante parámetros DH. Esta validación hasta este punto fue meramente cualitativa, pues solo se observó que la forma que tomaba el robot al adoptar cierta configuración concordara con la del robot en físico.

- Creación de un programa para calcular la cinemática directa: se generó una función en MATLAB que toma como parámetro la configuración del R17 y devuelve su cinemática directa en forma de matriz de transformación homogénea.
- Creación de un programa para realizar un movimiento en el R17 con cinemática directa: se generó una función en MATLAB que tome como parámetro la configuración del R17 y que devuelva los comandos adecuados que se deben de enviar al robot para moverlo.
- Creación de un programa para calcular la cinemática inversa: se generó una función en MATLAB que toma como parámetro la pose deseada del efector final, en forma de matriz de transformación homogénea, y devuelve el vector de configuración del R17.
- Creación de un programa para realizar un movimiento en el R17 con cinemática inversa: se generó una función en MATLAB que tome como parámetro la pose deseada del efector final y que devuelva los comandos adecuados que se deben de enviar al robot para moverlo.
- Creación de programa para planificación de trayectorias: se generó una función en MATLAB que tome como parámetros una tabla con las posiciones deseadas (en sus coordenadas x , y y z) por las que se quiere pasar en la trayectoria y el modo de corrido de la misma. Estos modos pueden ser segmentado, continuo y suave.
- Creación de librería para comunicarse con el R17: se programó una librería en MATLAB que permite comunicarse con el manipulador mediante al lenguaje ROBOFOTH II. Los comandos ya en ROBOFOTH II son enviados mediante a comunicación serial al robot, el cual cuenta con un conector RS-232 a USB.
- Creación de una interfaz para visualizar el control del R17: en MATLAB se creó una interfaz gráfica donde se integraron todos los programas realizados en este bloque, de tal manera que se puede controlar al robot más fácilmente y se puede validar su modelo de una manera cualitativa.

Integración del manipulador R17 con el ecosistema Robotat

- Pruebas iniciales con el ESP32: se hizo pruebas para familiarizarse con los comandos y librerías disponibles para comunicación WiFi por MQTT y comunicación serial. Estas permitieron validar su funcionamiento y comprensión.
- Programación de un librería para integrar el R17 al ecosistema Robotat: se creó una librería dentro del ESP32 que pueda recibir información desde el ecosistema para el manipulador, de tal manera que este pueda efectuar las trayectorias deseadas.
- Diseño y fabricación de sistema embebido para comunicar de forma inalámbrica el ESP32 con el R17: se probó un circuito para convertir la señal serial que otorga el ESP32 a una salida por un conector tipo DB9, el cual comunica con el robot por la interfaz RS-232. Una vez se validó el funcionamiento, se diseñó y fabricó un PCB para integrar toda esta funcionalidad en un sistema embebido.
- Pruebas en conjunto R17 - OptiTrack: se colocó un set de marcadores para identificar al R17 dentro del *mocap* y se realizaron pruebas de cinemática inversa y ejecución de trayectorias.

Montaje, configuración, pruebas y obtención de datos en el sistema de captura de movimiento OptiTrack

8.1. Montaje y configuración

El sistema de captura de movimiento empleado en este trabajo de graduación corresponde al OptiTrack, uno de los *mocaps* más reconocidos del mundo. Este sistema incluye el software Motive y cámaras de rastreo de alta velocidad. En el caso particular del ecosistema Robotat, se utilizan 6 cámaras *Prime^x 41*.

Parte importante de este trabajo era el levantamiento del sistema OptiTrack, lo cual se trabajó en conjunto con el estudiante Camilo Perafán. Este proceso involucró desde el posicionamiento de las cámaras en el entorno de observación y la instalación del software, hasta el enfoque y calibración de cada una de las cámaras. Este proceso se describe de forma detallada en [32]. El montaje final del OptiTrack se muestra en la Figura [21].



Figura 21: Montaje del sistema OptiTrack.

8.2. Pruebas y obtención de datos

Parte de los objetivos es obtener información sobre el R17 a través del sistema de captura de movimiento. En específico, es de interés la pose del efector final, por lo que se introdujo el R17 dentro del espacio de observación del Optitrack y se le colocó un marcador de cuerpo rígido en el efector final.



Figura 22: Cuerpo rígido en efector final del R17.

Una vez colocado el marcador y encendido el OptiTrack, se creó el cuerpo rígido dentro del software Motive. Para ello se debe de ubicar a los tres marcadores del cuerpo rígido, dirigirse dentro de Motive al apartado de *Rigid Bodies*, seleccionar los tres marcadores haciendo *click + ctrl* (Figura 23) y luego asignar un nombre y presionar sobre la opción de *Create* (Figura 24).

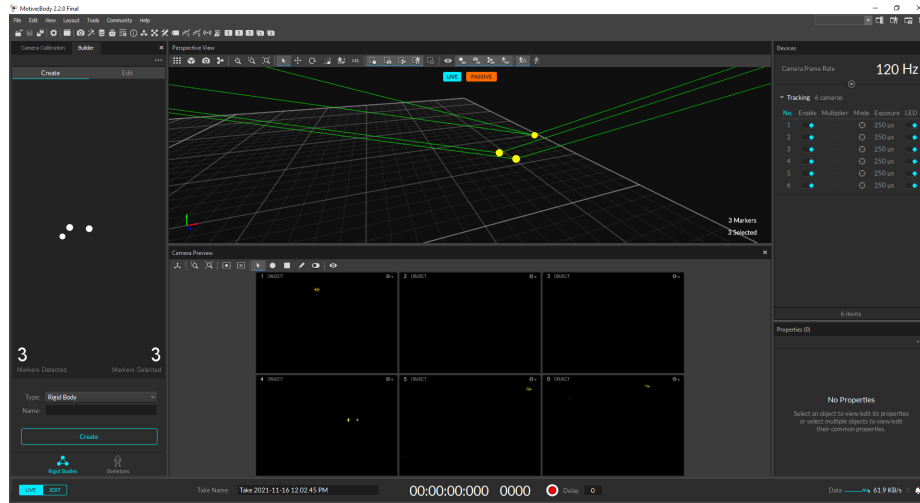


Figura 23: Selección de marcadores para creación de cuerpo rígido en el efector final del R17.

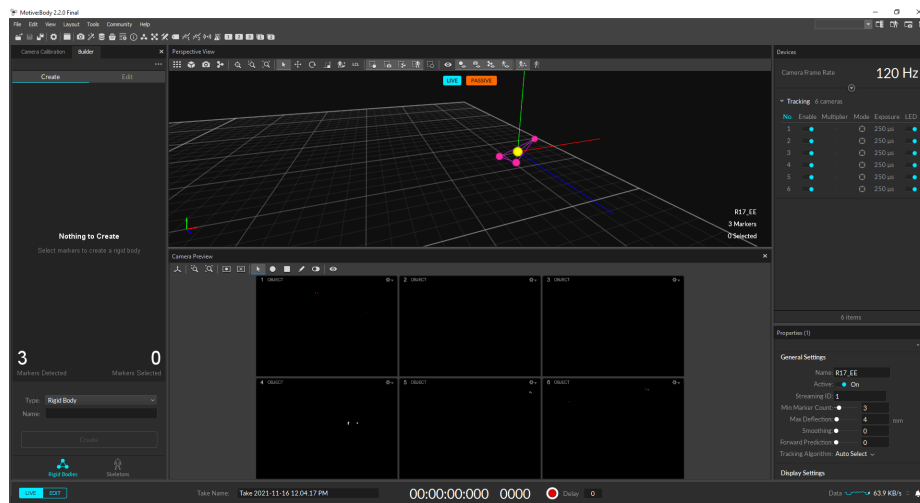


Figura 24: Cuerpo rígido en el efector final del R17.

Se desea verificar de manera intuitiva que el OptiTrack está tomando correctamente la información de pose del efector final. Para ello se colocó al R17 en cuatro posiciones diferentes y se comparó cualitativamente la posición y orientación del marco de referencia mostrado en Motive con el visto en vida real en el efector final del manipulador serial. También se observó la información de las coordenadas cartesianas de posición y ángulos de rotación en los ejes principales de Roll(z), Pitch(x) y Yaw(y) del cuerpo rígido.

8.2.1. Posición 1

En esta posición se colocó al R17 completamente vertical. Fue en esta posición en la que se hizo la creación del cuerpo rígido del efector final, por lo que el marco de referencia del cuerpo se encuentra alineado con el del origen, es decir, presenta valores cercanos a 0° en los ejes de rotación de Roll(z), Pitch(x) y Yaw(y) (se muestra en la pestaña de *info* de la Figura 26).

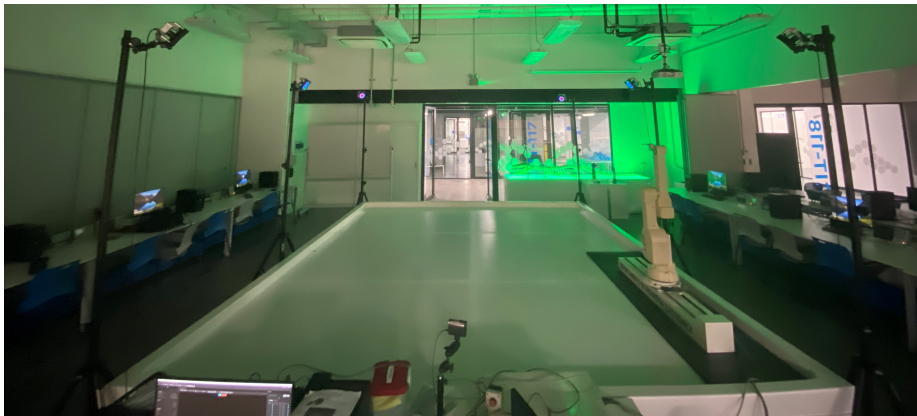


Figura 25: Posición 1 - R17 real.

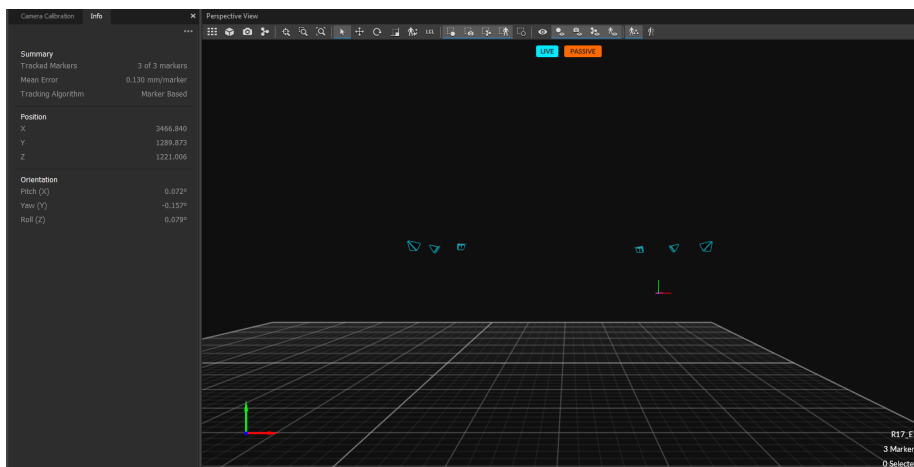


Figura 26: Posición 1 - Motive.

8.2.2. Posición 2

En esta posición se colocó al R17 completamente horizontal, tal como se muestra en la Figura 27. Comparándolo con lo obtenido en Motive, se ve que esto tiene sentido, pues existe una traslación significativa en y y z negativo, al igual que una rotación de -89.769° en Pitch.



Figura 27: Posición 2 - R17 real.

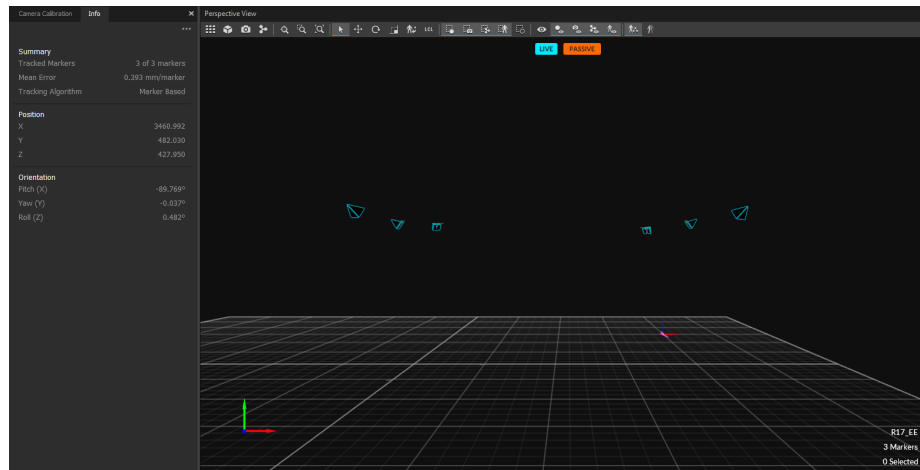


Figura 28: Posición 2 - Motive.

8.2.3. Posición 3

Esta posición se compara únicamente de manera cualitativa. Se puede apreciar en la Figura 29 que el efector final debe de poseer una posición más cercana al suelo de la plataforma y al origen, en comparación con la posición 1. Además, el efector final apunta en dirección diagonal hacia el suelo. Esto se hace sentido con la imagen y valores de coordenadas mostrados en la Figura 30. También se puede ver que ahora el eje y , el cual es normal al cuerpo rígido colocado sobre el efector final, también apunta en una dirección diagonal hacia el suelo.



Figura 29: Posición 3 - R17 real.

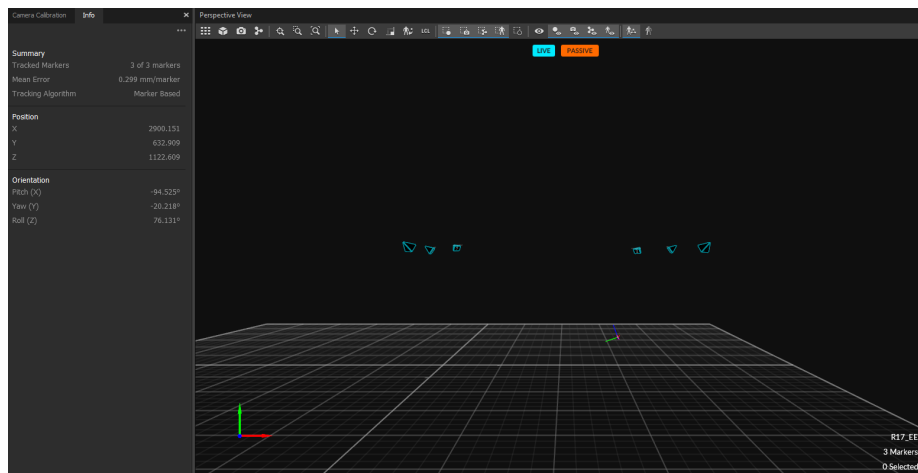


Figura 30: Posición 3 - Motive.

8.2.4. Posición 4

Esta posición se compara únicamente de manera cualitativa, al igual que la posición 3. Se puede apreciar en la Figura 31 que el efector final apunta en dirección diagonal contraria hacia el suelo de la plataforma. Esto se hace sentido con la imagen mostrada en la Figura 32 en donde el eje y , el cual es normal al cuerpo rígido colocado sobre el efector final, también apunta en esta dirección.



Figura 31: Posición 4 - R17 real.

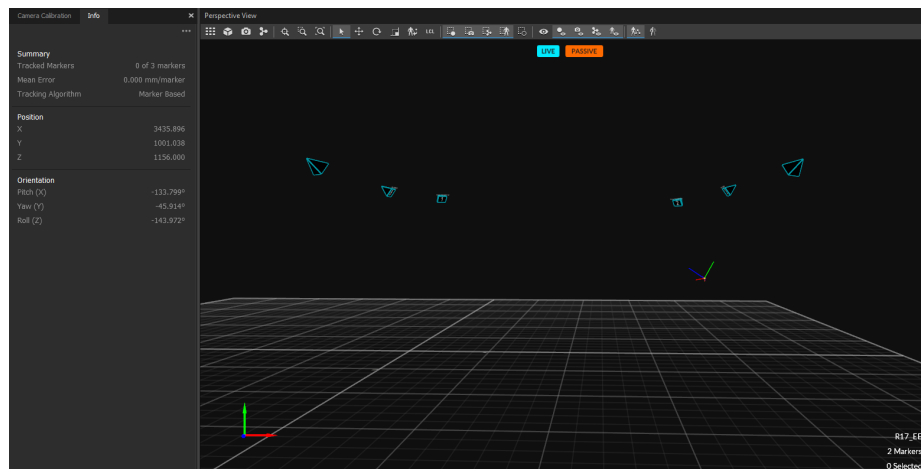


Figura 32: Posición 4 - Motive.

Con estas cuatro posiciones de prueba se pudo verificar cualitativamente el correcto funcionamiento del *mocap* OptiTrack con el R17. Además se obtuvo información importante de la pose del efector final, que fueron la posición en coordenadas cartesianas y la orientación en términos de los ángulos de rotación Roll(z), Pitch(x) y Yaw(y); esto es de suma importancia para la integración de las funcionalidades del R17 con el ecosistema Robotat (Capítulo 10).

Modelado cinemático y desarrollo de software para controlar al manipulador serial R17

9.1. Modelo cinemático y representación DH

La entrega final de este trabajo de graduación contempla que el robot R17 pueda ser operado en tres modos diferentes: cinemática directa, cinemática inversa y ejecución de trayectorias. Para cumplir con este objetivo resulta fundamental poder plantear el modelo cinemático del manipulador serial. Este permite a su vez, simular y animar al R17 dentro de su espacio de tarea, de tal manera que sea sencillo visualizar las posiciones y movimientos que tomará el robot una vez el control ya sea implementado. Por otra parte, con este modelo y el software para manejar al robot, se puede validar que efectivamente las instrucciones que se están enviando al R17 han sido procesadas correctamente y cumplen con su funcionamiento.

Para representar al manipulador siguiendo la convención DH (Denavit-Hartenberg) se debe de conocer únicamente los tipos de juntas y las longitudes efectivas de los eslabones. Se sabe de antemano que todas las juntas son revolutas a excepción del riel en la base, el cual es una junta prismática. Las longitudes fueron obtenidas de la pagina oficial del distribuidor del manipulador, ST Robotics, el cual mostraba un set de planos del robot. Esta información se muestra en la Figura [33](#).

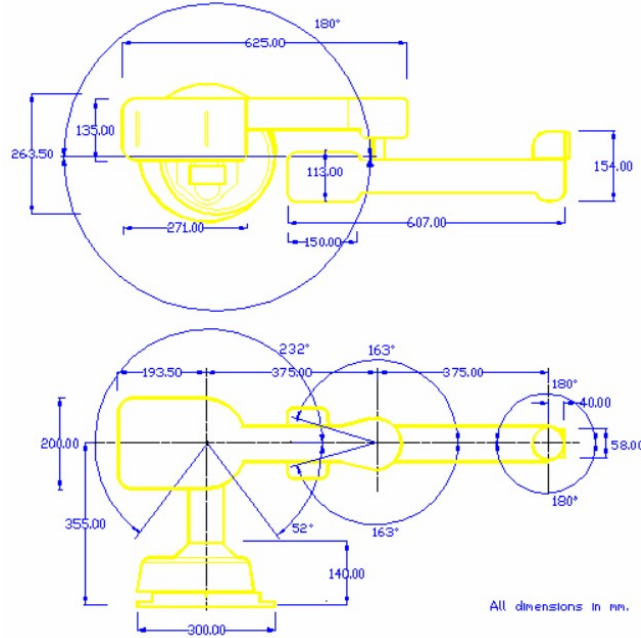


Figura 33: Planos del manipulador R17 [25].

Con las dimensiones de la Figura [33] se obtuvo la matriz de parámetros DH que se muestra en el Cuadro [3].

Matriz de parámetros DH			
θ_j (rad)	d_j (m)	a_j (m)	α_j (rad)
0	q_1	0	$-\pi/2$
$q_2 + \pi/2$	-0.355	0	$\pi/2$
$q_3 + \pi/2$	0	0.375	0
q_4	0	0.375	0
$q_5 - \pi/2$	0	0	$\pi/2$
$q_6 + \pi/2$	0	0	0

Cuadro 3: Matriz de parámetros DH de manipulador serial R17.

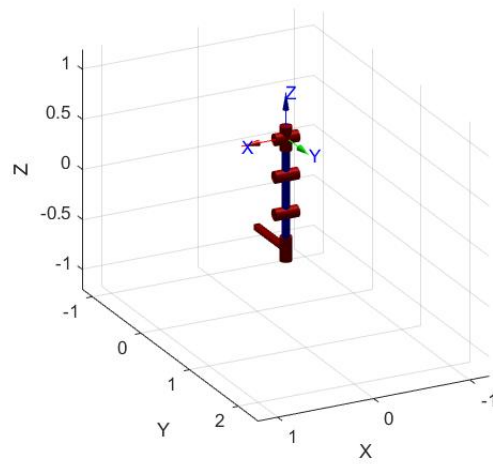
Para corroborar que la matriz del Cuadro [3] es correcta, se generó una simulación gráfica de la geometría del manipulador serial. Esto se hizo con el toolbox de robótica de Peter Corke [33], el cual es una herramienta que permite crear una representación de la forma física de un manipulador utilizando como entradas la información dentro de la matriz de parámetros DH. Lo que se presenta a continuación no es más que una validación cualitativa del modelo cinemático planteado del manipulador, la cual tiene el objetivo de generar una intuición acerca del correcto planteamiento del modelo.

De las Figuras [34] a la [36] se presenta una comparación visual entre la simulación utilizando el modelo cinemático planteado y la geometría real del robot. Se escogió la configuración inicial del robot, es decir, cuando todos sus parámetros valen cero y otras dos configuraciones aleatorias del robot. En todos los casos se puede observar que sí existe una semejanza

entre la posición real que toma el R17 y la mostrada por MATLAB al ser simulada según el modelo planteado.

Es importante destacar que el modelo planteado con la convención DH toma el origen del robot en la base del mismo, es decir, donde comienza la junta prismática (el riel). No obstante, el controlador del R17 define su origen en la junta del hombro (shoulder), por lo que se debió utilizar una transformación de base en el modelo para poder obtener coordenadas con respecto al mismo punto que el robot. Esta se muestra en (40).

$$T_{base} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0.355 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (40)$$

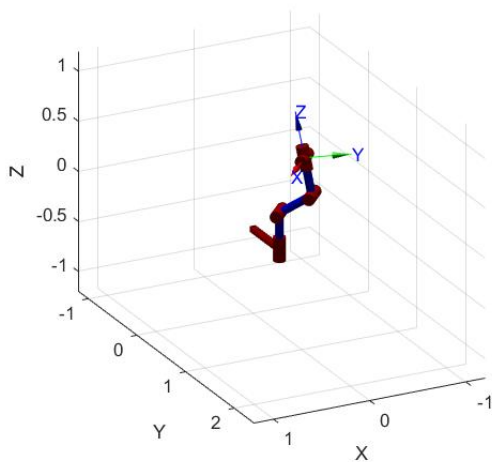


(a) Simulación de la geometría del R17 en MATLAB.

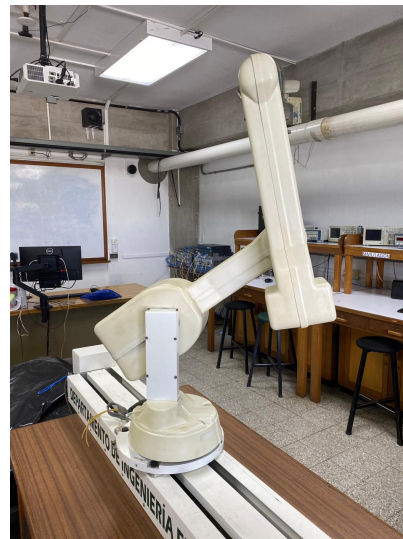


(b) R17 real.

Figura 34: Comparación entre la simulación de la geometría del R17 en MATLAB y el sistema real para una configuración $q = [0, 0, 0, 0, 0, 0]^T$.

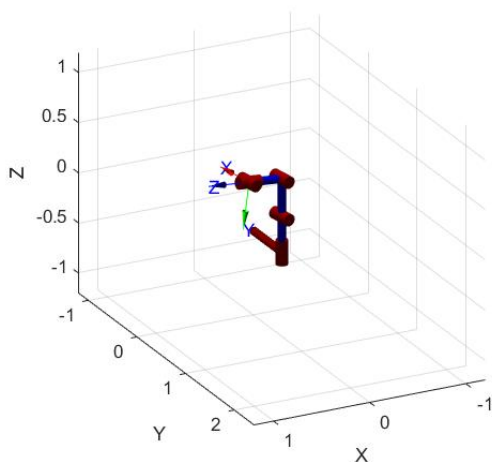


(a) Simulación de la geometría del R17 en MATLAB.



(b) R17 real.

Figura 35: Comparación entre la simulación de la geometría del R17 en MATLAB y el sistema real para una configuración $q = [0, -68, 59, -70, 0, 0]^T$.



(a) Simulación de la geometría del R17 en MATLAB.



(b) R17 real.

Figura 36: Comparación entre la simulación de la geometría del R17 en MATLAB y el sistema real para una configuración $q = [0, -85, 3, -80, 0, 0]^T$.

9.2. Implementación de la cinemática directa

El R17 cuenta con un modo de operación denominado *JOINT mode*, el cual permite efectuar cinemática directa en el robot, es decir, controlar directamente sus juntas. Parte de la sintaxis del robot para poder trabajar en este modo involucra mandarle el número de pasos (*steps*) que debe de hacer cada motor en cada junta. Los motores de las juntas del robot son *steppers*, y cada uno de estos tiene una tasa establecida la cual especifica cuantos pasos se mueve el robot por cada 90 grados que gira el eje, en el caso de las juntas revolutas, y cuantos pasos se mueve por cada 0.5 metros de desplazamiento lineal, en el caso de la junta prismática. Esta información es importante porque permite al usuario pensar en la configuración en términos de grados o metros y no de los pasos del motor. Otro aspecto a resaltar es que el robot no tiene una programación defensiva en este modo que evite tratar de realizar configuraciones que se encuentran fuera de sus límites. Enviar una configuración que se salga de estos límites puede implicar una desconfiguración por parte del R17. Por tal razón se implementó una programación defensiva en el toolbox.

La creación de los comandos en ROBOFORTH II que se encargan de controlar el R17, por medio de cinemática directa, se hacen dentro de una función de MATLAB llamada R17fKine. Esta función toma como argumento un vector \mathbf{q} , el cual contiene la configuración del robot, es decir, los valores de posición en cada una de sus juntas, y da como resultado la escritura en el puerto serial de estos comandos. El diagrama de flujo de esta función se muestra en la Figura 37.

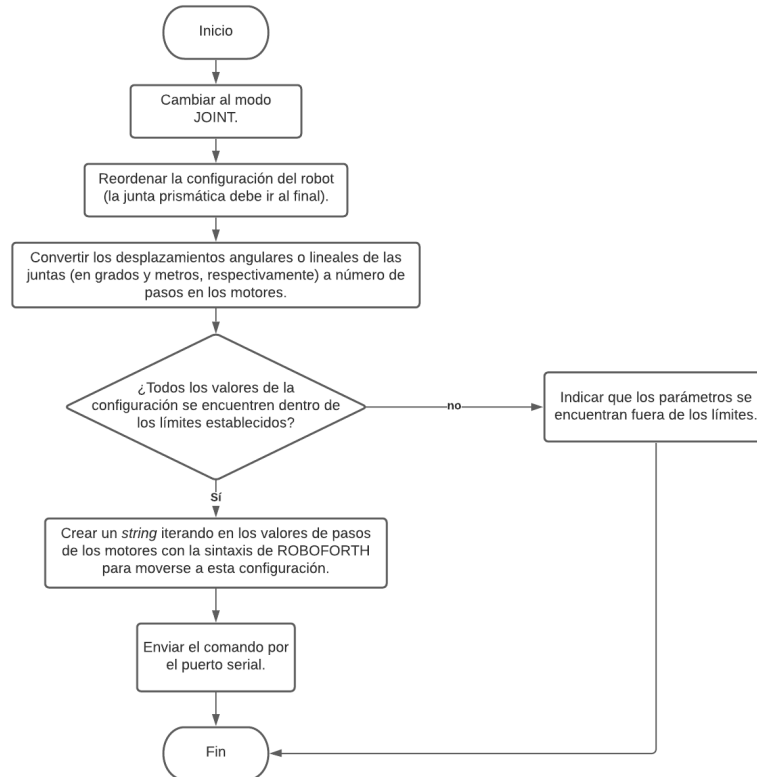


Figura 37: Diagrama de flujo de la función R17fKine.

Ya con la función implementada para controlar el R17, se puede tener una validación más cuantitativa del modelo cinemático planteado. El comando WHERE de ROBOFORTH II indica la posición en donde se encuentra el efector final del R17, por lo tanto, este se usó como valor teórico el cual se comparó con el resultado de efectuar la cinemática directa del manipulador con los valores planteados en el Cuadro 3. Para obtener los valores del modelo planteado se implementó en MATLAB una función que ejecutara la ecuación (24).

En el Cuadro 7 se muestran los porcentajes de error de la coordenada x , y y z entre los valores de la posición obtenida por el comando WHERE (ver las Figuras 58 a la 62 en la sección de Anexos) y los obtenidos por el algoritmo de cinemática inversa del modelo planteado. Se tomó la configuración inicial del robot (correspondiente a la posición denominada HOME en ROBOFORTH II) y 4 configuraciones aleatorias más. Cabe resaltar que el comando WHERE no contempla la influencia que tiene la junta prismática en el robot, ya que esta es un accesorio adicional que se le puso al manipulador. Por tal razón, en las pruebas para validar el modelo se tendrá un constante de cero para esta junta.

Configuraciones de prueba						
q_i	Career (m)	Waist ($^\circ$)	Shoulder ($^\circ$)	Elbow ($^\circ$)	Hand ($^\circ$)	Wrist ($^\circ$)
q_1	0	0	0	0	0	0
q_2	0	0	40	0	0	0
q_3	0	89	38	-37	0	0
q_4	0	-100	-64	12	90	0
q_5	0	77	-25	-64	56	79

Cuadro 4: Configuraciones de prueba para comparar el resultado del modelo cinemático planteado con el real del R17.

Coordenadas teóricas			
Configuración	x (mm)	y (mm)	z (mm)
q_1	0.00	0.00	750.00
q_2	0.00	477.50	578.10
q_3	235.30	-3.60	672.00
q_4	612.60	130.50	404.20
q_5	-521.60	-101.00	348.00

Cuadro 5: Coordenadas x , y y z teóricas para las configuraciones mostradas en el Cuadro 4.

Coordenadas del modelo planteado			
Configuración	x (mm)	y (mm)	z (mm)
q_1	0.00	0.00	750.00
q_2	0.00	482.09	574.53
q_3	237.38	4.14	670.45
q_4	622.94	109.84	395.26
q_5	-519.75	-119.99	346.41

Cuadro 6: Coordenadas x , y y z del modelo planteado para las configuraciones mostradas en el Cuadro 4.

Porcentajes de error			
Configuración	x (%)	y (%)	z (%)
q_1	0.00	0.00	0.00
q_2	0.00	0.96	0.62
q_3	0.88	215.10	0.23
q_4	1.69	15.83	2.21
q_5	0.35	18.81	0.46

Cuadro 7: Porcentajes de error de las coordenadas x , y y z del modelo planteado (Cuadro 6) y los valores teóricos obtenidos por el comando WHERE (Cuadro 5) del R17 para las configuraciones mostradas en el Cuadro 4

Los valores del Cuadro 7 muestran que para las coordenadas x , y y z los porcentajes de error no superan al 18.81%, por lo que puede mencionarse que el modelo del manipulador serial planteado es válido. A pesar de esto, existe una anomalía en el porcentaje de error para la coordenada y de la configuración de prueba q_3 , la cual muestra un error de 215%. Este alto porcentaje de error se ve explicado por el hecho que el valores teórico y experimental en este punto son de -3.60 mm y 4.14 mm respectivamente; la medición experimental es grande relativa a la teórica, sin embargo, una discrepancia de tan solo 7.74 mm no invalida el modelo generado, más porque la cinemática directa actúa en lazo abierto.

9.3. Implementación de la cinemática inversa

Tal como en la cinemática directa, el R17 tiene un modo de operación especializado para la cinemática inversa, este se denomina *CARTESIAN mode*. Este modo tiene una sintaxis más simple que la de la cinemática directa, pues no involucra cálculos a pasos de los motores y la entrada tiene menos parámetros. *CARTESIAN mode* toma como entrada el vector de posición deseada con coordenadas x , y y z del efector final del R17; con esta información, el robot hace el cálculo de los valores que debe de tener cada junta para poder alcanzar esta posición y se moviliza a esta configuración. A diferencia del modo *JOINT*, este modo sí cuenta con una programación defensiva, la cual arroja un mensaje indicando si una posición ingresada se encuentra fuera de los límites del espacio de trabajo del manipulador y dejando al robot en su última posición.

La creación de los comandos en ROBOFORTH II que se encargan de controlar el R17, por medio de cinemática inversa, se hacen dentro de una función de MATLAB llamada R17iKine. Esta función toma como argumento un vector \mathbf{p} , el cual contiene la posición meta del efector final, y da como resultado la escritura en el puerto serial de estos comandos. El diagrama de flujo de esta función se muestra en la Figura 38.

Con el objetivo de seguir validando el modelo cinemático y adicionalmente la correcta operación del robot en su modo de cinemática inversa, se efectuaron algunas pruebas. El procedimiento de estas pruebas fue definir cinco posiciones meta del efector final teóricas, en donde se variaron tanto en el eje x , y y z para contemplar su movimiento en todas direcciones. Estas posiciones teóricas corresponden a las que toma el robot cuando se utiliza la función R17iKine. Luego, dichas coordenadas que fueron enviadas al R17 por medio R17iKine, fue-

ron tomadas como parámetros para obtener la configuración del robot cuando se desea esa posición meta en el efector final, haciendo uso de otra función creada en MATLAB donde se desarrolla el algoritmo para cinemática inversa mostrado en la Figura 15, el cual implementa la ecuación (35). Dicha configuración fue ingresada en la función R17fKine, la cual implementa la cinemática directa del robot, para luego, mediante al comando WHERE obtener una posición, la cual fue comparada con los valores teóricos inicialmente ingresados en la función de la cinemática inversa en el R17.

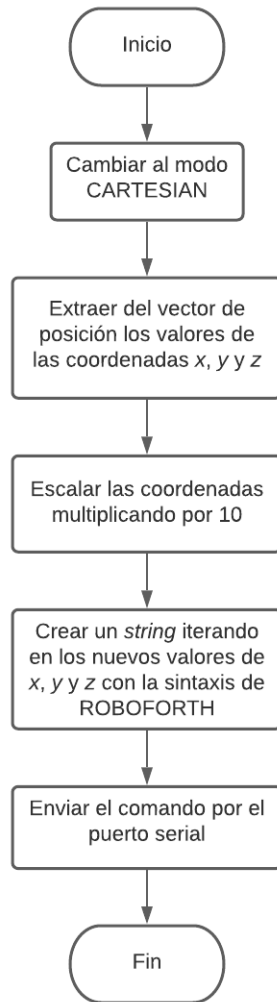


Figura 38: Diagrama de flujo de la función R17iKine.

El Cuadro 8 muestra las cinco posiciones meta definidas para estas pruebas; las coordenadas representan los valores teóricos. En el Cuadro 9 se muestran las configuraciones que devolvió el algoritmo de cinemática inversa planteado con el modelo representado en la convención DH para cada una de las posiciones meta del Cuadro 8. Se pudo identificar que el modo *CARTESIAN* del R17 no toma en cuenta la junta prismática, ya que al momento de enviar los comandos nunca hubo un desplazamiento en el riel; por otra parte, el algoritmo basado en la implementación de la ecuación (35) si contempla al riel como un grado

de libertad adicional para poder alcanzar esta posición de meta. Por la misma razón, se muestra un cuadro con la información de la posición obtenida al utilizar las configuraciones que se muestran en el Cuadro 9 y otro con la posición obtenida luego de hacer el ajuste de la junta prismática, el cual se ve reflejado como un desplazamiento sobre el eje y . Los valores de posición del efector final para estos dos casos se muestran en los Cuadros 10 y 11, respectivamente.

Posiciones meta de efector final (teóricos)			
Vector de posición	x (mm)	y (mm)	z (mm)
p_1	0.00	0.00	750.00
p_2	0.00	0.00	550.00
p_3	0.00	200.00	550.00
p_4	100.00	200.00	550.00
p_5	-150.00	120.00	400.00

Cuadro 8: Coordenadas x , y y z teóricas de las posiciones meta del efector final, utilizadas en la función para la cinemática inversa R17iKine.

Configuraciones obtenidas de cinemática inversa						
p_i	Career (m)	Waist ($^\circ$)	Shoulder ($^\circ$)	Elbow ($^\circ$)	Hand ($^\circ$)	Wrist ($^\circ$)
p_1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
p_2	-0.4892	0.0000	30.6118	22.0852	0.0000	0.0000
p_3	-0.2918	0.0000	31.4700	20.6687	0.0000	0.0000
p_4	-0.2821	11.7195	31.6383	20.3887	0.0000	0.0000
p_5	-0.4692	-14.2838	42.6675	27.9808	0.0000	0.0000

Cuadro 9: Configuraciones obtenidas del algoritmo de cinemática inversa al emplear las posiciones meta del efector final del Cuadro 8.

Posiciones meta de efector final			
Vector de posición	x (mm)	y (mm)	z (mm)
p_1	0.00	0.00	750.00
p_2	0.00	498.60	541.50
p_3	0.00	485.10	555.20
p_4	104.00	479.40	551.90
p_5	-149.60	588.00	401.40

Cuadro 10: Coordenadas x , y y z experimentales sin ajuste del efecto que tiene la junta prismática; posiciones de efector final alcanzadas al utilizar las configuraciones del Cuadro 9 en la función R17fKine.

Posiciones meta de efector final			
Vector de posición	x (mm)	y (mm)	z (mm)
p_1	0.00	0.00	750.00
p_2	0.00	9.40	541.50
p_3	0.00	193.30	555.20
p_4	104.00	197.30	551.90
p_5	-149.60	118.80	401.40

Cuadro 11: Coordenadas x , y y z experimentales con ajuste del efecto que tiene la junta prismática; posiciones de efector final alcanzadas al utilizar las configuraciones del Cuadro 9 en la función R17fKine.

El ajuste de la junta prismática para las posiciones de meta en el efector final mencionados con anterioridad corresponde a sumarle a la coordenada en y el valor en milímetros de la junta del riel. Por ejemplo, para la posición de prueba p_2 , se obtuvo un valor de 498.60 mm (ver Cuadro 10); aplicar el ajuste significa sumarle a este valor los -0.4892 m que presenta la junta prismática (ver Cuadro 9), lo cual da como resultado los 9.40 mm que se muestran en el Cuadro 11. Esto tiene sentido, ya que el riel presenta su desplazamiento a lo largo de un eje paralelo al eje y . Esto también ilustra que el modo de cinemática inversa incorporado en el R17, CARTESIAN mode, es restrictivo en el espacio de tarea del robot, ya que no aprovecha la junta prismática.

El Cuadro 12 muestra que no se presentaron errores mayores al 4% en lo que respecta a la comparación de la cinemática inversa implementada por el controlador del R17 y a la del algoritmo basado en el modelo planteado con la convención DH. Esto significa 2 cosas: en primer lugar, que la función para implementar la cinemática directa en el R17 ha sido desarrollada correctamente; en segundo lugar, que el modelo cinemático planteado para el manipulador serial sigue presentando un alto nivel de confiabilidad. Dentro de este mismo cuadro se puede ver la abreviatura N/A, la cual significa "no aplica"; fue utilizada debido a que existía un valor teórico de 0, en cuyo caso no se puede calcular el porcentaje de error.

Porcentajes de error			
Vector de posición	x (%)	y (%)	z (%)
p_1	0.00	0.00	0.00
p_2	0.00	N/A	1.55
p_3	0.00	3.35	0.95
p_4	4.00	1.35	0.35
p_5	0.27	1.00	0.35

Cuadro 12: Porcentajes de error de las coordenadas x , y y z al usar el algoritmo de cinemática inversa del modelo planteado (Cuadro 11) y los valores teóricos ingresados en la función R17iKine (Cuadro 8).

Los valores mostrados en el Cuadro 10 fueron obtenidos por el comando WHERE del R17 al ingresar la configuración del Cuadro 9 en la función R17fKine. Los mensajes desplegados por el R17 al usar este comando se muestran en las figuras 63 a la 67 de la sección de Anexos.

9.4. Implementación de ejecución de trayectorias

El R17 tiene la capacidad de poder ejecutar trayectorias. Una trayectoria no es más que un conjunto de puntos que sigue un cuerpo en movimiento, en este caso, el efector final. Dentro de la documentación del R17 también se les conoce como rutas y se definen como una lista de posiciones que pueden ser corridas mediante el comando RUN. Cada posición dentro de esta lista pueden ser un vector de configuración, para trabajar las rutas en modo JOINT, o coordenadas x , y y z del efector final, para trabajarlas en el modo CARTESIAN. En este trabajo de graduación se trabajan las coordenadas en el modo cartesiano, ya que representa una trayectoria en el espacio de tarea del robot, lo cual es más interpretable y útil.

Existen tres modos diferentes de poder correr una trayectoria, estos son: *segmented*, *continuous* y *smooth*. El primer modo implica que el efector final se colocará de manera exacta en cada una de las posiciones de la ruta a medida que este se va moviendo, por lo general esto implica que el robot acelerará y desacelerará para poder alcanzar dichas posiciones. El segundo modo sirve si se quiere que el robot pase por estas posiciones sin detenerse en cada una de ellas y manteniendo una velocidad constante a lo largo de todo el trayecto. No obstante, en realidad el robot no atraviesa estas posiciones exactamente, sino que pasa cercanos a ellos a medida que cambia de dirección para alcanzar el siguiente punto. Existe la posibilidad que en el modo continuo (*continuous*) el robot no pueda desempeñar la trayectoria, en cuyo caso el manipulador serial responde con el mensaje "too tight, line ...", lo cual significa que el R17 no podrá acercarse lo suficiente al siguiente punto sin modificar la velocidad establecida. Cuando este problema se presenta se utiliza el último modo, el modo suave (*smooth*), el cual ajusta la velocidad del robot para poder desempeñar la trayectoria de manera continua pero variando su velocidad.

La mejor analogía para entender el modo continuo es la de una serie de cruces a 90° . En la Figura 39 se muestran dos diagramas, el de la izquierda (diagrama 1) en el que el modo continuo sí se pudo ejecutar y el de la derecha (diagrama 2) en el que se obtuvo el error "too tight, line ...". En el diagrama 1, el robot redondea la esquina en la línea 2 con éxito a tiempo para comenzar a redondear la esquina en la línea 3. Pero en el diagrama 2, las líneas 2 y 3 están demasiado juntas para que esto sea posible. Esto resultaría en el error. Las soluciones son reducir la velocidad o aumentar la aceleración. Cualquiera resultaría en un giro más cerrado que podría encajar en el tiempo y distancia permitida para el giro.



Figura 39: Ejecución de trayectoria en modo continuo 34.

Se desarrolló una función MATLAB llamada R17CreateRoute. Esta se encarga de almacenar en memoria la lista de posiciones para una trayectoria que se desee ejecutar en el robot. Cabe aclarar que para que el robot pueda crear y guardar esta ruta, este debe de colocarse en cada una de las posiciones de la trayectoria y “aprender” dicho punto. Otro aspecto importante es que antes de guardar las rutas, se debe de reservar en memoria n espacios, que corresponden a la cantidad de líneas que tendrá la trayectoria; el máximo de líneas es 4,000. En la Figura 40 se muestra el diagrama de flujo de la función R17CreateRoute.

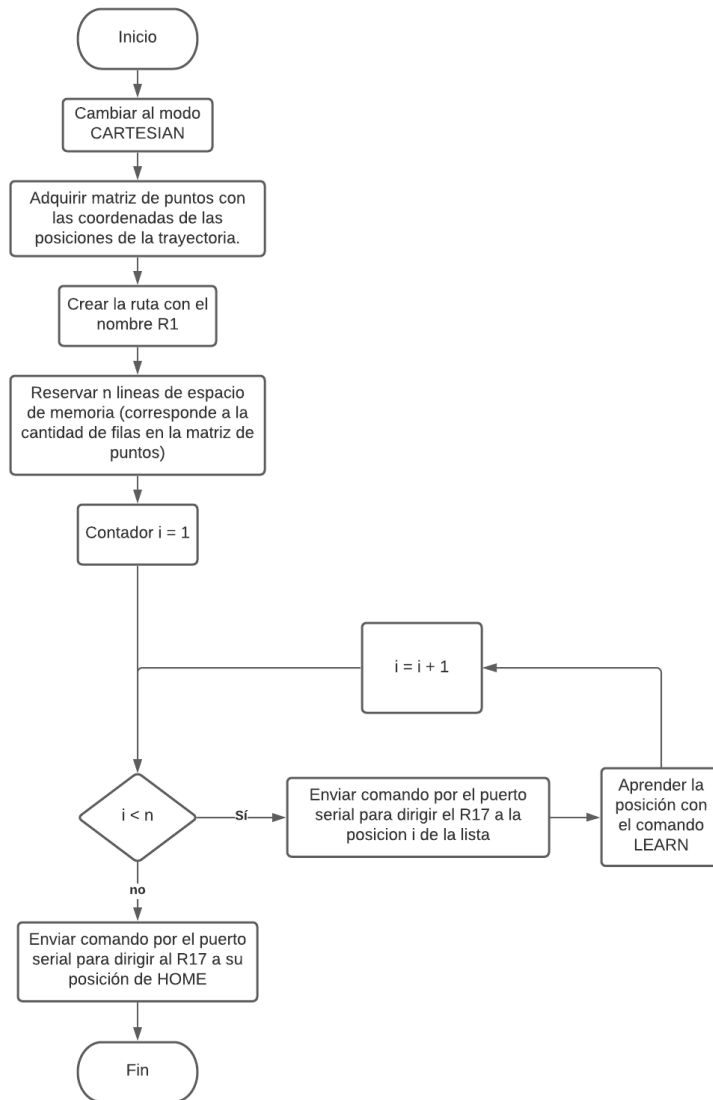


Figura 40: Diagrama de flujo de la función R17CreateRoute.

Se realizaron pruebas con ayuda del software Tracker [35]. Tracker es un programa que permite el análisis de movimientos en 1 y 2 dimensiones, siendo capaz de generar tablas y gráficos de posición, velocidad, aceleración, entre otros; el programa también es capaz de exportar los datos a archivos CSV. El objetivo de estas pruebas fue determinar las características de la trayectoria ejecutada en los modos segmentado y continuo del R17. Para el correcto funcionamiento del programa, se debe de colocar una marca distintiva (con colores contrastantes), a modo que Tracker pueda hacer un seguimiento de la posición en cada instante del efector final, que es el objeto que interesa.

Se definió una trayectoria con siete posiciones, esta se muestra en el Cuadro 13. Ya que Tracker es capaz de analizar el movimiento en un máximo de dos dimensiones, las posiciones de esta trayectoria fueron variantes únicamente en lo que respecta a las coordenadas y y z del espacio de trabajo del R17. Los pasos a seguir en Tracker para hacer el análisis de las trayectorias fueron los siguientes:

1. Grabar un vídeo del R17 efectuando la trayectoria en el modo segmentado y en el modo continuo, procurando que sea la más de frente posible.
2. Abrir los archivos de video en Tracker.
3. Definir un origen, que concuerde con la junta del hombro del R17.
4. Establecer una medida de calibración, en este caso una distancia de 750 mm desde el origen hasta el efector final cuando el R17 se encuentra en la posición HOME.
5. Crear un masa puntual sobre el marcador colocado en el efector final del R17.
6. Ejecutar la función de *autotracker*.
7. Desplegar en la tabla de información el tiempo, las coordenadas x y y , y la velocidad lineal. Cabe mencionar que las coordenadas x y y de tracker cooresponden a las y y z del R17, respectivamente.
8. Exportar los datos tomados a un archivo CSV, para poder analizar la información en MATLAB.

Trayectoria de prueba			
Línea	x (mm)	y (mm)	z (mm)
1	0.00	0.00	750.00
2	0.00	0.00	550.00
3	0.00	200.00	550.00
4	0.00	450.00	400.00
5	0.00	100.00	350.00
6	0.00	550.00	300.00
7	0.00	300.00	0.00

Cuadro 13: Tabla de posiciones cartesianas de trayectoria de prueba.

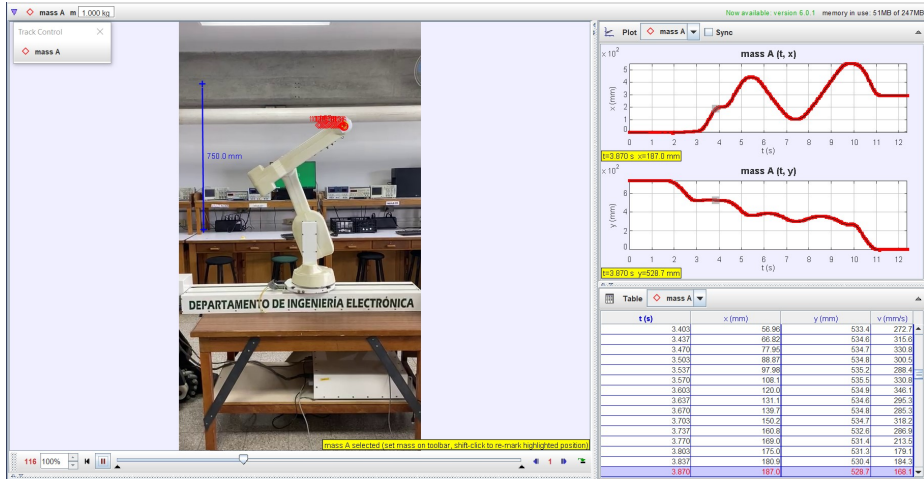


Figura 41: Análisis en Tracker de trayectoria corrida en modo segmentado.

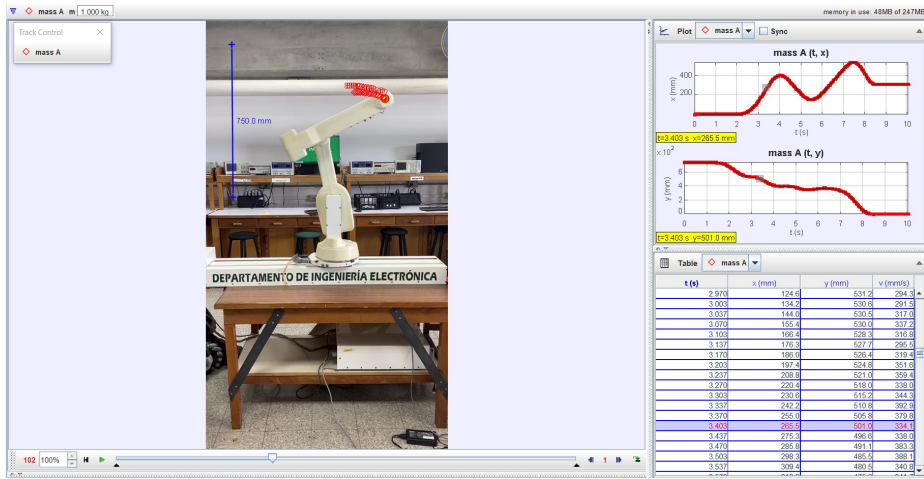


Figura 42: Análisis en Tracker de trayectoria corrida en modo continuo.

Con los datos obtenidos en Tracker, se hizo un análisis gráfico en MATLAB. Se obtuvo un mapa coordenado del efector final y una gráfica de velocidad lineal vs tiempo, en ambos casos tanto para el modo segmentado como para el continuo.

La gráfica de la Figura 43 muestra que durante el modo segmentado, la posición del efector final está muy cercana a los valores teóricos mostrados en el Cuadro 13. El error presente puede deberse a las calibraciones de la mediciones dentro de Tracker y a la perspectiva del vídeo tomado. También se aprecia esquinas o picos en los cambios de dirección, lo cual implica un cambio brusco en la dirección del efector final, el cual también se ve explicado en la gráfica de velocidad de la Figura 44. En la gráfica de velocidad se ve que el valor se mantiene o toca los 0 mm/s en siete ocasiones, lo cual implica que el robot se detiene en cada posición de la tabla de trayectoria, lo cual genera los picos de la gráfica de posición. Además, el efector final presenta vibraciones en los cambios de signo de aceleración, lo cual se observa en los picos de la gráfica. En conclusión, el modo segmentado tiene una mayor cercanía a las posiciones definidas, sin embargo presenta cambios abruptos en la velocidad lineal del efector final, aparte que se ve obligado a detenerse (llevar su velocidad a 0 mm/s) en cada posición de del Cuadro 13.

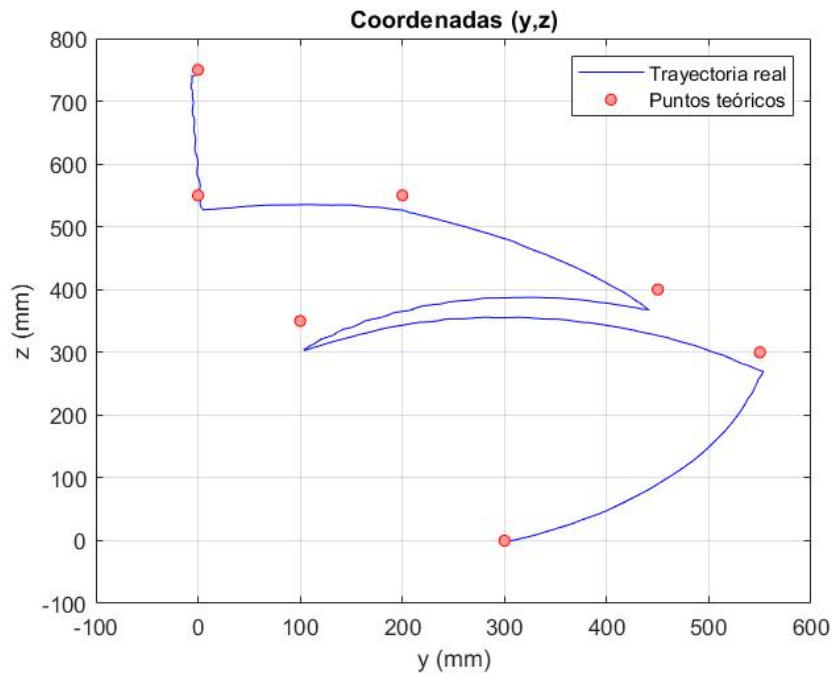


Figura 43: Coordenadas y y z de la trayectoria de prueba del efector final corrida en modo segmentado.

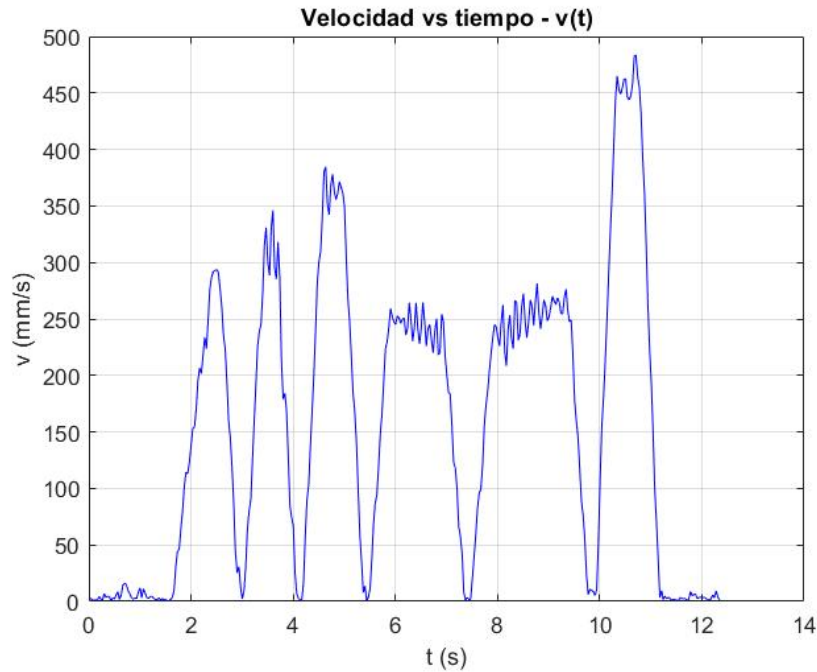


Figura 44: Velocidad lineal de la trayectoria de prueba del efector final corrida en modo segmentado.

La gráfica de la Figura 45, a diferencia de la presentada en el modo segmentado, no posee las esquinas distintivas del mencionado anteriormente. Por otra parte, la trayectoria real presenta mayor error con respecto a los puntos teóricos en el modo continuo que en el segmentado. Esto tiene relación directa con la velocidad lineal del efector final, mostrada en la Figura 46. En el modo continuo los valores de 0 mm/s solo se presenta al inicio y al final de la trayectoria, en todo el tramo intermedio existen valores bajos de velocidad, pero nunca una velocidad nula. Además se puede observar que la velocidad presenta mayor continuidad, ya que se vieron atenuadas las oscilaciones en los picos de velocidad (cuando hay un cambio de signo en la aceleración). En conclusión, el modo continuo tiene un mayor error de posición, sin embargo posee mayor continuidad en la velocidad y puede desempeñar una trayectoria en un menor tiempo.

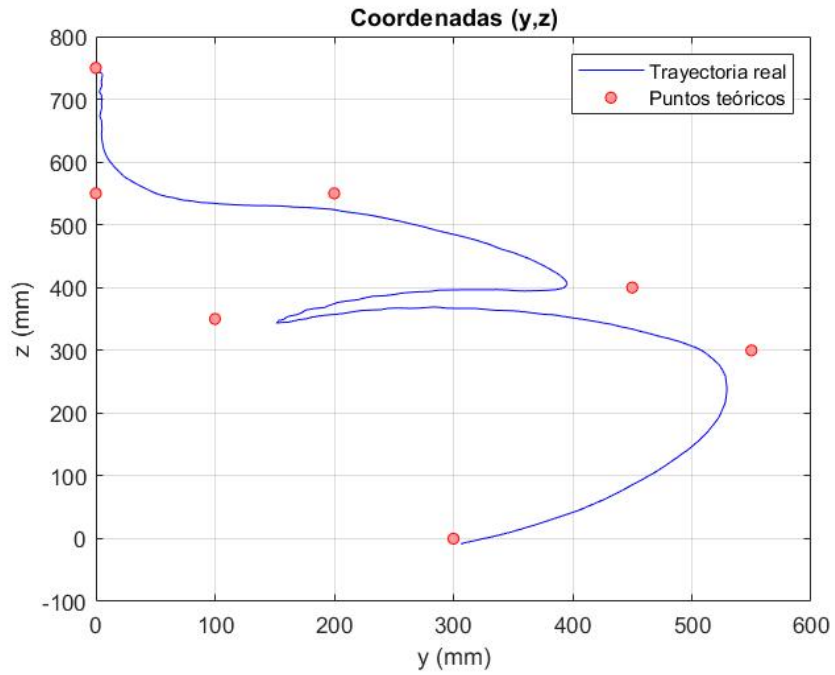


Figura 45: Coordenadas y y z de la trayectoria de prueba del efector final corrida en modo continuo.

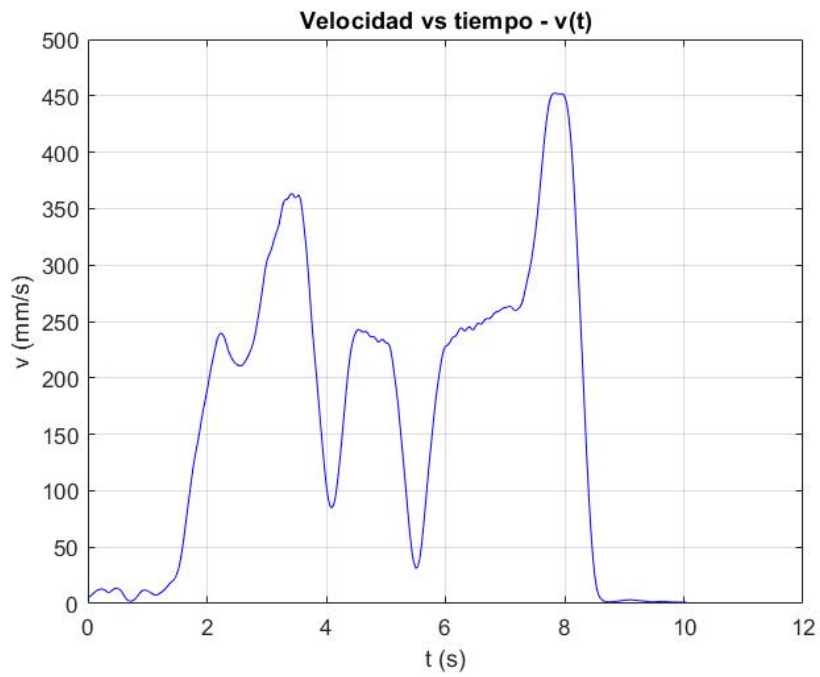


Figura 46: Velocidad lineal de la trayectoria de prueba del efector final corrida en modo continuo.

9.5. Interfaz gráfica

Con el fin de poder interactuar con el robot de una manera más intuitiva y simple, se creó una interfaz gráfica en MATLAB que integrara todas las funciones creadas anteriormente para controlar al R17. Este resultado es la culminación del tercer objetivo específico de este trabajo de graduación. Esta interfaz también corresponde al prototipo que será implementado en el microcontrolador ESP32, para luego poder ser integrado al sistema de captura de movimiento OptiTrack.

La interfaz cuenta con tres pestañas, donde cada una corresponde al modo de operación que se estará utilizando para controlar al R17. La pestaña principal comparte la funcionalidad de la conexión e inicialización del robot con la modalidad de cinemática directa. En la segunda pestaña se encuentra todo lo necesario para el uso del R17 en su modalidad de cinemática inversa. Finalmente, en la tercera pestaña, está la modalidad de ejecución de trayectorias.

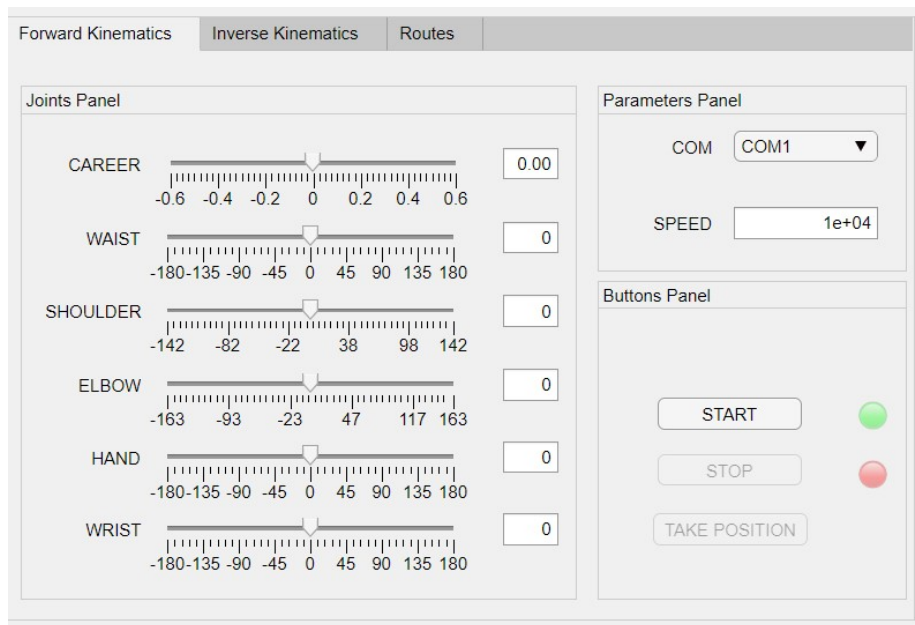


Figura 47: Pestaña principal.

A continuación se describe la pestaña de la interfaz para cada modalidad o funcionalidad.

- **Conexión e inicialización:**

Estas funcionalidades se encuentran en la misma pestaña que la modalidad de cinemática directa, específicamente en los paneles denominados *Parameters Panel* y *Button Panel*. Este proceso corresponde a seleccionar el puerto COM de la computadora por el cual se estará comunicando con el R17 y definiendo la velocidad con la que se desea que este se mueva. El botón de START solo puede ser presionado si el robot no presenta una conexión con la computadora todavía. Una vez la conexión ya se ha establecido, el botón de START se bloquea y el R17 responde con un *ok* por medio del puerto serial. La luz verde indica que el robot se encuentra conectado y la roja que está desconectado.

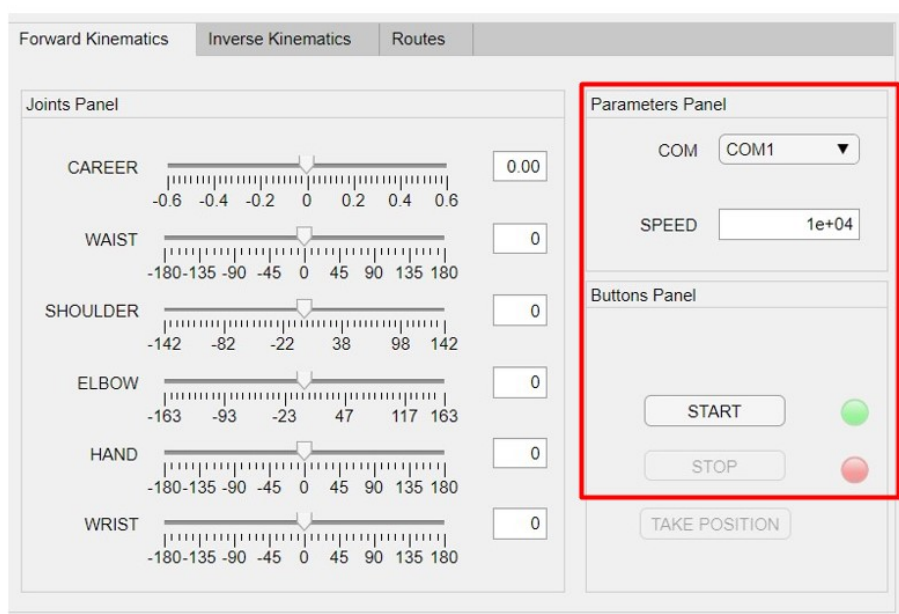


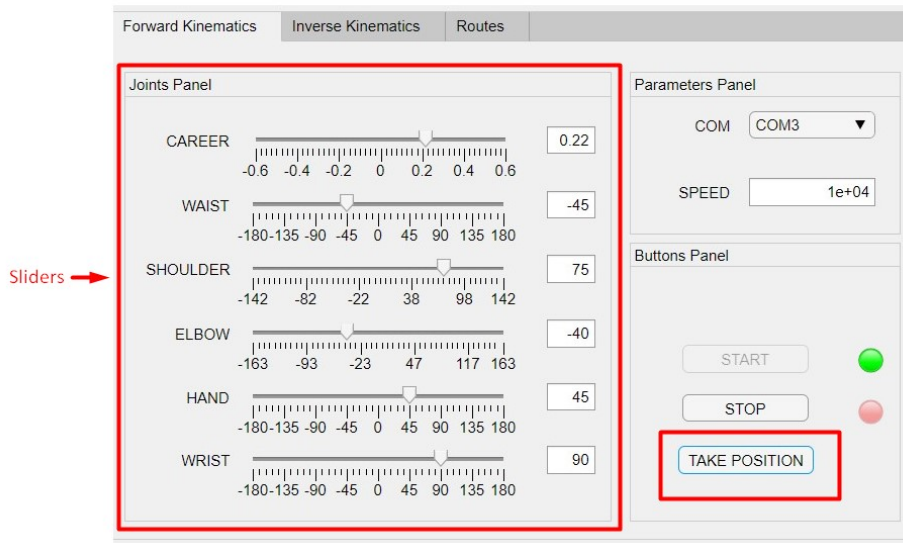
Figura 48: Pestaña principal: conexión e inicialización.

- **Cinemática directa:**

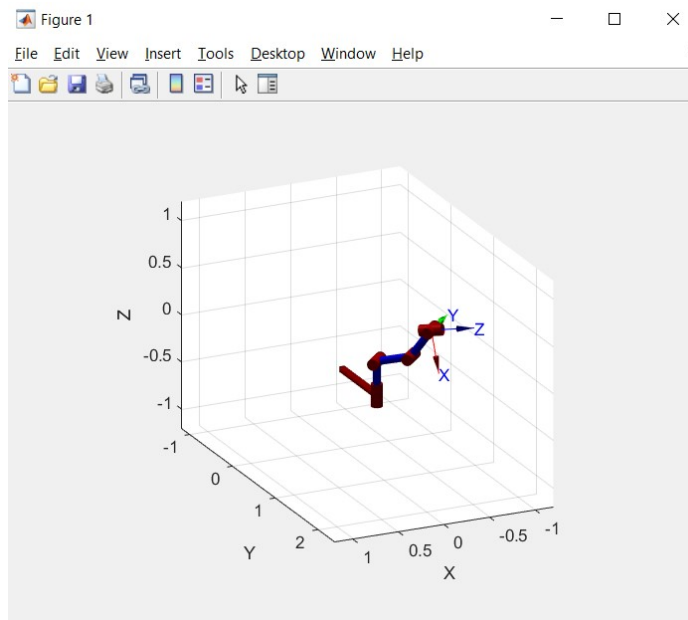
Esta modalidad funciona al hacer uso de dos tipos de elementos de la pestaña principal. El primer tipo de elemento son los deslizadores (*sliders*) de cada junta del R17. Como ya se ha mencionado anteriormente, el R17 cuenta con 6 juntas, las cuales pueden ser controladas individualmente en esta pestaña con los *sliders*. El primer *slider*, denominado CAREER corresponde a la junta prismática del manipulador serial y sus unidades están en metros. Para el resto de *sliders*, correspondientes a las juntas revolutas, las unidades de medida están en grados. Se colocó a la par de los sliders unos cuadros de entrada, los cuales solo aceptan valores numéricos en el intervalo permitido por cada junta, para poder tener un manejo más exacto de la magnitud que se quiere tener en cada junta.

Cuando se utiliza esta modalidad, la interfaz despliega una figura donde se puede visualizar una simulación de la geometría actual del R17 según los valores actuales de

los *sliders*. El objetivo de esta simulación es presentar la configuración que tomará el manipulador real antes de enviar el comando indicando que tome dicha configuración. Este comando se envía al interactuar con el segundo elemento, el botón de TAKE POSITION. Al presionar este botón la interfaz manda a llamar la función de cinemática directa creada, en donde esta toma los valores en los sliders y como respuesta envía por el puerto serial la instrucción en ROBOFORTH II. El resultado real es que el R17 toma la posición que se visualiza en la simulación de la interfaz con la velocidad previamente establecida.



(a) Pestaña principal: cinemática directa.



(b) Simulación de la geometría del R17.

Figura 49: Interfaz gráfica para modalidad de cinemática directa.

- **Cinemática inversa:**

Esta pestaña cuenta con tres cuadros de entrada numéricos, uno para cada valor de las coordenadas x , y y z del efector final. También se despliega una gráfica 3D, en donde el punto rojo representa la posición del efector final en el espacio de tarea del robot. Para poder hacer válida esta posición en el R17 se debe de presionar el botón de TAKE POSITION. Al momento de presionar este botón, la interfaz manda a llamar la función de cinemática inversa creada, en donde esta toma los valores de las coordenadas ingresadas en los cuadros de entrada y como respuesta envía por el puerto serial la instrucción en ROBOFORTH II. El resultado en el robot real es que el efector final del R17 se posiciona en las coordenadas establecidas.

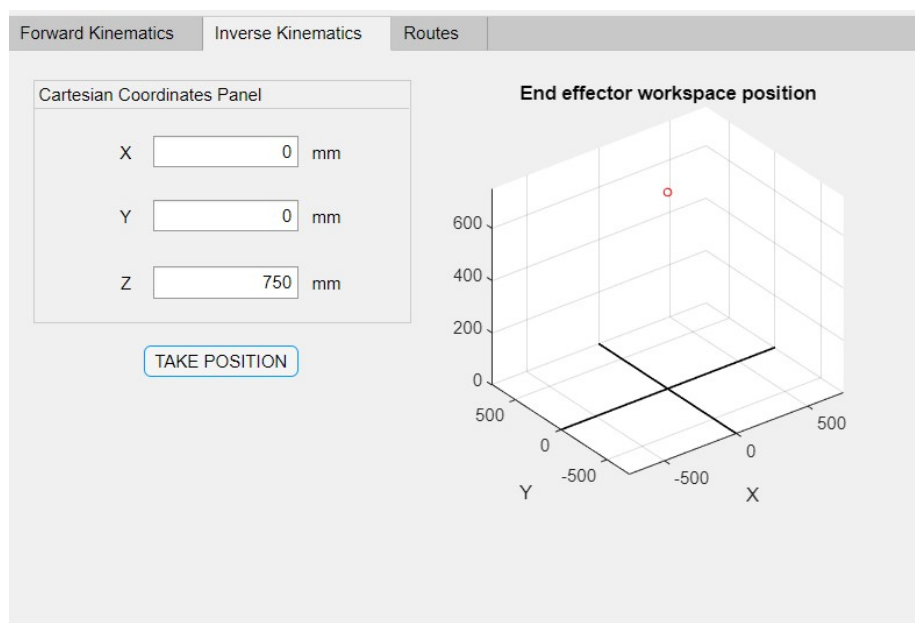


Figura 50: Interfaz gráfica para modalidad de cinemática inversa.

- **Trayectorias:**

Para desempeñar una trayectoria de manera exitosa con el R17 primero se debe de ingresar la cantidad de puntos por los cuales se desea que pase el efector final. Esto se ingresa en la opción de *Route lines* del panel *Route options panel*. Este valor también corresponde a la cantidad de líneas que tendrá la tabla de puntos, presente en el panel *Route table panel*. Posteriormente, se debe de ingresar los valores de estos puntos en la tabla, indicando las coordenadas x , y y z . Cabe mencionar que esta tabla adecúa su tamaño según la cantidad de líneas que se haya ingresado para la trayectoria. Una vez llenada la tabla, se debe de crear la trayectoria, para esto se debe de presionar el botón de *Create route* (este se encuentra habilitado únicamente si no existe una trayectoria previamente creada, de lo contrario esta debe de ser eliminada de primero al presionar el botón de *Erase route*). Luego, se selecciona en el panel *Play route panel*, el modo en que se desea correr la trayectoria, escogiendo entre las opciones que da el bloque *Route run mode*. Por último, se puede correr toda la trayectoria de corrido al presionar el botón *Run route* o se puede ir a una posición específica de la trayectoria al escogerla en el desplegable (*drop down*) *Route line* y presionando el botón *Go to line*.

Tras finalizar este proceso, la interfaz manda a llamar la función de creación de trayectorias, en donde esta toma la tabla de datos con las posiciones y como respuesta envía por el puerto serial el conjunto de instrucciones en ROBOFORTH II. El resultado en el robot real es que el efector final se posiciona en cada una de las coordenadas de la tabla, aprendiendo su trayectoria, y luego ejecutándola en el modo seleccionado.

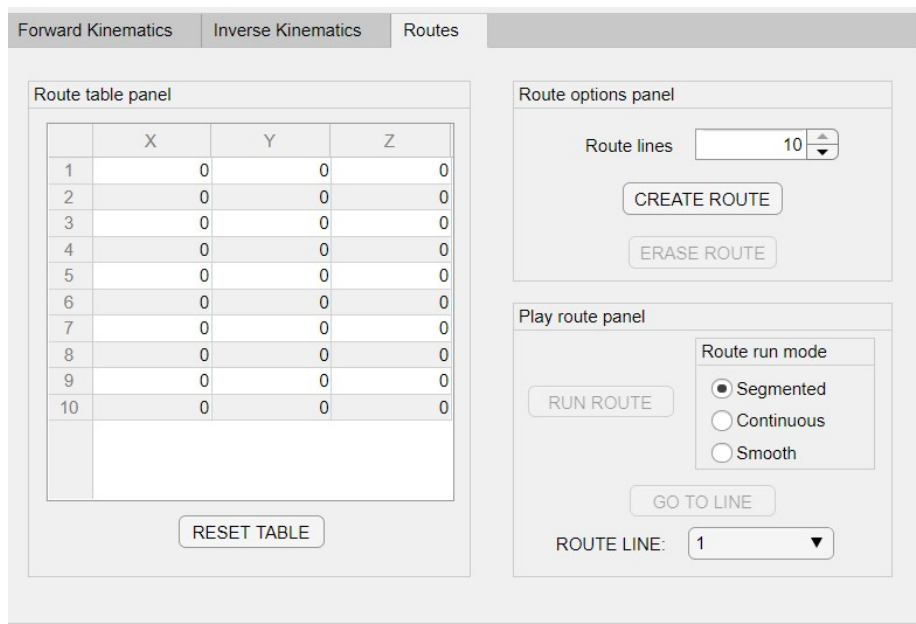


Figura 51: Interfaz gráfica para modalidad de ejecución de trayectorias.

Integración del manipulador serial R17 con el ecosistema Robotat

10.1. Sistema embebido: Robotat - R17 Module

Uno de los objetivos específicos de este trabajo de graduación es implementar un módulo de comunicación inalámbrica que permita la integración del manipulador R17 al ecosistema Robotat. Para cumplir con este objetivo, se decidió diseñar y fabricar un sistema embebido, el cual reciba toda la información necesaria del ecosistema y que este sea capaz de conectarse directamente al manipulador serial con la interfaz RS-232, la cual es la que ya tiene implementada el R17.

10.1.1. Selección de componentes

El Robotat - R17 Module cuenta con múltiples componentes electrónicos, los cuales en conjunto desempeñan la tarea del sistema embebido. El corazón del sistema es un microcontrolador, este debe ser capaz de poder recibir instrucciones por WiFi y enviar comandos en ROBOFORTH II. El sistema embebido también tiene incorporado su propia fuente de energía y un circuito para convertir los comandos al estándar RS-232. A continuación se detalla los componentes seleccionados.

- Microcontrolador:

Se utilizó el ESP32 [36], desarrollado por Espressif Systems. Este es un microcontrolador de bajo costo y consumo de energía, el cual tiene integrada tecnología WiFi y Bluetooth. Por sus características, el ecosistema Robotat ya tiene a este microcontrolador como requisito.

- Circuito convertidor de niveles de voltaje TTL a RS-232:

Se utilizó el circuito integrado MAX232 [37], frecuentemente empleado para convertir los niveles de las líneas de un puerto serie RS232 a niveles TTL y viceversa. Este posee una implementación muy sencilla, donde se necesita utilizar únicamente cuatro capacitores de $1\mu\text{F}$. Otra ventaja de este integrado es que es de fácil acceso con proveedores locales.

- Fuente de energía:

El ESP32 puede ser energizado de tres maneras: mediante un cable USB, con un voltaje regulado de 3.3V o con un voltaje no regulado entre 5V y 12V. Se desea eliminar cableado, por lo que se descarta la primera forma de alimentar al ESP32.

Se optó por utilizar dos baterías Li-ion 18650 Ewttto de 3,500mAh recargables [38]. Los criterios tomados en cuenta para esta selección fueron los siguientes: cada batería entrega un voltaje, cuando están en plena carga, de 3.7V, por lo que al colocarlas en serie son útiles para la forma 3 de alimentar al microcontrolador. Estas baterías son accesibles por distribuidores locales y tienen un precio moderado, además se puede montar al módulo mediante a sujetadores para PCB, los cuales también son accesibles por distribuidores locales.

El módulo también debe de energizar el MAX232, el cual funciona con un voltaje de entrada de 5V. El voltaje en el Max232 si debe ser regulado, por lo que se incorporó un regulador L7805cv [39].

Tomando todo esto en consideración, se hizo una medición del consumo máximo del módulo, el cual fue de 240 mA, para un uso continuo, es decir, si el modulo se encuentra enviando comandos en todo momento y usando las funcionalidades de WiFi y Bluetooth. Ya que las baterías tienen una capacidad de 3,500 mAh, esto significa que el modulo podría estar operando aproximadamente 15 horas en uso continuo.

10.1.2. Diseño electrónico y fabricación

El sistema embebido cuenta con una placa de circuito impreso (*Printed Circuit Board* o PCB por sus siglas en inglés). Esta se diseñó en el software Altium Designer [40]. A continuación se describe el esquemático de la placa, el cálculo para el ancho de los *tracks* y los componentes utilizados.

En la Figura [52] se muestra el esquemático del sistema embebido. Como se mencionó anteriormente, el módulo tiene tres partes importantes: alimentación, microcontrolador y circuito convertidor dual de señal TTL a RS-232. El componente BT1 BK-18650-PC4 corresponde al par de baterías Li-ion 18650 conectadas en serie para alimentar todo el circuito. Se agregó un interruptor de encendido y apagado, denominado en el esquemático como S1. El ESP32 (U2) es alimentado directamente por las baterías. Para el circuito del MAX232 (IC1), primero se regula el voltaje con un L7805cv (U1), el cual entrega un voltaje estable de 5V. También se añadió en el esquemático las conexiones para el conector hembra DB9, el cual se encarga de la transmisión asíncrona de datos según lo establecido por el estándar RS-232. En el Cuadro [14] se detallan todos los componentes usados en el sistema embebido.

El diseño del PCB se muestra en la Figura 68 de la sección de Anexos y la fabricación en la Figura 70 de la misma sección. Se fabricó un estuche de protección a base de acrílico para el módulo, este fue cortado con láser y diseñado en el software Inventor 41. El modelo 3D del diseño se muestra en la Figura 70 de la sección de Anexos.

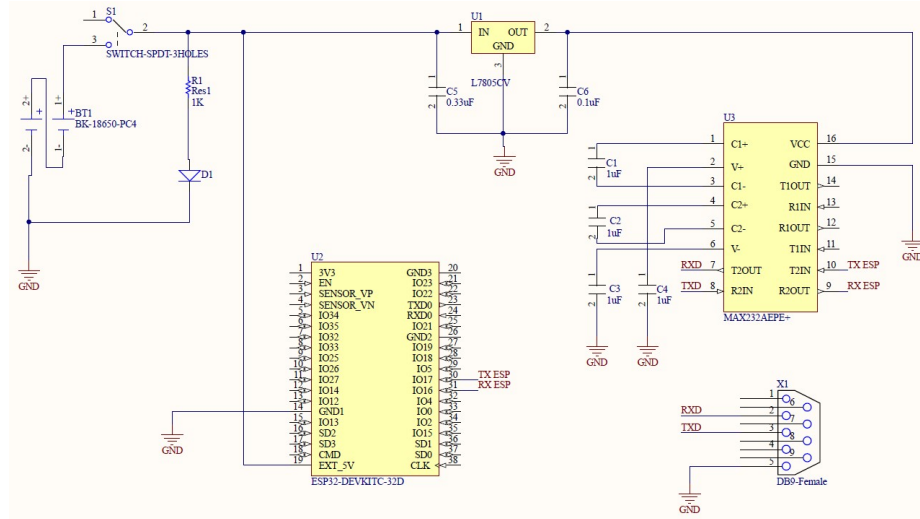


Figura 52: Esquemático de Robotat -R17 Module.

Componentes electrónicos		
Componente	Designador	Cantidad
Capacitor de $1\mu\text{F}$	C1, C2, C3, C4	4
Capacitor de $0.33\mu\text{F}$	C5	1
Capacitor de $0.1\mu\text{F}$	C6	1
Sujetador de baterías	BT1	1
Conector DB9 hembra	X1	1
LED rojo	D1	1
Resistencia de $1\text{k}\Omega$	R1	1
Interruptor SPDT de 3 agujeros	S1	1
Regulador de voltaje L7805cv	U1	1
ESP32	U2	1
MAX232	U3	1

Cuadro 14: Componentes electrónicos utilizados en el Robotat -R17 Module.

Tomando parámetros críticos en el circuito y siguiendo la norma IPC2221 para circuitos impresos 42 se determinó el ancho mínimo en los tracks del PCB. Las ecuaciones que determina este valor para capas externas se muestra en (41) y (42). Los parámetros definidos para efectuar este calculo fueron los siguientes:

- Corriente máxima (I): 240mA
- Aumento de temperatura (ΔT): 10°C

- Espesor de la placa de cobre (h): 35 μm

$$A = \left(\frac{I}{k\Delta T^{0.44}} \right)^{1/0.725} \quad (41)$$

$$w = \frac{A}{h} \quad (42)$$

Donde A corresponde a el área de la sección transversal del conductor y w es el ancho del mismo. Cabe mencionar que (41) regresa el valor del área en la unidad de medida mil^2 , por lo que se debe de hacer la conversión a mm^2 para poder utilizar el valor de h de 35 μm en (42). El resultado de emplear la norma IPC2221 es que el ancho mínimo del track que debe de ser utilizado en el PCB del Robotat - R17 Module es de 0.04 mm. Por recomendaciones del fabricante de la placa y facilidad en la soldadura, se utilizaron anchos en los *tracks* de 1 mm.

10.1.3. Validación del módulo

La señal de salida otorgada por el puerto serial del ESP32 corresponde a la de los niveles lógicos TTL de 3.3V. El estándar RS-232, por otra parte, toma como niveles altos valores de voltaje entre -5V y -15V; los niveles bajos de voltaje están entre +5V y +15V (43).

La Figura 53 muestra la señal de salida otorgada por el ESP32 y la señal de salida del Robotat - R17 Module. La señal en rojo muestra la del ESP32 y se puede ver que varía entre 0V y 3.3V. La señal azul corresponde a la salida por el conector DB9 del sistema embebido; tiene voltajes $\pm 8.2\text{V}$. Esta Figura valida el correcto funcionamiento del sistema embebido.

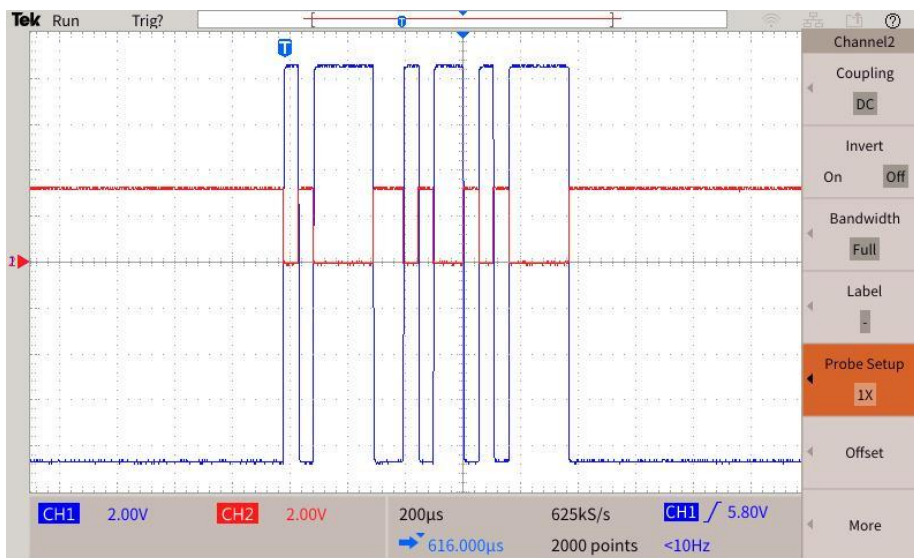


Figura 53: Señales de salida del ESP32 y el Robotat - R17 Module.

10.2. Librería R17

Se desea que el manipulador serial pueda desempeñar las mismas funcionalidades que las probadas y verificadas con la interfaz de MATLAB. Para ello se usó el microcontrolador ESP32 y en él se desarrolló una librería utilizando el entorno de trabajo esp-idf. Se optó por una programación orientada a objetos, el cual es un paradigma para el cual C no está diseñado, no obstante se puede simular con el uso de estructuras. El resultado fue entonces un pseudo-objeto usando estructuras y funciones globales que permiten simular los métodos.

El objeto, nombrado como *r17_t*, tiene ocho atributos. Al estar trabajando en C, estos son realmente arreglos de datos. Dichos atributos se describen en el Cuadro [15](#).

El objeto también cuenta con varios métodos (funciones) para controlar al R17. Se tienen tres métodos para configurar y utilizar el módulo UART2 del ESP32, 1 para obtener los datos del ecosistema Robotat y diez para funcionalidades especiales del R17. La descripción de estos métodos se presentan en el Cuadro [16](#).

Atributos	
Atributo	Descripción
<i>mode</i>	Contiene la información sobre el modo en que el R17 está trabajando (cinemática directa, cinemática inversa o trayectorias).
<i>speed</i>	Contiene el valor de la velocidad del manipulador serial.
<i>configuration</i>	Es un arreglo de seis posiciones que contiene la configuración del R17. Cada posición en el arreglo corresponde al valor en cada una de sus juntas.
<i>end_effector_position</i>	Es un arreglo de tres posiciones. Aquí se almacena las coordenadas <i>x</i> , <i>y</i> y <i>z</i> del efector final.
<i>trajectory_position_table</i>	Es una arreglo de tamaño variable que contiene las coordenadas <i>x</i> , <i>y</i> y <i>z</i> del efector final de cada uno de los puntos de la tabla de posiciones de la trayectoria. El tamaño se ve determinado por el número de puntos en esta tabla.
<i>table_points</i>	Contiene el total de puntos en la tabla de la trayectoria.
<i>route_run_mode</i>	Contiene la información sobre el modo en que se desea correr la trayectoria (segmentado, continuo o suave).
<i>gripper_status</i>	Contiene información acerca del estado de la pinza (<i>gripper</i>) del R17, es decir, si esta está abierta o cerrada.

Cuadro 15: Atributos del objeto *r17_t*.

Métodos		
Método	Parámetros	Descripción
<i>init_uart</i>	void ¹	Configura el módulo UART2 del ESP32 con un baudaje de 19,200 y un tamaño 8 bits. También habilita los pines RX y TX del microcontrolador.
<i>send_command</i>	Puntero del <i>string</i> que se desea enviar	Envía una cadena de caracteres (<i>string</i>).
<i>rx_task</i>	void	Lee y almacena la información que entra por el pin RX.
<i>get_from_robotat</i>	Puntero a <i>data_robotat</i>	Obtiene del Robotat toda la información relevante para el R17 y la almacena en variables.
<i>r17_update</i>	Puntero a <i>r17_t</i> Modo Velocidad Configuración Posición de efector final Tabla de posiciones Modo de trayectoria Estado del <i>gripper</i>	Actualiza toda la información importante recibida del ecosistema Robotat y lo almacena en el objeto <i>r17_t</i> .
<i>r17_init</i>	Puntero a <i>r17_t</i>	Inicializa el manipulador serial, lo cual significa que lo energiza, establece su velocidad y lo dirige a su posición HOME.
<i>r17_close</i>	void	Lleva al R17 a su posición HOME y lo des-energiza.
<i>r17_denergize</i>	void	Des-energiza al R17.
<i>r17_fkine</i>	Puntero a <i>r17_t</i>	Este método corresponde al modo de cinemática directa. Toma la configuración del robot y envía comandos a través del puerto serial para desempeñar su tarea.

Continúa en la siguiente página

¹El término *void* significa que no requiere parámetros

Cuadro 16 – *Continuación de la página previa*

Método	Parámetros	Descripción
<i>r17_ikine</i>	Puntero a <i>r17_t</i>	Este método corresponde al modo de cinemática inversa. Toma la posición deseada del efector final y envía comandos a través del puerto serial para desempeñar su tarea.
<i>r17_create_route</i>	Puntero a <i>r17_t</i>	Este método corresponde al modo de trayectorias. Toma la tabla de posiciones de la trayectoria y envía comandos a través del puerto serial para desempeñar su tarea. Es importante mencionar que este método solo crea la trayectoria, no la ejecuta.
<i>r17_erase_route</i>	void	Elimina la trayectoria previamente creada.
<i>r17_run_route</i>	Puntero a <i>r17_t</i>	Ejecuta la trayectoria previamente creada. Esta puede ser corrida en el modo segmentado, continuo o suave.
<i>r17_goto_line</i>	Número de línea	Lleva al efector final del R17 a la posición de una línea deseada de la tabla de posiciones de la trayectoria.

Cuadro 16: Métodos del objeto *r17_t*.

10.2.1. Interacción dentro del ecosistema Robotat

La librería mencionada anteriormente compone al objeto R17, el cual interactúa con los objetos GRIPPER y ROBOTAT, creados en otras librerías dentro del ecosistema por los estudiantes Rony Schumann [8] y Camilo Perafán [32], respectivamente. En la Figura 54 se muestra el diagrama UML que representa esta interacción.

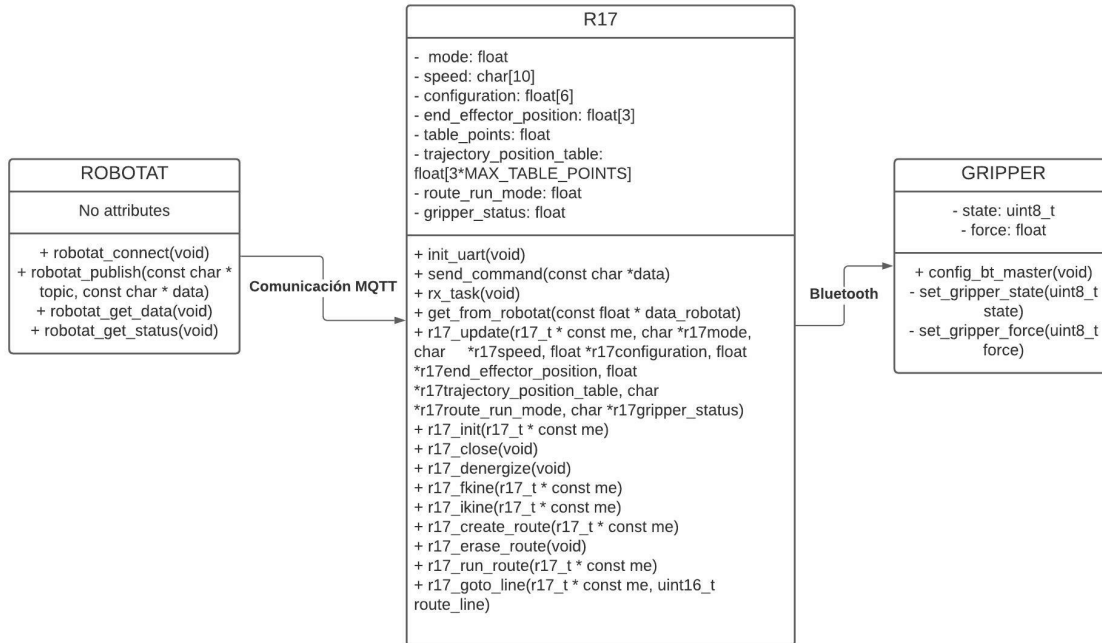


Figura 54: Diagrama UML.

10.3. Integración con OptiTrack

En esta sección se plantean pruebas que validan el correcto funcionamiento del R17 dentro del ecosistema Robotat, el cual tiene como base el *mocap* OptiTrack. El objetivo de estas pruebas es verificar que la información que despliega el Optitrack acerca de la pose del efector final del R17 concuerda con las instrucciones mandadas desde el Robotat para controlar al manipulador.

Se realizaron dos tipos de pruebas. La primera valida la modalidad de cinemática inversa y la segunda la ejecución de trayectorias. Cabe resaltar que la información de pose que muestra el OptiTrack es con respecto al marco de referencia inercial del *mocap* (el punto que se define como el *origen* en la calibración y configuración dentro de *Motive*); por otra parte, las instrucciones que se envían al R17 se encuentran con respecto al marco de referencia de la base del robot, el cual está en el centro de la junta *shoulder*. Esto implica que para poder comparar las posiciones por las que pasa el efector final del manipulador, se debe de hacer una serie de transformaciones (traslaciones y rotaciones) para poder referenciar el punto a analizar con respecto al mismo marco de referencia.

Estas transformaciones se ejemplifican con la Figura 55. El marco de referencia denotado por $\{I\}$ corresponde al origen del OptiTrack, $\{B\}$ es la base del robot y $\{E\}$ el efector final. La pose del efector final con respecto a la base se obtiene al aplicar las transformaciones de (43).

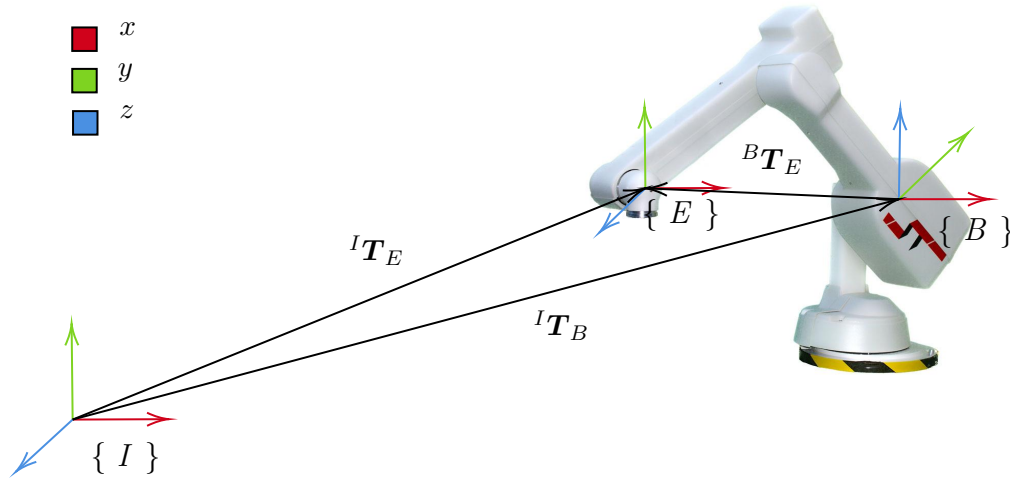


Figura 55: Diagrama de transformaciones.

$${}^B\mathbf{T}_E = {}^B\mathbf{T}_I {}^I\mathbf{T}_E = ({}^I\mathbf{T}_B)^{-1} {}^I\mathbf{T}_E \quad (43)$$

10.3.1. Cinemática inversa

La prueba que valida la modalidad de cinemática inversa consistió en enviar cinco posiciones meta desde el *script* en python del Robotat hacia el módulo inalámbrico y comparar su valor teórico (Cuadro 17) con el obtenido por el OptiTrack (Cuadro 18). Tal como en las pruebas del capítulo 8, se colocó un marcador de cuerpo rígido en el efector final del brazo robótico. Cabe resaltar que las posiciones mostradas en el Cuadro 18 son resultado de aplicar las transformaciones de (43).

Posiciones meta de efector final (teóricos)			
Vector de posición	x (mm)	y (mm)	z (mm)
p_1	0.00	0.00	750.00
p_2	0.00	0.00	400.00
p_3	0.00	350.00	400.00
p_4	0.00	-350.00	400.00
p_5	-300.00	0.00	400.00

Cuadro 17: Coordenadas x , y y z teóricas de las posiciones meta del efector final dentro del OptiTrack.

Posiciones meta de efector final (OptiTrack)			
Vector de posición	x (mm)	y (mm)	z (mm)
p_1	0.00	0.00	750.00
p_2	-2.09	1.03	395.78
p_3	0.73	351.68	395.24
p_4	52.89	-397.09	393.47
p_5	-295.41	-49.62	396.07

Cuadro 18: Coordenadas x , y y z de las posiciones meta del efector final obtenidas por el OptiTrack.

El error, a diferencia de las pruebas realizadas en el capítulo 9, no se calculó con respecto al valor de cada coordenada cartesiana. En este caso, se realizaron con respecto a las distancias desde el marco de referencia de la base del R17 hasta su posición de efector final. La razón es que existían variaciones en las coordenadas cuyo valor teórico era 0 mm, por lo tanto la expresión para el cálculo de porcentaje de error (error relativo) se indefinía.

Vector de posición	Error	
	Error absoluto (mm)	Error relativo (%)
p_1	0.00	0.00
p_2	4.21	1.05
p_3	2.45	0.46
p_4	30.00	5.64
p_5	3.41	0.68

Cuadro 19: Errores absolutos y relativos de las distancias de las posiciones meta del efector final teóricas (Cuadro 17) y las obtenidas por el OptiTrack (Cuadro 18).

Los valores del Cuadro 19 muestran errores absolutos por debajo de los 30 mm y errores relativos por debajo del 6%. Esto significa una poca desviación de la posición final del efector obtenida por el sistema de captura de movimiento. Las posibles fuentes de error pueden ser atribuidas a la calibración y reflejos del entorno que ingresan al espacio de observación del *mocap*, además del error con el que ya cuenta el R17 debido a juegos en los engranajes de sus mecanismos.

10.3.2. Ejecución de trayectorias

La prueba que valida la modalidad de ejecución de trayectorias consistió en hacer un análisis gráfico de la posición del efector final obtenida por el OptiTrack para un recorrido preestablecido. Este recorrido de prueba se definió con seis posiciones (Cuadro 20), las cuales corresponden a los puntos teóricos por los cuales debería de pasar el efector final. Se muestra el resultado de correr la trayectoria en su modo segmentado en la Figura 56 y en su modo continuo en la Figura 57.

Trayectoria de prueba			
Línea	x (mm)	y (mm)	z (mm)
1	0.00	0.00	750.00
2	0.00	0.00	550.00
3	-200.00	0.00	550.00
4	-200.00	300.00	350.00
5	-600.00	0.00	100.00
6	-200.00	-300.00	350.00

Cuadro 20: Tabla de posiciones cartesianas de trayectoria de prueba dentro del OptiTrack.

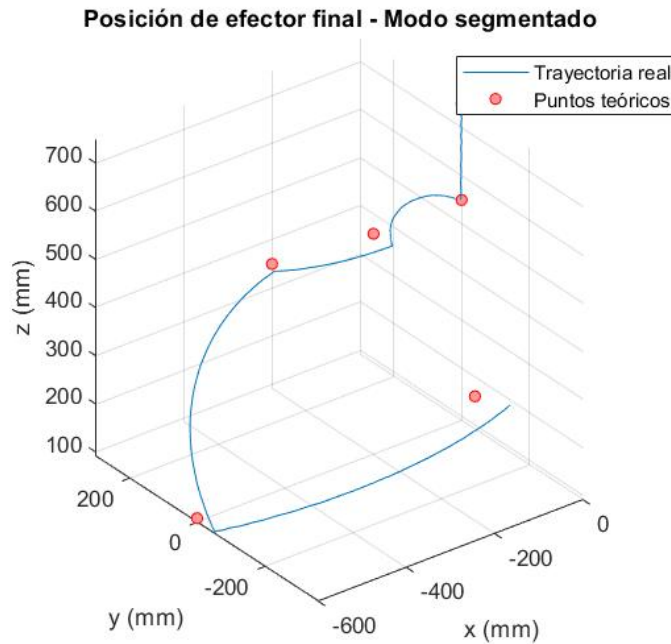


Figura 56: Coordenadas y y z de la trayectoria de prueba del efector final corrida en modo segmentado dentro del OptiTrack.

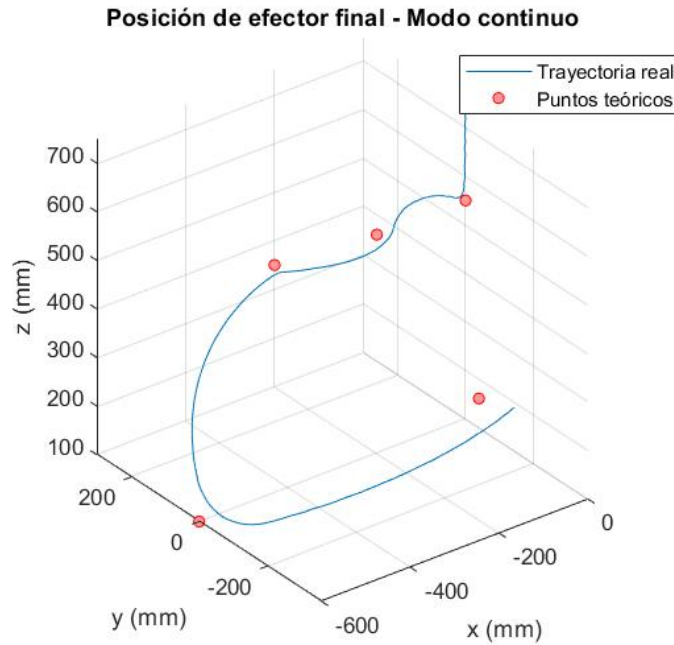


Figura 57: Coordenadas y y z de la trayectoria de prueba del efector final corrida en modo continuo dentro del OptiTrack.

Tanto la gráfica de la Figura 56 como la de la Figura 57 nos muestran una trayectoria con sentido para alcanzar los puntos deseados. El comportamiento en el recorrido también hace sentido con los resultados obtenidos en el capítulo 9. Estas pruebas validan la correcta integración del R17 con el ecosistema Robotat.

- Se logró obtener información acerca de la pose del efector final del R17 utilizando el sistema de captura de movimiento OptiTrack, el cual fue verificado cualitativamente mediante a las pruebas del capítulo 8.
- El modelo cinemático basado en la convención de Denavit-Hartenberg, planteado en el Cuadro 3, es válido para el manipulador serial R17 y permite su simulación en software. Esto se demostró de manera cualitativa y cuantitativa, siendo el primer caso las pruebas presentadas en las Figuras 34 - 36; y de manera cuantitativa con las pruebas de cinemática directa e inversa. Para el caso de las pruebas de cinemática directa se obtuvo errores absolutos de posición menores a 20.66 mm y en el caso de cinemática inversa porcentajes de error no mayores al 4%.
- El modo segmentado para correr trayectorias en el R17 tiene una mayor cercanía a las posiciones definidas en comparación con el modo continuo, sin embargo presenta cambios abruptos en la velocidad lineal del efector final, aparte que se ve obligado a detenerse (llevar su velocidad a 0 mm/s) en cada una de estas posiciones.
- El modo continuo para correr trayectorias en el R17 tiene un mayor error de posición en comparación con el modo segmentado, sin embargo posee mayor continuidad en la velocidad lineal del efector final y puede desempeñar una trayectoria en menor tiempo.
- Se desarrolló e implementó una interfaz de usuario que permite la planificación y visualización de trayectorias en el manipulador serial R17, además de su uso con cinemática directa e inversa. Esta interfaz a su vez es capaz de comunicarse con el robot mediante a su lenguaje de operación ROBOFORTH II.
- Se diseñó e implementó un módulo de comunicación inalámbrica que permite la integración del manipulador R17 al ecosistema Robotat. Este es un sistema embebido que cuenta con una placa electrónica y un estuche de protección diseñado y fabricado a la medida. El PCB cuenta con un microcontrolador ESP32, un par de baterías Li-ion 18650 y un circuito encargado de convertir los niveles de las líneas de un puerto serie RS232 a niveles TTL y viceversa.

- El uso del *Robotat - R17 Module* permite la integración exitosa del manipulador R17 con el ecosistema Robotat. Esto se ve demostrado en los errores relativamente bajos del Cuadro 19 para la prueba de cinemática inversa y en las gráficas de las Figuras 56 y 57.

- Los modos de cinemática inversa y trayectorias no contemplan la influencia que posee la junta prismática en el movimiento del robot, ya que el riel es un accesorio adicional del mismo. Se demostró que las funciones desarrolladas en MATLAB para la cinemática directa e inversa, basadas en la implementación numérica de los algoritmos presentados en la expresiones (24) y (35), respectivamente, son válidos para el R17. Por tal razón, se recomienda utilizar estos algoritmos en conjunto con el modo JOINT, como base para controlar al manipulador serial en lugar de las funciones de cinemática inversa y trayectorias nativas del robot. Con esto se podría adquirir la versatilidad que ofrece la junta prismática.
- El modelo actual del sistema embebido, Robotat - R17 Module, posee baterías recargables, sin embargo estas deben de ser extraídas de la carcasa para poderse cargar de nuevo. Se recomienda incorporar un módulo de carga dentro del sistema embebido, de tal manera que sea más simple y cómodo energizarlas de nuevo.
- Se recomienda agregar terminales de prueba en puntos estratégicos del Robotat - R17 Module. Por ejemplo, en la salida del puerto serial del ESP32, para poder analizar las señales antes de pasar por el circuito del MAX232; antes y después del regulador, para poder identificar averías en las baterías o el regulador; finalmente, después del MAX232, para identificar si el circuito está efectuando su tarea correctamente. Añadir estos puntos de prueba facilitarán a que usuarios puedan diagnosticar problemas dentro del módulo.

-
- [1] B. Karlik y S. Aydin, “An improved approach to the solution of inverse kinematics problems for robot manipulators,” *Engineering applications of artificial intelligence*, vol. 13, n.º 2, págs. 159-164, 2000.
 - [2] J. J. V. González, C. A. P. Caro, H. V. González y col., “Cinemática inversa de robot serial utilizando algoritmo genético basado en MCDS,” *Tecnura*, vol. 19, n.º 44, págs. 33-46, 2015.
 - [3] A. Godil, R. Bostelman, K. Saidi, W. Shackleford, G. Cheok, M. Shneier y T. Hong, “3D ground-truth systems for object/human recognition and tracking,” en *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2013, págs. 719-726.
 - [4] *Fast Robot Arm Catches Flying Objects - IEEE Spectrum*, <https://spectrum.ieee.org/automaton/robotics/robotics-hardware/epfl-fast-robot-arm-catches-flying-objects>, (Accessed on 04/10/2021).
 - [5] I. Martínez, E. Aristondo, D. Castellanos, E. Chang y E. Zea, “Megaproyecto GC-R17: Sistema háptico de control a distancia,” Universidad del Valle de Guatemala, oct. de 2009.
 - [6] *Robotarium - A Robotics Lab Accessible to All | The George W. Woodruff School of Mechanical Engineering*, https://www.me.gatech.edu/featured_Robotarium_Robotics_Lab_For_All, (Accessed on 04/10/2021).
 - [7] J. Castañeda, “Diseño e implementación de una red de comunicación wifi e interfaz gráfica para una mesa de pruebas de robótica de enjambre,” Universidad del Valle de Guatemala, oct. de 2020.
 - [8] R. Schumann, “Diseño de gripper para robot serial R17 controlado de manera inalámbrica,” Universidad del Valle de Guatemala, ene. de 2021.
 - [9] Vicon, *What is Motion Capture? | What Can I Use Motion Capture For?* <https://www.vicon.com/about-us/what-is-motion-capture/>, (Accessed on 03/25/2021).
 - [10] P. Nogueira, “Motion capture fundamentals,” en *DOCTORAL SYMPOSIUM IN INFORMATICS ENGINEERING*, 2011, pág. 303.

- [11] K. Kurihara, S. Hoshino, K. Yamane e Y. Nakamura, “Optical motion capture system with pan-tilt camera tracking and real time data processing,” en *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, IEEE, vol. 2, 2002, págs. 1241-1248.
- [12] A. J. Davison, J. Deutscher e I. D. Reid, “Markerless motion capture of complex full-body movement for character animation,” en *Computer Animation and Simulation 2001*, Springer, 2001, págs. 3-14.
- [13] *OptiTrack - About OptiTrack*, <https://optitrack.com/about/>, (Accessed on 04/01/2021).
- [14] *OptiTrack - Hardware*, <https://optitrack.com/cameras/>, (Accessed on 04/02/2021).
- [15] *OptiTrack - Prime 41 - In Depth*, <https://optitrack.com/cameras/primex-41/>, (Accessed on 04/01/2021).
- [16] *OptiTrack - Motive - Optical motion capture software*, <https://optitrack.com/software/motive/>, (Accessed on 04/01/2021).
- [17] K. M. Lynch y F. C. Park, *Modern Robotics*. Cambridge University Press, 2017.
- [18] J. M. Chaverria, “Desarrollo de un manipulador didáctico con una cadena cinemática abierta de 6 grados de libertad,” Tesis doct., Universidad Tecnológica de Pereira. Facultad de Ingeniería Mecánica . . . , 2016.
- [19] B. Siciliano, L. Sciavicco, L. Villani y G. Oriolo, *Robotics: modelling, planning and control*. Springer Science & Business Media, 2010.
- [20] M. Zea, *Lección 1: Cinemática de cuerpos rígidos en 2D*, feb. de 2021.
- [21] P. Corke, *Robotics, vision and control: fundamental algorithms in MATLAB® second, completely revised*. Springer, 2017, vol. 118.
- [22] E. G. Hemingway y O. M. O’Reilly, “Perspectives on Euler angle singularities, gimbal lock, and the orthogonality of applied forces and applied moments,” *Multibody System Dynamics*, vol. 44, n.º 1, págs. 31-56, 2018.
- [23] S. R. Buss, “Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods,” *IEEE Journal of Robotics and Automation*, vol. 17, n.º 1-19, pág. 16, 2004.
- [24] P. Courrieu, “Fast computation of Moore-Penrose inverse matrices,” *arXiv preprint arXiv:0804.4809*, 2008.
- [25] S. Robotics, *R17 low cost bench top articulated robot arm*, <https://strobotics.com/articulated-robot-arm.htm>, (Accessed on 04/05/2021).
- [26] D. N. Sands, *R17 Robot Manual*, 3.ª ed., ST Robotics, 103 Carnegie Center, Princeton NJ, 08540, United States, abr. de 2017.
- [27] S. Robotics, *ST robotics software suite*, <https://www.strobotics.com/roboforth.htm>, (Accessed on 04/05/2021).
- [28] J. P. Blázquez, “Introducción a los sistemas de comunicación inalámbricos,” *FUOC. Fundació per a la Universitat Oberta de Catalunya*, 2015.
- [29] J. J. A. Horno, “Redes de área local inalámbricas: Diseño de la wlan de wheelers lane technology college,” *Universidad de Sevilla*, 2008.
- [30] *MQTT - The Standard for IoT Messaging*, <https://mqtt.org/>, (Accessed on 04/18/2021).

- [31] HiveMQ, *MQTT & MQTT 5 Essentials: A comprehensive overview of MQTT facts and features for beginners and experts alike*, HiveMQ GmbH Ergoldinger Str. 2A 84030 Landshut Germany. [Online]. dirección: shorturl.at/eo0Q8.
- [32] C. Perafán, “Robotat: Un Ecosistema Robótico de Captura de Movimiento y Comunicación Inalámbrica,” Universidad del Valle de Guatemala, ene. de 2021.
- [33] *Robotics Toolbox* / Peter Corke, <https://petercorke.com/toolboxes/robotics-toolbox/>, (Accessed on 08/17/2021).
- [34] D. N. Sands, *Robotics Self-learn Tutorial – 6-Axis Robots R12,R17*, 1.^a ed., ST Robotics, 103 Carnegie Center, Princeton NJ, 08540, United States, dic. de 2012.
- [35] *Tracker Video Analysis and Modeling Tool for Physics Education*, <https://physlets.org/tracker/>, (Accessed on 09/21/2021).
- [36] *ESP32 Wi-Fi & Bluetooth MCU I Espressif Systems*, <https://www.espressif.com/en/products/socs/esp32>, (Accessed on 09/21/2021).
- [37] *MAX232 Dual RS-232 Driver and Receiver With IEC61000-4-2 Protection*, SLLS723C, Rev. August 2016, Texas Instruments, abr. de 2006.
- [38] *18650 Battery Specifications: Datasheet and Properties - SM Tech*, <https://somanotech.com/18650-battery-specifications-datasheet-18650-battery-specs/>, (Accessed on 09/21/2021).
- [39] *L7800 Series*, Rev. 12, ST, nov. de 2004.
- [40] *Software y herramientas de diseño de PCB para construir la próxima generación en electrónica* / Altium, <https://www.altium.com/es>, (Accessed on 09/21/2021).
- [41] *Software Inventor* / Consulta los precios y compra Inventor 2022 oficial, <https://latinoamerica.autodesk.com/products/inventor/overview>, (Accessed on 09/21/2021).
- [42] IPC-2221 Task Group (D-31b), “IPC-221A,” en, IPC, 2215 Sanders Road, Northbrook, iL 60062-6135, Standard ISBN #1-580982-68-9, 2003. dirección: www.ipc.org.
- [43] *Fundamentals of RS-232 Serial Communications*, <https://www.maximintegrated.com/en/design/technical-documents/tutorials/8/83.html>, (Accessed on 09/13/2021).

14.1. Pruebas de cinemática directa

```
>WHERE
R17:
      X      Y      Z      PITCH  W (ROLL)  LEN.  OBJECT
R17:  0.0    0.0   750.0   -90.0    0.0     0.0
R17:
PREV  0.0    0.0   750.0   -90.0    0.0     0.0
R17:
OK
```

Figura 58: Posición del efector final dada por el comando WHERE del R17 para una configuración q_1 del Cuadro 4.

```
>WHERE
R17:
      X      Y      Z      PITCH  W (ROLL)  LEN.  OBJECT
R17:  0.0   477.5  578.1   -50.4    0.0     0.0
R17:
PREV  0.0    0.0   750.0   -90.0    0.0     0.0
R17:
OK
```

Figura 59: Posición del efector final dada por el comando WHERE del R17 para una configuración q_2 del Cuadro 4.

```

>WHERE
R17:
      X      Y      Z  PITCH  W (ROLL)  LEN.  OBJECT
R17:
    235.3   -3.6   672.0  -88.9    0.0    0.0
R17:
PREV    0.0    0.0   750.0  -90.0    0.0    0.0
R17:
OK

```

Figura 60: Posición del efector final dada por el comando WHERE del R17 para una configuración q_3 del Cuadro 4.

```

>WHERE
R17:
      X      Y      Z  PITCH  W (ROLL)  LEN.  OBJECT
R17:
    612.6   130.5   404.2  -51.0   -90.0    0.0
R17:
PREV    235.3   -3.6   672.0  -88.9    0.0    0.0
R17:
OK

```

Figura 61: Posición del efector final dada por el comando WHERE del R17 para una configuración q_4 del Cuadro 4.

```

>WHERE
R17:
      X      Y      Z  PITCH  W (ROLL)  LEN.  OBJECT
R17:
   -521.6  -101.0   348.0  -123.2    23.2    0.0
R17:
PREV    0.0    0.0   750.0  -90.0    0.0    0.0
R17:
OK

```

Figura 62: Posición del efector final dada por el comando WHERE del R17 para una configuración q_5 del Cuadro 4.

14.2. Pruebas de cinemática inversa

```

>WHERE
R17:
      X      Y      Z  PITCH  W (ROLL)  LEN.  OBJECT
R17:
    0.0    0.0   750.0  -90.0    0.0    0.0
R17:
PREV    0.0    0.0   750.0  -90.0    0.0    0.0
R17:
OK

```

Figura 63: Posición del efector final dada por el comando WHERE del R17 para una configuración correspondiente a el vector de posición p_1 del Cuadro 9.


```

>WHERE
R17:
      X      Y      Z  PITCH  W(ROLL)  LEN.  OBJECT
R17:
      0.0  498.6  541.5  -36.3    0.0    0.0
R17:
PREV  0.0    0.0  750.0  -90.0    0.0    0.0
R17:
OK

```

Figura 64: Posición del efector final dada por el comando WHERE del R17 para una configuración correspondiente a el vector de posición p_2 del Cuadro 9.

```

>WHERE
R17:
      X      Y      Z  PITCH  W(ROLL)  LEN.  OBJECT
R17:
      0.0  485.1  555.2  -38.3    0.0    0.0
R17:
PREV  0.0  489.1  550.0  -37.3    0.0    0.0
R17:
OK

```

Figura 65: Posición del efector final dada por el comando WHERE del R17 para una configuración correspondiente a el vector de posición p_3 del Cuadro 9.

```

>WHERE
R17:
      X      Y      Z  PITCH  W(ROLL)  LEN.  OBJECT
R17:
    104.0  479.4  551.9  -38.3    0.0    0.0
R17:
PREV  0.0  498.6  541.5  -36.3    0.0    0.0
R17:
OK

```

Figura 66: Posición del efector final dada por el comando WHERE del R17 para una configuración correspondiente a el vector de posición p_4 del Cuadro 9.

```

>WHERE
R17:
      X      Y      Z  PITCH  W(ROLL)  LEN.  OBJECT
R17:
   -149.6  588.0  401.4  -19.4    0.0    0.0
R17:
PREV  104.0  479.4  551.9  -38.3    0.0    0.0
R17:
OK

```

Figura 67: Posición del efector final dada por el comando WHERE del R17 para una configuración correspondiente a el vector de posición p_5 del Cuadro 9.

14.3. Diseño y fabricación de sistema embebido

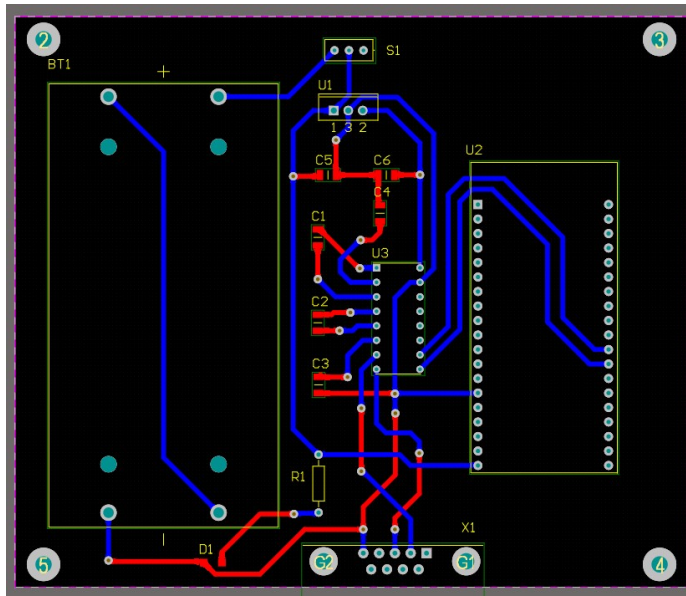


Figura 68: Diseño electrónico de PCB del sistema embebido Robotat - R17 Module.

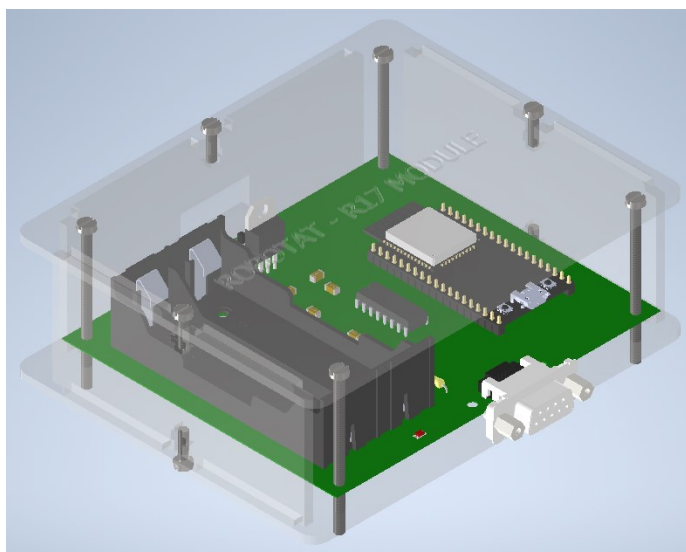


Figura 69: Modelo 3D de Robotat - R17 Module.

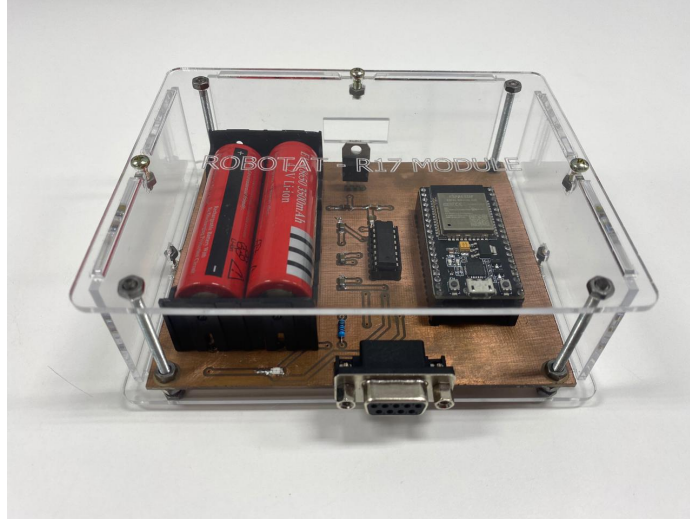


Figura 70: Sistema embebido Robotat - R17 Module.

14.4. Repositorio de trabajo

Toda la documentación adicional de este trabajo de graduación se encuentra subida en el *branch 17003-Dev* del siguiente repositorio de GitHub.

Repositorio: <https://github.com/mezea-uvg/robotat.git>

14.5. Videos explicativos y demostrativos

Videos del uso de la interfaz en MATLAB y el sistema embebido Robotat - R17 Module se presentan en el siguiente enlace.

Carpeta en Drive: <https://bit.ly/3y3FdkD>