CONTROL POR GUIADO GESTUAL PARA UN ROBOT ANTROPOMÓRFICO BASADO EN TÉCNICAS DE VISIÓN ARTIFICIAL

ANDRÉS FELIPE PADILLA MANTILLA

MAESTRÍA EN CONTROLES INDUSTRIALES

FACULTAD DE INGENIERÍAS

UNIVERSIDAD DE PAMPLONA

PAMPLONA

2021

CONTROL POR GUIADO GESTUAL PARA UN ROBOT ANTROPOMÓRFICO BASADO EN TÉCNICAS DE VISIÓN ARTIFICIAL

ANDRÉS FELIPE PADILLA MANTILLA

Trabajo de grado prestado como requisito para optar el título de

Magister en controles industriales

Director:

Ph.D. César Augusto Peña Cortés

MAESTRÍA EN CONTROLES INDUSTRIALES

FACULTAD DE INGENIERÍAS

UNIVERSIDAD DE PAMPLONA

PAMPLONA

2021

TABLA DE CONTENIDO

LISTA DE TABLAS

LISTA DE FIGURAS

LISTA DE ANEXOS

RESUMEN

En este proyecto se desarrolló un control por guiado gestual para el robot antropomórfico Raplim de la Universidad de Pamplona *UP* a partir de la captura de los movimientos de un operador a través de una cámara de luz estructurada, permitiendo la reconstrucción tridimensional del movimiento para programar trayectorias teniendo en cuenta la cinemática del robot.

A partir de los comandos gestuales se inicia y se finaliza la captura de los movimientos utilizando técnicas de visión artificial, estos movimientos se transforman en trayectorias aplicando el control cinemático y estas trayectorias pueden ser simuladas en el espacio de trabajo del robot aplicando la realidad aumentada. Para la implementación del proyecto se desarrollaron algoritmos de optimización que permiten identificar la intensión de la trayectoria que desea realizar el usuario.

La metodología de investigación aplicada es de tipo experimental donde se pusieron a prueba algunas formas en las que se basan las trayectorias que fueron simuladas en el ambiente real a través de la realidad aumentada con un robot virtual dando validez al algoritmo desarrollado evidenciando que dichas trayectorias son semejantes a los movimientos capturados. Los resultados de las pruebas para cada modelo tienen una mejor representación en el plano XY debido a que la captura que hace la cámara de luz estructurada presenta variación en el eje Z en la identificación del punto de la articulación de interés con la que se realiza el recorrido para generar la trayectoria. De esta manera como la trayectoria debe generarse en el espacio tridimensional se desarrollaron filtros para reducir el efecto de las variaciones en el eje Z de la captura.

Definido lo anterior, se evidencia que programar trayectorias para un robot a partir de los gestos o movimientos de un operador puede ser una tarea muy compleja que se vio simplificada al aplicar los algoritmos de optimización y filtros reduciendo el error que presenta la captura con respecto al modelo de la muestra.

ABSTRACT

In this project a gestural guidance control was developed for the anthropomorphic robot Raplim of the University of Pamplona *UP* from the capture of the movements of an operator through a structured light camera, allowing the three-dimensional reconstruction of the movement to program trajectories taking into account the kinematics of the robot.

From the gestural commands, the capture of movements is initiated and completed using artificial vision techniques, these movements are transformed into trajectories by applying kinematic control and these trajectories can be simulated in the robot's workspace by applying augmented reality. For the implementation of the project, optimization algorithms were developed to identify the intention of the trajectory that the user wants to perform.

The applied research methodology is of experimental type where some forms were tested based on the trajectories that were simulated in the real environment through augmented reality with a virtual robot giving validity to the developed algorithm showing that these trajectories are similar to the captured movements. The results of the tests for each model have a better representation in the XY plane due to the fact that the capture made by the structured light camera presents variation in the Z axis in the identification of the point of the joint of interest with which the path is made to generate the trajectory. Thus, since the trajectory must be generated in three-dimensional space, filters were developed to reduce the effect of the variations in the Z axis of the capture.

Having defined the above, it is evident that programming trajectories for a robot from the gestures or movements of an operator can be a very complex task that was simplified by

applying optimization algorithms and filters to reduce the error presented by the capture with respect to the sample model.

1. INTRODUCCIÓN

La forma de programar robots industriales está cambiando con el avance de la tecnología y la aplicación de la ciencia moderna en este campo de la investigación, por eso hoy se puede hablar de realidad aumentada en el campo de la enseñanza, los videojuegos, el marketing, el entretenimiento, entre otros, este proyecto busca aplicar esta nueva ciencia así como la visión artificial a la programación y simulación de robots industriales construyendo algoritmos de optimización que evalúen y generen una trayectoria satisfactoria para cargar en el robot real.

El algoritmo de control cinemático para el robot antropomórfico *Raplim* de la *Universidad de Pamplona* transforma los movimientos de un operador capturados a través de una cámara de luz estructurada (*Kinect*) en comandos gestuales que luego son transformados en trayectorias que se simulan utilizando la visión artificial antes de implementarlas en el robot real. Para todos los algoritmos se desarrolló una interfaz de usuario que facilite la manipulación e interpretación de los datos procesados, simulándolos en el ambiente virtual del robot antes de cargarle la programación.

Este libro se divide en varios capítulos que estructuran la investigación y desarrollo del proyecto, en el capítulo 1 después de abordar la introducción se presenta el título del proyecto, los objetivos, el planteamiento del problema y sus beneficios. En el capítulo 2 se desarrolló el marco teórico donde se dan a conocer de forma breve todos los conceptos

necesarios para interpretar el desarrollo del proyecto. En el capítulo 3 se da a conocer el modelo cinemático del robot. En el capítulo 4 se describe el procedimiento para aplicar la programación por guiado gestual, los sistemas coordenados y se definen los comandos que se utilizaron. En el capítulo 5 se muestran las interfaces de usuario que fueron creadas para interactuar con el operador a medida que genera la trayectoria. En el capítulo 6 se desarrolla la aplicación de realidad aumentada como herramienta que le permite al usuario observar el comportamiento del robot virtual en un ambiente real. El capítulo 7 da a conocer los resultados que se obtienen al aplicar los algoritmos de optimización a los movimientos capturados. En el capítulo 8 se dan a conocer las conclusiones y recomendaciones del desarrollo de este proyecto. Finalmente se concluye con las publicaciones y ponencias producto de este trabajo de investigación, la lista detallada de las referencias bibliográficas consultadas y los anexos.

1.1 Título

Control por guiado gestual para un robot antropomórfico basado en técnicas de visión artificial.

1.2 Objetivos

1.2.1 Objetivo general

Desarrollar un algoritmo de control cinemático por medio de comandos gestuales para un robot antropomórfico basado en técnicas de visión artificial.

1.2.2 Objetivos específicos

- ✓ Generar el algoritmo de reconstrucción tridimensional del cuerpo humano identificando los puntos de referencia para realizar el control cinemático.
- ✓ Generar un algoritmo de control que le permita a un usuario gestionar una trayectoria tridimensional en el espacio de la tarea a partir de las referencias generadas con su propio cuerpo.
- ✓ Elaborar un sistema supervisorio que permita validar la trayectoria planteada, corregirla o desecharla teniendo en cuenta consideraciones como: el espacio de trabajo del robot, las posibles colisiones de los elementos de su estructura, entre otros.
- ✓ Desarrollo de una interfaz hombre-máquina que le permita al usuario interactuar con los parámetros del algoritmo de control previamente realizado.

1.3 Planteamiento del problema y justificación

Actualmente la mayoría de las grandes empresas acuden a los fabricantes de tecnología para el suministro de máquinas inteligente, capaces de desarrollar trabajos específicos en las diferentes áreas de proceso según sus necesidades. Algunas de estas empresas son *ABB* y *KUKA*, las cuales cuentan con reconocimiento mundial en la fabricación de robots industriales con resultados óptimos, pero a costos elevados. Cada empresa se encarga de gestionar un robot según las necesidades de su industria, por ende, se deben especificar los métodos de programación que permiten ajustar las tareas que deben ejecutar según sus características mecánicas.

Entre los tipos de lenguaje de programación de robots industriales se encuentra el off-line y el on-line. El primero no necesita interactuar directamente con el robot, es decir, se debe desarrollar un código o escribir una serie de comandos en los que se especifiquen las tareas a realizar y los pasos a seguir, basándose en un lenguaje de programación específico para controlar el robot. En el segundo no se necesita saber un lenguaje específico de programación para controlar el robot, basta con guiarlo de forma manual y enseñarle los movimientos necesarios para desarrollar la tarea como método de programación.

Ahora bien, la programación on-line es la más conveniente a la hora de programar robots de tipo brazo robótico y las empresas buscan ejemplares que puedan ser programados de esta manera, dejando de lado la programación por comandos o textual en la que el operario debe tener conocimiento de un tipo de lenguaje de programación específico para modificar la velocidad, las trayectorias y los movimientos del robot. Actualmente las industrias se han

especializado en fabricar robots con estas características, pero con costos muy elevados, lo que hace que las empresas de mediano y pequeño capital no consideren invertir en esta tecnología si ya cuentan con una máquina que lleva a cabo las tareas de producción, complejidad de modificar el código cada vez que se deba de mejorar, ajustar o cambiar la actividad industrial de la empresa.

Teniendo en cuenta lo anterior se plantea una solución para mejorar el sistema de programación de los robots tradicionales utilizando dispositivos auxiliares que sirvan como intérpretes para programar de forma guiada las trayectorias que deba realizar el robot. Este proyecto busca responder a la pregunta; ¿es posible mejorar el método de programación de un robot antropomórfico utilizando algoritmos para definir su trayectoria mediante visión artificial con el fin de hacerla más intuitiva?

1.4 Beneficios

El avance de la tecnología proporciona ventajas en todas las áreas del conocimiento que se deben aprovechar para brindar oportunidades de modernización en el sector industrial e institucional. Este proyecto combina algunas tecnologías como la visión artificial, la realidad aumentada y la robótica en su desarrollo, las cuales hacen parte de la industria 4.0 con las que se pudieron obtener los siguientes beneficios tanto en sector tecnológico como en el sector institucional:

1.4.1 Beneficios Tecnológicos.

- Incorpora el uso de la visión artificial en la interpretación de los movimientos de un usuario para generar de trayectorias óptimas que se puedan programar en un robot industrial.
- Desarrolla una plataforma en donde el usuario puede simular con realidad aumentada el comportamiento de la trayectoria previo a cargar la programación al robot real para observar con mayor facilidad el comportamiento que tiene el robot al instante en el que el usuario realiza los movimientos.
- Los dispositivos para hacer uso de la realidad aumentada no tienen que ser los más avanzados del mercado, basta con descargar la aplicación en un celular o table y utilizar un marcador para observar el robot en realidad aumentada a través del equipo.

1.4.2 Beneficios Institucionales.

- Desarrollar trabajos de investigación que incorporen las nuevas tecnologías de la industria 4.0 refuerza el reconocimiento académico a nivel nacional de la institución.
- Aporta conocimientos actuales para desarrollar laboratorios de prueba, donde se pongan en práctica las interfaces de usuario desarrolladas para llevar a cabo prácticas de laboratorio para la generación de trayectorias del robot Raplim de la UP.
- Motiva a la adquisición de mejores equipos tecnológicos con los que se mejoren y facilite la generación de trayectorias utilizando la visión artificial.
- El uso de los algoritmos de optimización que se usan en el desarrollo del proyecto sirve como plantilla para futuros trabajos donde se aplique la realidad aumentada.

2. MARCO TEÓRICO

En el siglo *XX* la revolución tecnológica se desarrolló a un ritmo acelerado y los avances en la electrónica, las telecomunicaciones, la automatización y la computación fueron los principales protagonistas, gracias a estos avances hoy se pueden hablar de nanotecnología y robótica. La revolución del siglo *XXI* estará dada por los avances en la robótica y sus aplicaciones en las diferentes áreas de la ciencia combinando los sistemas mecánicos, eléctricos, la realidad aumentada y la inteligencia artificial. Unificando las múltiples tecnologías existentes y las nuevas que se están desarrollando se tiene la oportunidad de modificar la percepción de lo que debería ser un *robot* y su clasificación (Rossiter, 2016).

2.1 Definición y clasificación de robots

La definición de robot industrial tiene 2 contrapartes, la primera es el concepto que le da el mercado japonés donde un robot industrial es cualquier dispositivo mecánico dotado de articulaciones móviles destinado a la manipulación. Y la segunda es el concepto del mercado occidente que lo define como: “Manipulador multifuncional reprogramable con varios grados de libertad, capaz de manipular materiales, piezas, herramientas o dispositivos espaciales según trayectorias variables programadas para realizar tareas diversas”, según la Organización Internacional de Estándares (Norma *ISO*) (Barrientos, Peñin, Balaguer, & Aracil, 2007).

Michael Knasel, director del centro de Aplicaciones Robóticas de *Science Application Inc.*, expone la clasificación de los robots en cinco generaciones (Knasel, 1986):

- Primera generación – "*pick-and-place*": *Robots* que operan desde una base fija como auxiliares de máquinas industriales moviendo objetos de un lugar a otro.

- Segunda generación – Servocontrol: *Robots* que operan desde una base móvil dotados de programas con trayectorias continuas, empleados normalmente en soldaduras y pintura.

- Tercera generación – *AGVs*: *Robots* que operan automáticamente de forma auto-guiada incluyendo la visión artificial, se emplean fundamentalmente en funciones de acoplamiento y montaje.

- Cuarta generación – *Robots* móviles: *Robots* con ruedas o piernas articulares para movilizarse en exteriores e interiores dotados de sensores inteligentes y utilizados principalmente en construcción y procesos de mantenimiento.

- Quinta generación – basado en inteligencia artificial: *Robots* que operan con controladores basados en inteligencia artificial, adaptados a diferentes bases móviles según las necesidades de su aplicación.

Esta clasificación permite proyectar los avances futuros dentro de la quinta generación mejorando la movilidad, la autonomía e inteligencia para llevar a cabo tareas y tomar decisiones, incorporando así las múltiples tecnologías existentes y las a desarrollar en el campo de la ciencia (Hernández Herrera, 2012).

2.2 Elementos básicos de un robot industrial

Las características estructurales de la morfología de un robot industrial están dadas por los siguientes elementos: estructura mecánica, transmisiones, sistemas de accionamiento, sistemas sensoriales, sistema de control y elementos terminales.

2.2.1 Estructura mecánica.

Un robot industrial está formado por una serie de elementos estructurales rígidos llamados enlaces o eslabones, conectados mediante juntas o articulaciones que permiten movimientos parciales cada 2 eslabones consecutivos. El conjunto de eslabones y articulaciones se denomina cadena cinemática abierta, donde cada eslabón se conecta únicamente con la articulación anterior y siguiente, exceptuando el primero (base) que normalmente se encuentra fijo a un soporte y el ultimo que se encuentra libre donde se conecta un elemento terminal o actuador final (Ollero Baturone, 2001).

Una articulación puede ser:

- Lineal (deslizante, traslacional o prismática), cuando un eslabón se desliza sobre un eje solidario al eslabón anterior.
- Rotacional, Cuando un eslabón gira en torno a un eje solidario al eslabón anterior.

La construcción física de la mayoría de los manipuladores robóticos industriales posee cierta similitud al brazo humano por lo que en ocasiones para referirse a los distintos

elementos que componen el robot se usan términos como cuerpo, brazo, codo y muñeca (Loaiza Bernal, 2012).

El movimiento del eslabón-articulación puede ser de desplazamiento, de giro o de ambos, conocidos como grados de libertad (*GDL*), una articulación puede tener 2 o más *GDL*, en la práctica los robots emplean movimientos prismáticos y de rotación. Los grados de libertad (*GDL*) de un *robot* describen el estado del sistema mecánico del *robot* (posición y orientación) en el espacio (Chang Herrera, 2014).

Los diferentes tipos de articulación que se usan para los robots industriales y sus *GDL* se ilustran en la Figura 1.

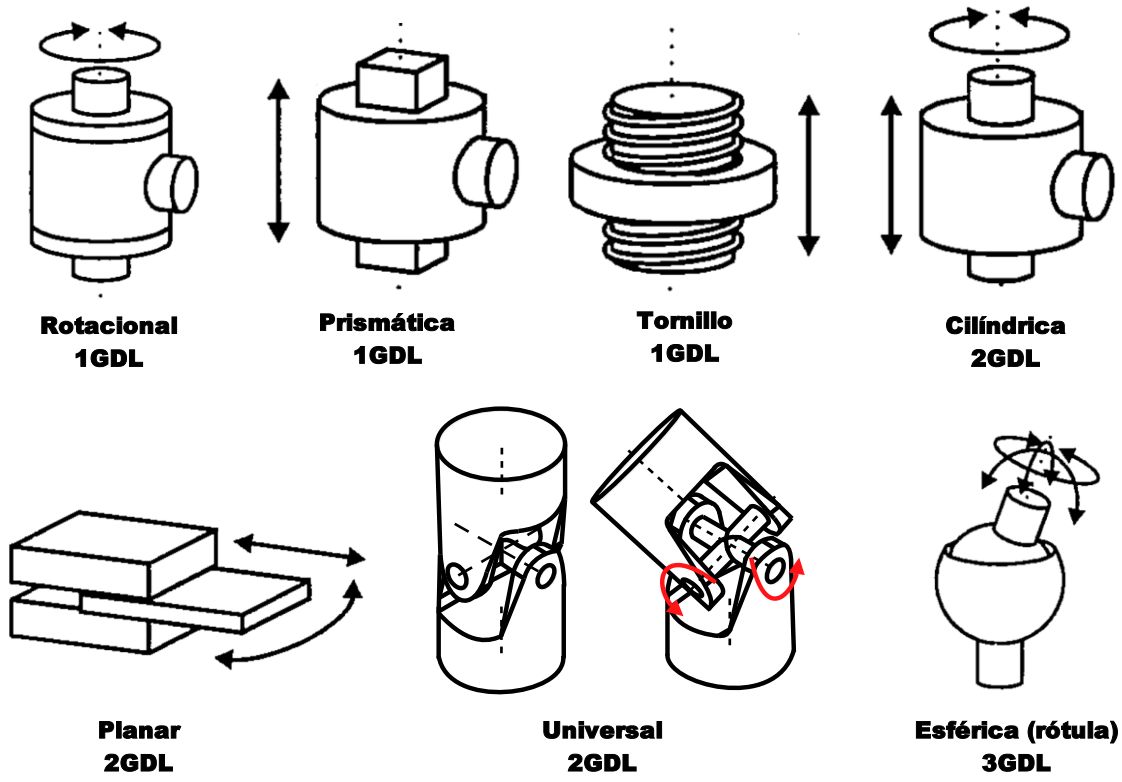
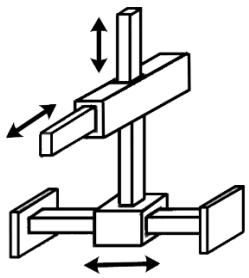


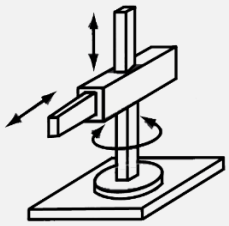
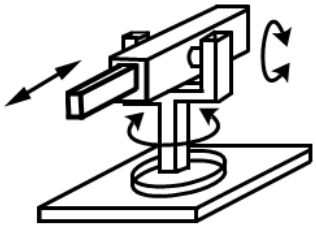
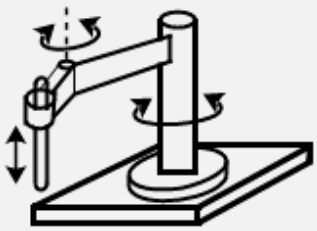
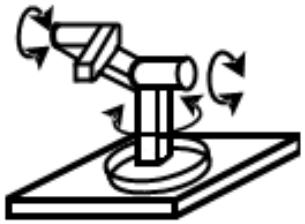
Figura 1. Articulación para robot y sus grados de libertad (Barrientos, Peñin, Balaguer, & Aracil, 2007; Bayani, Rastegari, & Cheraghpour Samavati, 2015).

Los grados de libertad (*GDL*) de un robot industrial están dados por la suma de los *GDL* de las articulaciones que los componen, es decir, que dependiendo del tipo de articulación que utilice el robot dependerán sus grados de libertad.

Para posicionar y orientar el extremo libre (actuador final) del robot industrial en el entorno de trabajo, se necesitan 6 parámetros, 3 para definir la posición y 3 para definir la orientación, es decir, al menos 6 grados de libertad. Pero en la realidad, aunque son necesario los 6 *GDL* para lograr obtener la posición y orientación del extremo final del robot, muchos solo alcanzan a tener 4 o 5 *GDL* logrando llevar a cabo las tareas para lo que fueron diseñados. Las configuraciones más usadas en el diseño estructural de los robots industriales se clasifican en la Tabla 1 (Barrientos, Peñin, Balaguer, & Aracil, 2007).

Tabla 1.
Configuraciones de las estructuras de los robots industriales.

NOMBRE DE LA CONFIGURACIÓN	CONFIGURACIÓN	CARACTERÍSTICAS	USO
Robot cartesiano		Su desplazamiento lo hace en las 3 coordenadas cartesianas x, y, z. Son rápidos, muy precisos, fácil control, amplia zona de trabajo y elevada capacidad de carga. Se usan en aplicaciones que requieren movimientos lineales de alta precisión en planos paralelos. Ocupan mucho espacio y su efector final no se puede orientar.	En este tipo de estructura robótica se basa el diseño de la *CNC* y de las impresoras 3D.

Robot cilíndrico		El robot cilíndrico trabaja con coordenadas cilíndricas, se controla fácilmente y es rápido, pero solo se usa en casos en los que no haya obstáculos en su zona de trabajo con acceso a ella de forma horizontal.	Se usaban mucho cuando la estructura de los robot eran neumáticos y/o hidráulicos, pero empiezan a quedarse obsoletos y a ser reemplazados por los robot´s con accionamientos electrónicos.
Robot esférico o polar		Los robots esféricos están dotados normalmente de accionamientos hidráulicos y/o neumáticos, se utilizan para tareas de descarga de máquinas, palatización de piezas y soldadura por puntos.	Están prácticamente fuera de uso
Robot SCARA		*SCARA (*Selective Compliant Assembly Robot Arm*)*, es un robot articulado de 4 *GDL* con posicionamiento horizontal. Es un robot barato, rápido y preciso con gran capacidad de carga, pero solo puede trabajar en planos perpendiculares.	Se usa en la industria para la selección, contabilización, levantamiento y ensamble de objetos. Se comporta como un brazo robótico.
Robot angular o antropomórfico		Se llama antropomórfico porque simula los movimientos de un brazo humano, este tipo de robot es muy rápido, con gran accesibilidad y maniobrabilidad y ocupa poco espacio en el área de trabajo.	Se usan en aplicaciones de soldadura y proceso de doblado, adicional a esto se usa en aplicaciones de palatización.

Nota: adaptado de "Fundamentos de Robótica", por A. Barrientos, L.F. Peñin, C. Balaguer y R. Aracil, pag 35, figura 2.4, Mc Graw Hill, Madrid, 2007.

2.2.2 Características y especificaciones de los robots industriales

Teniendo en cuenta las características y especificaciones técnicas de un robot industrial se puede identificar el tipo de robot y las tareas o trabajos aptos para realizar (León, 2012). Los fabricantes tienen la potestad de especificar las características del diseño de su *robot*, por lo que no todos los robots presentan las mismas, a continuación, se mencionan las más comunes:

- Grados de libertad: Los grados de libertad de un robot industriales están entre los 3 y 6 *GDL*, sin contar los movimientos que realiza la herramienta acoplada en el efector final. A más *GDL* mayor flexibilidad en el posicionamiento del efector final.

- Zona de trabajo: Es básicamente el volumen espacial que ocupa el robot, depende mucho del tipo de robot, la cantidad y el tamaño de sus articulaciones y los ejes en los que opere.

- Capacidad de carga: Se refiere al máximo peso que el robot puede transportar a su velocidad nominal, incluyendo el peso de la pieza o herramienta acoplada en el efector final (muñeca-mano), se debe garantizar el posicionamiento del robot cuando se le aplique la carga. Normalmente el peso varía entre 2 a 200 kg aproximadamente

- Resolución: Depende fundamentalmente de la unidad de control del robot para incrementar o variar el mínimo desplazamiento que puede realizar el elemento final.

- Precisión: Es el grado de ajuste del valor del punto medio de una secuencia de movimientos repetitivos para alcanzar el punto programado.

- Repetibilidad: Son los grados de exactitud en la repetición de movimientos, es decir, que al momento de repetir un movimiento tenga una baja desviación, aunque no alcance el valor del punto programado. Si el robot se programa por el método de aprendizaje, la repetibilidad es más importante que la resolución y la precisión, ya que lo importante en este caso es que el robot repita la posición de los puntos que ha grabado. La repetibilidad de los robots industriales varía desde 0.01 hasta 2mm y algunos fabricantes engloban precisión y repetibilidad en una solo característica denominada: precisión en la repetibilidad.

- Velocidad: La velocidad es muy importante en un robot industrial, sobre todo si el robot tiene que hacer movimientos muy largos y en aplicaciones de inserción, manipulación y montaje. La velocidad puede darse para cada articulación independientemente o bien solo para el movimiento de la muñeca que es la parte más importante para el usuario, la velocidad puede variar de 0.5 a 2 m/s según el robot y sus aplicaciones.

- Aceleración: La aceleración en un robot industriales es muy necesaria, más aún cuando los movimientos son cortos y se necesita de un buen arranque y un freno rápido. Algunos de los controles que se implementan presentan perfiles de movimiento con rampas de aceleración y frenado que optimizan los movimientos en función de la distancia, la velocidad y la inercia.

- Armario o unidad de control: Son una serie de características que integra la comunicación del robot con el entorno para controlar la capacidad y potencia del manipulador. Estas características están en constante evolución y entre ellas se encuentran las siguientes:
    - Memoria de almacenamiento y programa.
    - Lenguaje de programación.
    - Tipos de comunicaciones con el entorno; número y tipos de E/S.
    - Periféricos conectables, tipo de mando manual ***(teach pendant)*** con sus posibilidades de programación, movimientos manuales y otras funcionalidades de las que dispongan.
    - Posibilidades de ampliación y control de otros ejes externos al brazo robótico; parámetros de control para aplicaciones de pintado, soldadura, etc.

Estas características técnicas son algunas de las más importantes a tener en cuenta al momento de implementar un robot industrial, además de saber la tarea, trabajo o aplicación que va a llevar a cabo (Hernandez Matías & Vizán Idoipe, 2015).

2.3 Localización espacial

Para conocer la ubicación del efector final de un *robot* en su espacio de trabajo se deben especificar su posición y orientación en relación a un sistema de referencia definido

(Hernández Méndez, 2011). La representación de la posición del efector final con respecto a un sistema de referencia bidimensional o tridimensional se puede hacer utilizando cualquiera de las siguientes herramientas matemáticas:

- Sistema cartesiano de referencia.
- Coordenadas cartesianas.
- Coordenadas polares y cilíndricas.
- Coordenadas esféricas.

El espacio de trabajo del *robot* se representa en un plano tridimensional por lo que es necesario implementar 3 componentes *(X, Y, Z)* para la localización del efector final. Para describir la orientación del efector final respecto a su espacio de trabajo (celda de manufactura) se debe asignar en el plano del *robot* un nuevo sistema de referencia para estudiar la relación espacial existente entre ambos, donde la posición y orientación del nuevo sistema de prueba con respecto a la celda de manufactura será equivalente a la posición y orientación del efector final, partiendo de que ambos tienen el mismo origen (García Luna, Rodrígues Ramírez, & Luviano Cruz, 2019). A continuación, se mencionan los métodos matemáticos que sirven como herramienta para determinar la orientación del efector final:

- Matrices de rotación.
- Ángulos de Euler.

Para unificar la representación de la posición y orientación (localización) del efector final se recomienda trabajar con las denominadas coordenadas homogéneas.

2.4 Cinemática del robot

La cinemática es el estudio de los movimientos espaciales de un *robot* con respecto a un sistema de referencia en función del tiempo que relaciona la posición y orientación del efector final y el valor que tomen sus coordenadas articulares (Zaplana, Arnau Claret, & Basanez, 2018). Por otro lado, la cinemática del robot busca encontrar las relaciones entre las velocidades del movimiento de las articulaciones y las del extremo (efector final). Este modelo diferencial se expresa mediante la matriz Jacobiana.

Los dos problemas fundamentales a resolver en la cinemática del robot es la cinemática directa y la cinemática inversa (Fernandez Yanez & Sotomayor Reinoso, 2016). La Figura 2 ilustra la relación que existe entre ambas y las variables involucradas.

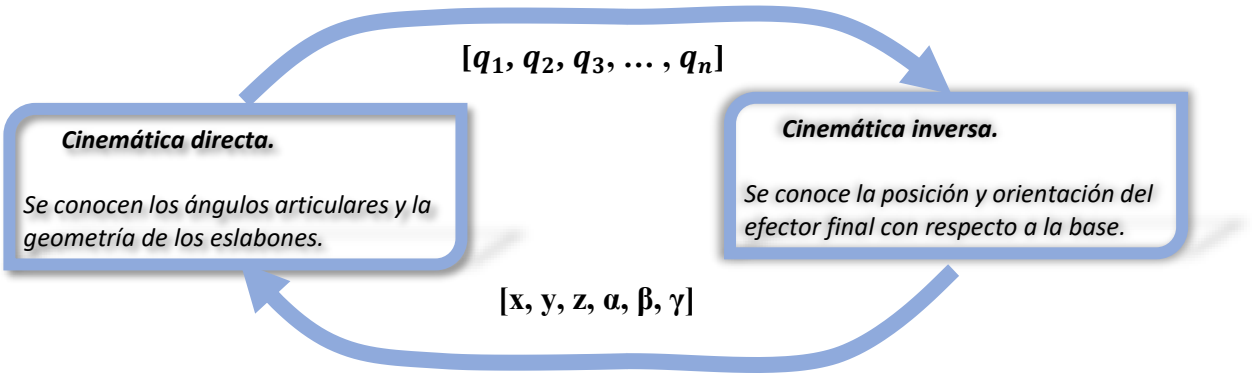


Figura 2. Diagrama de relación entre cinemática directa e inversa (Barrientos, Peñin, Balaguer, & Aracil, 2007).

A continuación, se define la cinemática directa y la cinemática inversa para tener sus conceptos y funcionamientos claros.

2.4.1 Cinemática directa

La cinemática directa da a conocer la localización del efector final para cualquier valor de posición y orientación que tomen las articulaciones a través de las variables dadas por los sensores del robot almacenadas en la unidad de control. La unidad de control necesita de la cinemática directa para mostrarle al usuario la información relativa de la posición y orientación del efector final (Ramires Arias & Rubiano Fonseca, 2012).

El modelo cinemático directo puede apoyarse en dos métodos matemáticos diferentes; el método geométrico aplicado cuando el caso a resolver es simple, limitándose a *robots* con pocos grados de libertar y los métodos basados en cambios de sistemas de referencia, utilizado para obtener el modelo cinemático de robots con *n* grados de libertad (Navarro, Robles, & Paulsen, 2007).

Son escasos los robots industriales que tienen pocos grados de libertad, la mayoría tienen de 4 grados en adelante por lo que es preciso mencionar un método con el que se pueda resolver el modelo cinemático de los robots industriales con n grados de libertad. El método que describe este capítulo se menciona a continuación.

3.4.1.1 Algoritmo de Denavit-Hartenberg

Denavit y Hartenberg proponen un modelo matricial con el que se localiza el sistema coordenado S_i perteneciente a cada eslabón ***i*** de una cadena articulada, donde para pasar de un eslabón a otro hay que hacer 4 transformaciones exclusivas dependiendo de las características de cada eslabón. Estas 4 transformaciones se apoyan en una sucesión de rotaciones y translaciones que relaciona el sistema de referencia del entorno $i-1$ con el

sistema del elemento i (Soto Duran, Peña, & Gualdron, 2013). A continuación, se mencionan las 4 transformaciones que se usan para pasar del sistema $\{S_{i-1}\}$ al $\{S_i\}$**.** Donde las transformaciones se refieren al sistema móvil.

1. Rotación alrededor al eje z_{i-1} un ángulo θ_i.
2. Translación a lo largo de z_{i-1} una distancia d_i; vector $d_i(0,0, d_i)$.
3. Translación a lo largo de x_i una distancia a_i; vector $a_i(a_i,0,0)$.
4. Rotación alrededor del eje x_i un ángulo α_i.

Como el producto de matrices no es conmutativo las transformaciones se deben realizar en el orden indicado en la ecuación 1:

$$^{i-1}A_i = \text{Rotz}\,(\theta_i)\ \text{T}\,(0,0,\ d_i)\ \text{T}\,(\alpha_i,0,0)\ \text{Rotx}(\alpha_i) \tag{1}$$

Del producto de matrices de las 4 transformaciones se obtiene la ecuación 2:

$$^{i-1}A_i = \begin{bmatrix} C\theta_i & -S\theta_i & 0 & 0 \\ S\theta_i & C\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\alpha_i & -S\alpha_i & 0 \\ 0 & S\alpha_i & C\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \therefore$$

$$^{i-1}A_i = = \begin{bmatrix} C\theta_i & -C\alpha_i S\theta_i & S\alpha_i S\theta_i & a_i C\theta_i \\ S\theta_i & C\alpha_i C\theta_i & -S\alpha_i C\theta_i & a_i S\theta_i \\ 0 & S\alpha_i & C \propto_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

Donde θ_i, d_i, a_i, $\propto_i$ son los parámetros de *D-H* del eslabón i.

De este modo, basta con identificar los parámetros θ_i, d_i, a_i, $\propto_i$ para obtener las matrices $^{i-1}A_i$ y relacionar así todos y cada uno los eslabones del robot.

Como se ha indicado, para que la matriz ${}^{i-1}A_i$, definida en la ecuación 2 relacione los sistemas $\{S_{i-1}\}$ y $\{S_i\}$, es necesario que los sistemas se hayan escogido de acuerdo a unas determinadas normas (Ramíres Benavides). Estas normas junto con los 4 parámetros de *D-H* conforman la resolución del problema cinemático directo, a continuación, se numeran dichos parámetros:

DH 1. Numerar los eslabones comenzando por *1* (primer eslabón móvil de la cadena) y acabado con n (último eslabón móvil). El eslabón *0* corresponde a la base fija del robot.

DH 2. Numera cada articulación comenzando por *1* (la correspondiente al primer grado de libertas) y acabando en n.

DH 3. Localizar el eje de cada articulación. Si ésta es rotativa, el eje será su propio eje de giro. Si es prismática, será el eje a lo largo del cual se produce el desplazamiento.

DH 4. Para i de 0 a $n-1$ situar el eje z_i sobre el eje de la articulación $i+1$.

DH 5. Situar el origen del sistema de la base $\{S_0\}$ en cualquier punto del eje z_0. Los ejes x_0 e y_0 se situarán de modo que formen un sistema dextrógiro con z_0.

DH 6. Para i de 1 a $n-1$, situar el origen del sistema $\{S_1\}$ (solidario al eslabón i) en la intersección del eje z_i con la línea normal común a z_{i-1} y z_i. Si ambos ejes se cortasen se situaría $\{S_i\}$ en el punto de corte. Si fuesen paralelos $\{S_1\}$ se situaría en la articulación $i+1$.

DH 7. Situar x_i en la línea normal común a z_{i-1} y z_i.

DH 8. Situar y_1 de modo que forme un sistema dextrógiro con x_i y z_1.

DH 9. Situar el sistema $\{S_n\}$ en el extremo del robot de modo que z_n coincida con la dirección de z_{n-1} y x_n sea normal a z_{n-1} y z_n.

DH 10. Obtener θ_i como el ángulo que hay que girar en torno a z_{i-1} para que x_{i-1} y x_i queden paralelos.

DH 11. Obtener d_i como la distancia, medida a lo largo de z_{i-1}, que habría que desplazar $\{S_{i-1}\}$ para que x_i y x_{i-1} quedasen alineados.

DH 12. Obtener, así como la distancia medida a lo largo de x_i (que ahora coincidiría con $\mathrm{x_i} - 1$) que habría que desplazar el nuevo $\{S_{i-1}\}$ para que su origen coincidiese con $\{S_i\}$.

DH 13. Obtener $\boldsymbol{\alpha_i}$ como el ángulo que habría que girar en torno a x_i, para que el nuevo $\{S_{i-1}\}$ coincidiese totalmente con $\{S_i\}$.

DH 14. Obtener las matrices de transformación ${}^{\mathrm{i-1}}\mathrm{A_i}$ definidas en la ecuación 2.

DH 15. Obtener la matriz de transformación que relaciona el sistema de la base con el del extremo del robot $T = {}^{0}\mathrm{A}_1\ {}^{1}\mathrm{A}_2\ {}^{\mathrm{n-1}}\mathrm{A_n}$.

DH 16. La matriz T define la orientación (sub-matriz de rotación) y posición (sub-matriz de traslación) del extremo referido a la base, en función de las **n** coordenadas articulares.

Los cuatro parámetros de *D-H* (θ_i, d_i, a_i, α_i) dependen únicamente de las características geométricas de cada eslabón y de las articulaciones que se enlazan en ese punto.

θ_i Es el ángulo que forman los ejes x_{i-1} y x_i medido en un plano perpendicular al eje z_{i-1}, utilizando la regla de la mano derecha. Se trata de un parámetro variable en articulaciones giratorias.

d_i Es la distancia a lo largo del eje z_{i-1} desde el origen del sistema de coordenadas $(i-1)$−ésimo hasta la intersección del eje z_{i-1} con el eje x_i. Se trata de un parámetro variable en articulaciones prismáticas.

a_i Es la distancia a lo largo del eje x_i que va desde la intersección del eje z_{i-1} con el eje x_i hasta el origen del sistema i −ésimo, en el caso de articulaciones giratorias. En el caso de articulaciones prismáticas, se calcula como la distancia más corta entre los ejes z_{i-1} y z_i.

α_i Es el ángulo de separación del eje z_{i-1} y el eje z_i, medido en un plano perpendicular al eje x_i, utilizando la regla de la mano derecha.

Una vez obtenidos los parámetros *D-H*, el cálculo de las relaciones entre los eslabones consecutivos vienen dadas por las matrices ${}^{i-1}A_i$, que se calculan con la expresión general de la ecuación 2. Las relaciones entre los eslabones consecutivos dos a dos vienen dadas por la ecuación 3 y su conjunto de matrices *A* y sus grados de libertad.

$$T(q_1 \dots q_n) = {}^{0}A_1(q_1)\ {}^{1}A_2(q_2) \dots {}^{n-1}A_n(q_n) \qquad (3)$$

Obtenida la matriz T, ésta expresará la orientación (sub-matriz 3×3 de rotación) y la posición (sub-matriz 3×1 de traslación) del extremo del robot en función de sus coordenadas articulares, con lo que quedará resuelto el problema cinemático directo para cualquier robot industrial con n grados de libertad.

2.4.2 Cinemática inversa

La cinemática inversa como su nombre lo advierte hace lo contrario a la cinemática directa, es decir, consiste encontrar los valores que deben tomar las coordenadas articulares del robot $q = [q_1, q_2, \dots, q_n]^T$ según la posición y orientación de extremo final (efector final) en el espacio $(p, [n, o, a])$ (Giraldo, Delgado, & Castellanos, 2006). Para llevar a cabo este procedimiento inverso la obtención de ecuaciones está fuertemente ligado a la configuración del robot por lo que se hace necesario encontrar una relación matemática explicita como se observa en la ecuación 4.

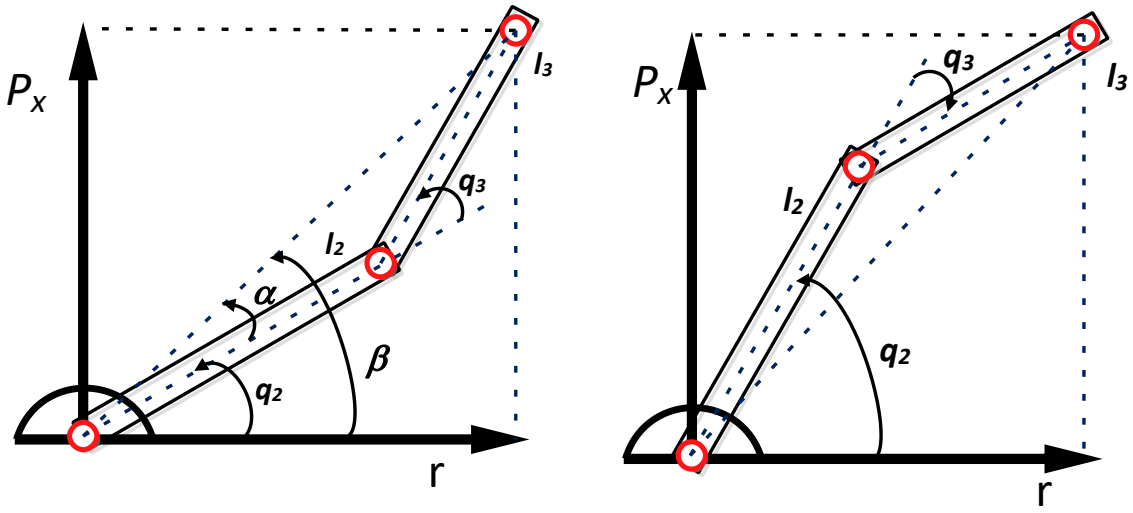
$$q_k = f_k(x, y, z, \phi, \theta, \psi) \tag{4}$$

Donde k = 1, 2, …, n grados de libertad GDL,

Cuando los grados de libertad del robot son mayores a 3 lo más recomendable es resolver la cinemática inversa por secciones para garantizar una limitada cantidad de variables para evitar los complejos cálculos matemáticos y los errores.

3.4.2.1 Resolución de la cinemática inversa por métodos geométricos.

Este método se basa en encontrar un gran número de relaciones geométricas en las que intervengan las coordenadas del efector final, las coordenadas articulares y las dimensiones físicas de los elementos del robot. Se comienza con las coordenadas (P_x, P_y, P_z) referidas a $\{S_0\}$ en las que se quiere posicionar el extremo del robot (Ramírez Benavides). La Figura **3** muestra las variables que intervienen en un robot articular para determinar su cinemática inversa cuando se encuentra codo abajo (a) y codo arriba (b).



a) Codo abajo. b) Codo arriba.

Figura 3. Variables geométricas que intervienen en el cálculo de las articulaciones $\boldsymbol{q_2}$ y $\boldsymbol{q_3}$ (Barrientos, Peñin, Balaguer, & Aracil, 2007).

La primera variable articular se denomina q_1 y se obtienen utilizando la ecuación 5.

$$q_1 = \text{atan2}(P_y, P_x) \tag{5}$$

Para determinar la coordenada articular q_3 se tiene la ecuación 6:

$$\sin q_3 = \pm\sqrt{1 - \cos^2 q_3} \tag{6}$$

Resolviendo queda la ecuación 7.

$$q_3 = \operatorname{atan2}\left(\pm\sqrt{1 - \cos^2 q_3}, \cos q_3\right) \tag{7}$$

Con la ecuación 8 se encuentra de la solución de la coordenada articular q_3.

$$\cos q_3 = \frac{P_x^2 + P_y^2 + P_z^2 - l_2^2 - l_3^2}{2 I_2 I_3} \tag{8}$$

La solución de la articulación q_3 toma valores positivos y negativos cuando el robot se encuentra codo arriba o codo abajo.

Para el calcular la coordenada articular q_2 se parte de la diferencia de β y α, como se muestra en la ecuación 9:

$$q_2 = \beta - \alpha \tag{9}$$

Donde:

$$\beta = \operatorname{atan2}(P_z, r) = \operatorname{atan2}\left(P_z, \pm\sqrt{P_x^2 + P_y^2}\right) \tag{10}$$

$$\propto = \operatorname{atan2}(l_3 \sin q_3, l_2 + l_3 \cos q_3) \tag{11}$$

Reemplazando en la ecuación 9 se resolver la articulación q_2 en la ecuación 12.

$$q_2 = \operatorname{atan2}\left(P_z, \pm\sqrt{P_x^2 + P_y^2}\right) - \operatorname{atan2}(l_3 \sin q_3, l_2 + l_3 \cos q_3) \quad (12)$$

Como q_3, q_2 también tiene 2 soluciones, una negativa y otra positiva, equivalentes a las configuraciones de codo arriba y codo abajo. Con las ecuaciones 5, 6 y 7 se resuelve el problema cinemático inverso para los primeros 3 grados de libertad de un robot articulado.

2.5 Control cinemático

Con el control cinemático se fijan las estrategias necesarias para que el robot mejore la calidad en sus movimientos, definiendo las trayectorias a seguir de cada una de las articulaciones a través del tiempo con el objetivo de representar la localización del efector final fijada por el usuario, tomando como referencia el punto de destino, las trayectorias cartesianas, el tiempo invertido, la cantidad de datos a manipular, entre otros (Garcia Calandín, 2008).

La trayectoria tiene restricciones que coinciden con las limitaciones físicas propias de cada accionamiento del robot, lo que influye en la calidad de sus movimientos, afectando la suavidad y precisión con la que se lleva a cabo el desplazamiento del efector final en el tiempo (Reyes Cortés, 2011).

Entre las funciones del control cinemático se encuentran las siguientes:

- **Función CC-1.** Interpretar las coordenadas del movimiento mediante una trayectoria analítica en el espacio cartesiano (evolución de cada coordenada cartesiana en función del tiempo).

- **Función CC-2.** Muestrear la trayectoria cartesiana con un número finito de puntos que representen dicha trayectoria. Cada uno de estos puntos vendrá dado por una *6-upla*, típicamente $(x, y, z, \phi, \theta, \psi)$.

- **Función CC-3.** Convertir cada uno de estos puntos en sus correspondientes coordenadas articulares $(q_1, q_2\ q_3, q_4, q_5, q_6)$, aplicando la cinemática inversa. Debe asegurarse de la continuidad de la trayectoria, teniendo en cuenta una posible solución múltiple de la transformación homogénea inversa, así como la posibilidad de ausencia de solución y puntos singulares.

- **Función CC-4.** Interpolación de los puntos articulares obtenidos, genere para cada variable articular una expresión $q_i(t)$ de la trayectoria realizable por el robot, transformándose en una trayectoria cartesiana lo más parecidas a las coordenadas registradas en cuanto a precisión, velocidad, entre otros.

- **Función CC-5**. Muestreo de la trayectoria articular para generar las referencias necesarias para el control dinámico.

Hay que tener en cuenta que dependiendo del tipo de robots y sus grados de libertad se podrán omitir algunas de las funciones anteriores (Checa, Luna, & Mosquera, 2009). Por tanto, para abordar el control cinemático de un robot se deben conocer el tipo de trayectorias que puede reproducir el robot *(CC-1)* así como él tipo de interpoladores que unen las articulaciones con las que el robot representa los movimientos *(CC-4)*. Adicional a esto es conveniente establecer criterios de selección para escoger cuales y cuantos puntos serán muestreados en el sistema coordenado *(CC-2)* y sobre todo con qué frecuencia se actualizan las referencias articulares del control dinámico *(CC-5)* (Moran, Künning, & Cuello, 2011).

2.5.1 Tipos de trayectorias en robots industriales

Definir el movimiento que debe ejecutar un robot para desplazarse de un lugar a otro requiere planificar su trayectoria en función del tiempo, teniendo en cuenta las restricciones físicas de cada robot con el fin de generar trayectorias de calidad con características como la suavidad, la precisión, entre otros (Ocampo, Olague, & Clemente, 2014) . Para planificar correctamente la trayectoria se pueden seguir los siguientes pasos:

1. Estudiar las necesidades del movimiento que se quiere generar para obtener una expresión analítica en coordenadas cartesianas de la trayectoria deseada en función del tiempo libre de colisiones con el entorno.
2. Muestrear la trayectoria deseada utilizando una serie de puntos finitos llamados nudos de control que se usan como puntos de referencia para indicar el inicio y el final de cada segmento trazado, especificando todas sus componentes cartesianas de posición y orientación *(x, y, z, α, β, γ).*
3. Pasar estos puntos a coordenadas articulares del robot utilizando la transformación homogénea inversa.
4. Realizar la interpolación de los puntos que describen la trayectoria coordenada y obtener para cada articulación la expresión de tipo $q_i(t)$ para cada segmento.

Teniendo en cuenta los anteriores puntos, la generación de trayectorias se encarga de controlar y coordinar los movimientos individuales de cada eslabón del robot para que lleven a cabo la trayectoria a desarrollar en el espacio de las variables de la articulación o en el

espacio cartesiano del manipulador (Saltarén Pazmiño, Azorín Poveda, Almonacid Kroeger, & Sabater Navarro). Los robots disponen de dos tipos de trayectorias punto a punto y continuas, definidas a continuación:

2.5.1.1 Trayectoria punto a punto

En este tipo de trayectoria cada articulación se mueve independientemente de las otras para ir desde su posición inicial hasta la final en el menor tiempo posible (Barrientos, Peñin, Balaguer, & Aracil, 2007), para realizar la interpolación de estos puntos se debe considerar la distancia entre ellos, ni muy separados ni muy juntos con el fin de no generar movimientos impredecibles ni muy tedioso para su cálculo de control, logrando así, que el robot haga su trayectoria recorriendo dichos puntos.

2.5.1.2 Trayectoria continua

En este caso el sistema de control de trayectoria continua debe hacer que el robot reproduzca los movimientos lo más fielmente posible, haciendo que coincidan con la trayectoria conocida por el usuario (Universidad Sntiago de Chile, 2020). Como resultado, cada articulación reproduce un movimiento caótico con posibilidades de cambio de dirección y velocidad, sin coordinación con el resto de los eslabones presentando una compleja evolución temporal. Sin embargo, el resultado conjunto de las articulaciones muestra que el extremo del robot describe la trayectoria deseada (Barrientos, Peñin, Balaguer, & Aracil, 2007).

2.6 Interpolación de trayectorias

Al unir esta sucesión de puntos articulares se deben garantizar junto con la configuración del tiempo las restricciones de velocidad y aceleración máxima de cada articulación, de manera que se consiga una trayectoria realizable y suficientemente suave (Figueroa de godos, 2015). Para no utilizar una función interpoladora de grado $n + 1$, se utilizan un conjunto de funciones más simples que unen solo unos pocos puntos consecutivos de un intervalo sobreponiéndose entre sí para garantizar su continuidad (Sierra Bueno & Martinez Angel, 2002). Existen varias técnicas de interpolación, pero se detalla a continuación la que mejor se ajusta al desarrollo de este proyecto.

2.6.1 Interpolador splin cúbico

Para asegurar los puntos que pasan por la trayectoria se une cada par de puntos adyacentes generando un polinomio de grado tres (Suñer, Valero, Ródenas, & Besa, 2007). La trayectoria está compuesta ahora por un conjunto de polinomios cúbicos, validos solo para dos puntos consecutivos denominados splines que dan continuidad en posición y velocidad al formar la trayectoria (Soto Cajiga, Vargas Soto, & Pedraza Ortega, 2006). La expresión de la trayectoria que une dos puntos adyacentes está dada por las ecuaciones 13 a la 15 que se muestran a continuación:

$$q(t) = a + b\left(t - t^i\right) + c\,(t - t^i)^2 + d\,(t - t^i)^3 \qquad (13)$$

Donde:

$a = q^i$

$b = \dot{q}^i$

$t^i < t < t^{i+1}$

$$c = \frac{3}{T^2}\left(\dot{q}^{i+1} - \dot{q}^i\right) - \frac{1}{T}\left(\dot{q}^{i+1} - 2\dot{q}^i\right) \quad (14)$$

$$d = \frac{2}{T^3}\left(\dot{q}^{i+1} - \dot{q}^i\right) - \frac{1}{T^2}\left(\dot{q}^{i+1} - \dot{q}^i\right) \quad (15)$$

Y donde:

$T = t^{i+1} - t^i$

Para resolver las anteriores ecuaciones se debe conocer las velocidades de paso $\dot{q}$ utilizando las siguientes alternativas expresadas en las ecuaciones 16 y 17.

$$\dot{q}^i = \begin{cases} 0 & si\ signo\left(q^i - q^{i-1}\right) \neq signo\left(q^{i+1} - q^i\right) \\ \frac{1}{2}\left[\frac{q^{i+1} - q^i}{t^{i+1} - t^i} + \frac{q^i - q^{i-1}}{t^i - t^{i-1}}\right] & si \begin{cases} signo\left(q^i - q^{i-1}\right) = signo\left(q^{i+1} - q^i\right) \\ o \quad q^{i-1} = q^i \\ o \quad q^i = q^{i+1} \end{cases} \end{cases} \quad (16)$$

Si se utiliza la ecuación 16 solo se permite la continuidad en función de la velocidad de los puntos adyacentes sin tener en cuenta la posición y velocidad de los mismos. Para que el resultado del spline cúbico sea continuo en función de la posición, velocidad y aceleración

de los puntos adyacentes que forman la trayectoria se debe resolver el sistema de ecuaciones con la expresión que se muestra en la ecuación 17.

$$\begin{bmatrix} t^3 & 2(t^2+t^3) & t^2 & 0 & 0 & \cdots \\ 0 & t^4 & 2(t^3+t^4) & t^3 & 0 & \cdots \\ 0 & 0 & t^5 & 2(t^4+t^5) & t^4 & \cdots \\ \cdots & \cdots & 0 & t^6 & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix} \cdot \begin{bmatrix} \dot{q}^1 \\ \dot{q}^2 \\ \dot{q}^3 \\ \vdots \\ \dot{q}^k \end{bmatrix}$$

$$= \begin{bmatrix} \frac{3}{t^2 t^3}[(t^2)^2(q^3-q^2)+(t^3)^2(q^2-q^1)] \\ \frac{3}{t^3 t^4}[(t^3)^2(q^4-q^3)+(t^4)^2(q^3-q^2)] \\ \vdots \\ \frac{3}{t^{k-1}t^k}[(t^{k-1})^2(q^k-q^{k-1})+(t^k)^2(q^{k-1}-q^{k-2})] \end{bmatrix} \quad (17)$$

Una vez definido el sistema de ecuación 17 es preciso considerar que:

$$\dot{q}^1 = \dot{q}^k = 0$$

Donde la articulación parte de un punto y llega a una situación de reposo, permitiendo obtener las k velocidades de paso de los spline cúbicos, asegurando la continuidad hasta la segunda derivada de la trayectoria global (Barrientos, Peñin, Balaguer, & Aracil, 2007).

2.7 Programación de robot

La programación de un *robot* es el procedimiento mediante el cual se le indican las acciones o movimientos que debe ejecutar para realizar una tarea específica, la trayectoria que se vaya a programar en el robot debe estar predefinida, es decir, su entorno de trabajo y los objetos a manipular se deben conocer previamente (Somolinos Sánchez & Salido Tercero,

2002), con el objetivo de programar las instrucciones necesarias para que desempeñe de forma controlada las tareas para las que fue diseñado (Cañas, Matellán, & Montúfar, 2006).

En la programación se deben manipular y actualizar las variables del sistema almacenadas en la memoria interna del *robot*, encargadas de envía las señales a los accionamientos del *robot* según las especificaciones del movimiento teniendo en cuenta las entradas y salidas del sistema, ayudando a la sincronización del robot con su entorno y demás objetos a su alrededor (López García & González Librán, 1996).

En perspectiva la programación es el método con el que el usuario puede acceder a todas las funcionalidades del *robot*, indicando paso a paso las diferentes acciones que deberán realizar de forma automática cuando se ponga en funcionamiento (Garcia, 2015). Normalmente los fabricantes de *robots* industriales desarrollan un método, una plataforma o un lenguaje de programación específico para sus *robots* (Guaraca Medina & Ochoa Ochoa, 2015). Existen 2 métodos de programación de robots que se mencionan a detalla a continuación:

2.7.1 Programación por guiado o aprendizaje

Este tipo de programación consiste en guiar de forma manual con ayuda de un operario las articulaciones del robot para formar la trayectoria con la que se realiza una tarea en específico, muchas veces para llevar a cabo este tipo de programación se requiere que el robot salga de la línea de operación, es decir, que se pause la cadena de producción durante

el tiempo que se necesite reprogramar el robot, lo cual perjudica la productividad de la empresa (Tornil Sin & Gámiz Caro, 2014).

En este tipo de programación existen dos sub divisiones:

- Guiado pasivo: El operador toma el extremo del robot y guía su trayectoria por los puntos deseados, mientras la unidad de control registra la señal de los sensores de manera automática en cada uno de los puntos del recorrido para luego reproducirlos. Suele utilizarse un prototipo del robot para que sea mucho más fácil de mover y mientras el operador realiza los movimientos que generan la trayectoria, estos se ven reflejados en el robot real donde la unidad de control muestrea y almacena con una frecuencia elevada los valores que van tomando los sensores de cada articulación a medida que se dibuja la trayectoria. La unidad de control guarda a la vez la velocidad con la que se generan los movimientos, así como la secuencia en la que el operador los realiza por lo que en muchas ocasiones no es necesario incluir ninguna estructura de control dentro del programa (Suaréz , 1999).
- Guiado activo: En este tipo de programación se utiliza un panel de control manual o *joystick* algo así como un control remoto, con el que se manipulan los accionamientos encargados de mover las articulaciones hasta una configuración determinada y guardarla. Con ayuda del panel se pueden configurar algunas características del movimiento, como tipo de trayectoria, velocidad, precisión, entre otras, así como también incluir instrucciones para el control de flujo del programa como saltos, repeticiones del movimiento, atención de entradas/salidas binarias, entre otros (Morales Corral, 2020).

Esta programación aparentemente es muy sencilla por lo que no es necesario programar coordenadas ni tener en cuenta las dimensiones del entorno ni los objetos alrededor para calibrar los movimientos, librando al programador de los problemas asociados a la existencia de soluciones múltiples o puntos singulares. Pero presenta otro tipo de inconvenientes como sacar al robot de la línea de operación para reprogramar su trayectoria, adicional a esto, no se puede documentar el código por lo que es imposible agregar alguna modificación, lo que hace difícil su depuración y puesta en marcha de la aplicación (Barrientos, Peñin, Balaguer, & Aracil, 2007).

2.7.2 Programación textual

Este tipo de programación utiliza un lenguaje informático específico para programar las instrucciones sobre las tareas a realizar de un robot, permitiendo movimientos más complejos y precisos, sin necesidad de sacarlo de la línea de producción (García Monsálvez, 2017).

Para establecer una relación entre el robot y su entorno se deben utilizar a la hora de programar los datos proporcionados por los sensores de los actuadores con el fin de que el robot actué por sí mismo tal como lo hacen los robots inteligentes. La programación textual puede ser representada en dos tipos explicados a continuación (AUTOMÁTICA INDUSTRIAL, 2020):

- Programación textual explicita: Es un lenguaje de programación estructurado, donde se programa de manera secuencial y estructurada el conjunto de instrucciones que

debe realizar el robot para llevar a cabo una tarea específica, teniendo en cuenta si se desea las características de su entorno (Soto Cajiga, Vargas Soto, & Pedraza Ortega, 2005).

- Programación textual especificativa: Utiliza un lenguaje de programación orientada a objetos, enfocada en las piezas que debe manipular el robot y las acciones que debe realizar con ellas teniendo muy presente el entorno en el que se desarrollan dichas tareas (Román Romero, 2014).

Los niveles de programación se dividen en robot, objeto y tarea: Donde el nivel de robot es el que actualmente se sigue utilizando con mayor frecuencia, debido a que este lenguaje se enfoca en desarrollar instrucciones o comandos que tengan que ver con los movimientos que debe realizar el robot para llevar a cabo una tarea en específico (Bermudez, 2002).

El nivel de objeto, es el que mayores dificultades presenta en su desarrollo, así como poca eficiencia, por lo que no es muy común desarrollar lenguajes de programación a este nivel (Slawiñski, Postigo, Mut, Carestía, & Castro, 2007).

Por otro lado el nivel tarea es de los más complejos, ya que busca soluciones a retos propios de la inteligencia artificial donde por ejemplo, el robot debe ser capaz de reconocer el estado de su entorno para identificar las diferencias entre el modelo previsto y el estado real, adicional a esto debe escoger la mejor forma de agarrar el objeto para que durante su transporte no pierda estabilidad, demostrando así un adecuado tratamiento de la información

a través de la recopilación de datos suministrados por el sistema sensorial del robot (Barrientos, Peñin, Balaguer, & Aracil, 2007).

De forma general se presenta el siguiente cuadro contextual en la Figura 4 donde se resume los conceptos y tipos de programación de robots.

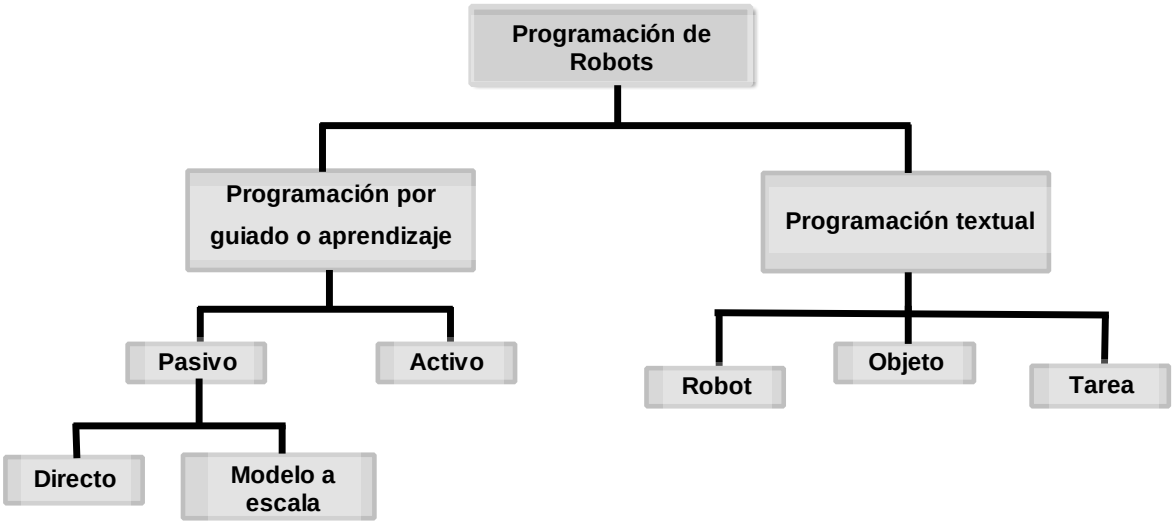


Figura 4.Diagrama de flujo de los tipos de programación (Barrientos, Peñin, Balaguer, & Aracil, 2007).

2.7.3 ¿Cómo programar un robot?

Como en muchos de los casos programar debe tener ciertos criterios o estándares que faciliten la interpretación e interacción con el programador, pero cuando se trata de programar robots industriales las reglas de juego son diferentes (Rico Riveros, y otros, 2020). Para programar robots se necesitan algoritmos que faciliten el desarrollo de las tareas para las que fue diseñado, permitiendo la traducción de dicho algoritmo a un lenguaje de programación inteligente que el sistema de control del robot pueda interpretar (Crus Lara, 2018). Entre los requerimientos generales que se deben tener en cuenta cuando se realiza el

código con el que se programa un robot industrial están los siguientes (Barrientos, Peñin, Balaguer, & Aracil, 2007):

- Entorno de programación
- Modelado del entorno
- Tipo de datos
- Manejo de Entradas/Salidas (digitales y análogas)
- Comunicaciones
- Control de movimiento
- Control del flujo de ejecución del programa

En la programación de robots hay dos tipos de código, los habituales o de propósito general utilizados en la programación de robots educativos o de investigación (Meza & Mamaní, 2018) y el código especializado, que es básicamente un algoritmo estructurado para cada robot en específico desarrollado por el fabricante sin poder emplearse en otros robots (Ramos Fuentes, 2000).

2.8 Realidad aumentada

El aprendizaje en la programación de robots puede mejorar aplicando las tecnologías de la informática y la comunicación, ya que como es sabido, enseñar o aprender esta especialidad requiere tener conocimientos en múltiples disciplinas como la mecánica, la electrónica y la programación, además de la complejidad del control involucrado en los

sistemas (Solis Bautista, Mendoza Pérez, & Cruz Flores, 2020). Teniendo en cuenta lo anterior, se ha buscado mejorar significativamente el aprendizaje de la programación de robot utilizando una de las herramientas tecnológicas de la informática y la comunicación que actualmente presenta mejores resultados y es la Realidad Aumentada (RA) o *Augmented Reality (AR)* con la que programar robots se puede hacer de forma rápida y sencilla sin necesidad de ser un especialistas en el área, lo que ayuda en el aprendizaje de dicha disciplina y puesta en marcha de robots industriales en las empresas sin generar sobre costos de personal (Mendoza Pérez, Cruz Flores, Vilbalba Hernández, Calderón Rodríguez, & Arreola Patiño, 2017).

2.8.1 ¿Qué es la realidad aumentada?

La realidad amentada es la tecnología que permite percibir el mundo a través de los dispositivos electrónicos, como cámaras, teléfonos, computadoras, visores, entre otros y un software especializado alguna de las variaciones virtuales que se sobreponen en el espacio real, potenciando los sentidos del tacto, la vista, el oído, el olfato y el gusto para complementar el mundo real con la era digital (Jara Bravo, 2020) y (Artopoulos, y otros, 2010).

Con la realidad aumentada se logra incluir objetos o entornos virtuales a un ambiente real, haciendo que el usuario experimente la sensación de que están ahí (Solis Bautista, Mendoza Pérez, & Cruz Flores, 2020). La realidad aumentada tiene varios niveles, los cuales miden la complejidad de la tecnología con la que se desarrollan los sistemas de realidad aumentada, generalmente se usa un marcador que corresponde al nivel 1, basado en el

reconocimiento de patrones 2D para incluir los elementos virtuales al mundo real utilizando información digital (Prendes Espinosa, 2015). Los componentes de un sistema de realidad aumentada se dividen en: generador de escenas, sistema de rastreo, técnicas de rastreo y pantallas (Arce, 2014).

Los tipos de realidad aumentada se dividen en (Arce, 2014):

- Realidad Aumentada Estática: Corresponde a una aplicación en la que los objetos aparecen de forma estática unidimensional o tridimensional, sin que ellos realicen ninguna acción o movimiento.
- Realidad Aumentada Dinámica: Incluye objetos con movimiento para interactuar con el usuario, dando información adicional del entorno.
- Realidad Aumentada de Geolocalización: La realidad aumentada con la geolocalización permite crear aplicaciones de gran calidad en diseño y usabilidad. Combinando ambas tecnologías se consigue un aliado competente en los contenidos de todo tipo de aplicaciones.

La realidad aumentada está surgiendo como una de las tecnologías vanguardistas de este siglo y su uso en diversas áreas se debe al aporte que está ofrece en cada una de ellas de forma relevante, entre las aplicaciones en las que se utiliza la realidad aumentada se encuentran la medicina, la arquitectura, la educación, la fabricación, la ingeniería, la reparación, la robótica, entre otras (Arce, 2014).

La realidad aumentada puede complementarse con (Negrón, 2017):

- La Realidad Virtual: Consiste en un ambiente artificial creado solamente por computadora muy parecido al mundo real, capaz de llevar al usuario a experimentar dicho entorno como si fuera real.
- La Realidad Mixta: Mezcla en el ambiente real con objetos o espacios virtuales haciendo que la interacción artificial parezca real, utilizando los cambios de perspectiva que el usuario experimenta si estuviera en un entorno real.

La aplicación de la realidad aumentada en la robótica, más en específico en mejorar la teleoperación, es la de mayor interés en el campo académico y tecnológico. Consiste en transportar al operador a un espacio de trabajo virtual donde él pueda proporcionar de forma intuitiva los movimientos que generen a partir de comandos la información necesaria para construir una trayectoria que luego sea reproducida por el robot (Peña Tapia, Roldán, Garzón, Martín Barrio, & Barrientos, 2017).

Para llevar a cabo la teleoperación de la mano con la realidad aumentada es necesario apoyarse en un *software* especializado que a partir de un marcador especifico active el entorno virtual que se superpone al ambiente real cuando sea captado a través de una cámara o lente. Dicho marcador debe ser renderizado, es decir, que a partir de una imagen digital se simule de manera fotorrealista el ambiente, los materiales, las luces, los objetos y demás características del ambiente virtual (Arce, 2014). El *software* más popular para desarrollar

entornos virtuales de la mano con la realidad aumentada es Unity utilizado en su mayoría para desarrollar videojuegos.

2.8.2 ¿Qué es Unity?

Unity es un *software* de desarrollo de videojuegos en tiempo real creada por la empresa *Unity Tecnologies***.** Esta herramienta cuenta con un motor para renderizar imágenes, motor de audios y un motor de animaciones, que involucran la realidad virtual en proyectos de arquitectura, ingeniería, robótica, cine, marketing, entre otras (Quality devs, 2020). Unity está basado en el lenguaje de programación *C++* y su propósito es facilitar una plataforma para mejorar el desarrollo de videojuegos en tiempo real, que sea autónoma y se actualice constantemente (Platzi, 2020).

Algunas de las funciones que tiene Unity en el desarrollo de videojuegos específicamente son (mister.D, 2020):

- Motor para renderizar gráficos 2D y 3D.
- Motor que simula las leyes de la física.
- Animaciones.
- Sonido.
- Inteligencia artificial.
- Programación o *Scripting*.
- Entre otros.

3. MODELO CINEMÁTICO DEL ROBOT

El análisis del modelo cinemático se aplica a un robot antropomórfico de 4 grados de libertad llamado Raplim que se encuentra en la Universidad de Pamplona UP (Sede Principal), con el que se evalúa la ejecución de la trayectoria analizada y simulada de los movimientos capturados mediante visión artificial del operador.

En la Figura 5 se describe por medio de un diagrama de bloques los pasos para generar una trayectoria aplicando el control por guiado gestual para un robot tipo brazo robótico.

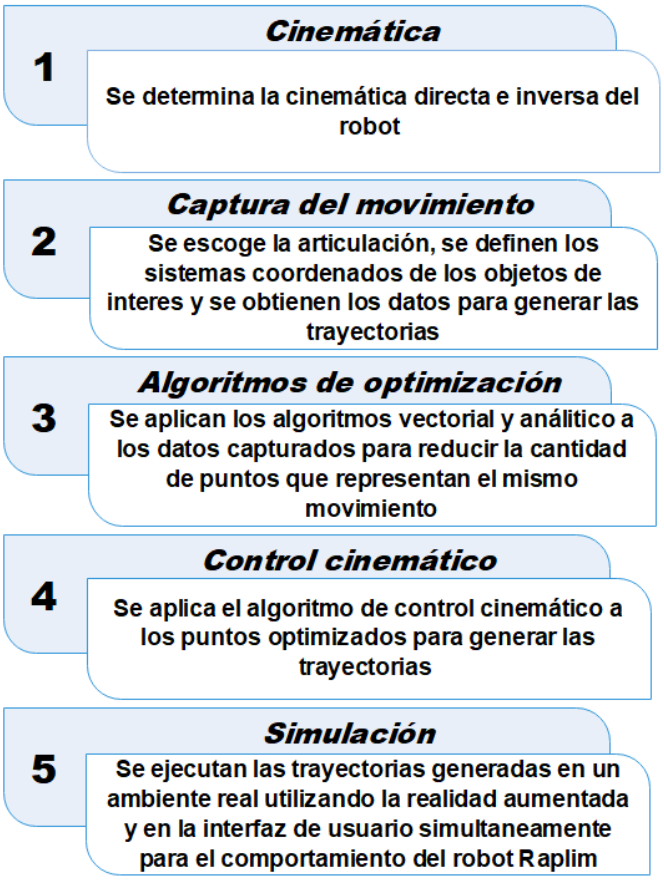


Figura 5. Diagrama de bloques paso a paso para la generación de la trayectoria.

Para calcular la cinemática del robot se debe encontrar la relación entre las articulaciones y la localización (posición y orientación) del efector final aplicando la cinemática directa e inversa, vista en el capítulo 2.4.

3.1 Cinemática directa.

Conociendo la morfología y estructura mecánica del robot Raplim de la universidad de Pamplona se obtienen las longitudes de los eslabones y los tipos de articulaciones con los cuales se determina la posición y orientación del efector final del robot, a partir de las coordenadas articulares con respecto a un sistema de referencia fijo (base del robot).

Para hacer el cálculo de la cinemática directa se usa una serie de reglas conocidas como el algoritmo de *Denavit Hartenberg*, donde se toman cuatro transformaciones básicas indicadas en el capítulo 3, que permiten relacionar el sistema de referencia del eslabón **i** con el sistema de referencia del eslabón $i + 1$.

En la Figura 6 se detalla el modelo simplificado del robot Raplim y la orientación de los ejes coordenados correspondientes a cada eslabón.

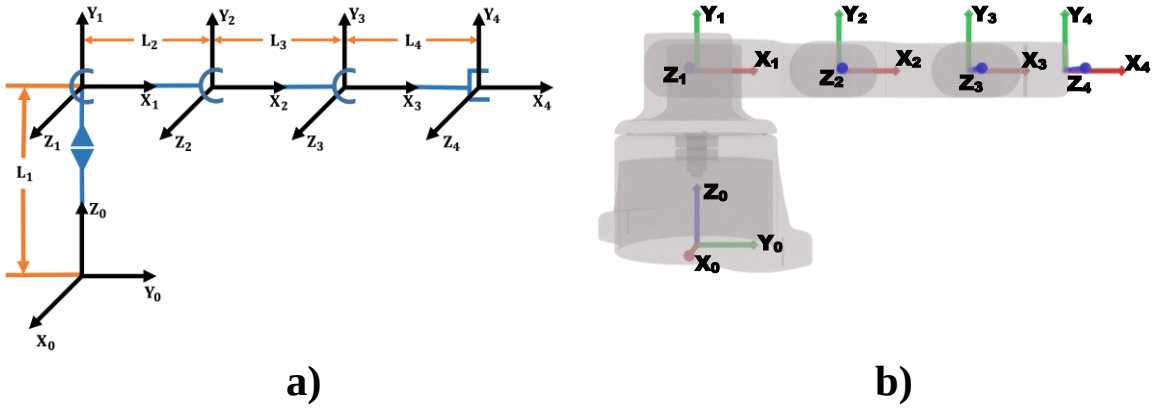


Figura 6. Modelo simplificado del robot Raplim a) Modelo de alambres del robot Raplim y b) Modelo estructural del robot Raplim.

De acuerdo a la Figura 6 se obtienen los parámetros de *Denavit-Hartenberg* para el robot Raplim como se muestra en la Tabla 2.

Tabla 2.
Parámetros D - H del robot.

Articulación	θ	d(mm)	a(mm)	α
1	$q_1+\pi/2$	$L_1 = 135.5$	0	$\pi/2$
2	q_2	0	$L_2 = 120.23$	0
3	q_3	0	$L_3 = 94.6$	0
4	q_4	0	$L_4 = 65$	0

Nota: Los datos de la tabla fueron registrados a partir de las características de la estructura mecanica del robot Raplim.

Los valores de la Tabla 2 se reemplazan dentro de la matriz de *Denavit-Hartenberg* de la ecuación 2, por lo que se debe calcular una matriz por cada articulación.

Aplicando el torema de coseno y seno de la suma de ángulos de las ecuaciones 18 y 19 se obtienen los valores de cada una de las posiciones de las matrices.

$$\cos(a+b) = \cos(a)\cos(b) - \sin(a)\sin(b) \tag{18}$$

$$\sin(a+b) = \sin(a)\cos(b) - \sin(b)\cos(a) \tag{19}$$

Apartir de los teoremas anteriores se desarrollan calculos repetitivos que se encuentran en alguno de los terminos de las matrices descritos a partir de las ecuaciones 20 a la 31.

$$\cos\left(q_1 + \frac{\pi}{2}\right) = \cos(q_1)\cos\left(\frac{\pi}{2}\right) + \sin(q_1)\sin\left(\frac{\pi}{2}\right) \tag{20}$$

$$\cos\left(q_1 + \frac{\pi}{2}\right) = \sin(q_1) \tag{21}$$

$$\sin\left(q_1 + \frac{\pi}{2}\right) = \sin(q_1)\cos\left(\frac{\pi}{2}\right) + \sin(\pi/2)\cos(q_1) \tag{22}$$

$$\cos\left(q_1 + \frac{\pi}{2}\right) = \cos(q_1) \tag{23}$$

Las matrices correspondientes a cada articulación se definen en las ecuaciones 24 a la 26.

Primera articulación

$${}^{0}A_1 = \begin{vmatrix} \cos(q_1+\pi/2) & -\cos(\pi/2)\sin(q_1+\pi/2) & \sin(\pi/2)\sin(q_1+\pi/2) & 0\cos(q_1+\pi/2) \\ \sin(q_1+\pi/2) & \cos(\pi/2)\cos(q_1+\pi/2) & -\sin(\pi/2)\cos(q_1+\pi/2) & 0\sin(q_1+\pi/2) \\ 0 & \sin(\pi/2) & \cos(\pi/2) & 135.5 \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{24}$$

$${}^{0}A_1 = \begin{vmatrix} \sin(q_1) & 0 & \cos(q_1) & 0 \\ \cos(q_1) & 0 & -\sin(q_1) & 0 \\ 0 & 1 & 0 & 135.5 \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{25}$$

Segunda articulación

$${}^{1}A_2 = \begin{vmatrix} \cos(q_2) & -\cos(0)\sin(q_2) & \sin(0)\sin(q_2) & 120.23\cos(q_2) \\ \sin(q_2) & \cos(0)\cos(q_2) & -\sin(0)\cos(q_2) & 120.23\sin(q_2) \\ 0 & \sin(0) & \cos(0) & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{26}$$

$$ {}^{1}A_2 = \begin{vmatrix} \cos(q_2) & -\sin(q_2) & 0 & 120.23\cos(q_2) \\ \sin(q_2) & \cos(q_2) & 0 & 120.23\sin(q_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \quad (27) $$

Tercera articulación

$$ {}^{2}A_3 = \begin{vmatrix} \cos(q_3) & -\cos(0)\sin(q_3) & \sin(0)\sin(q_3) & 94.6\cos(q_3) \\ \sin(q_3) & \cos(0)\cos(q_3) & -\sin(0)\cos(q_3) & 94.6\sin(q_3) \\ 0 & \sin(0) & \cos(0) & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \quad (28) $$

$$ {}^{2}A_3 = \begin{vmatrix} \cos(q_3) & -\sin(q_3) & 0 & 94.6\cos(q_3) \\ \sin(q_3) & \cos(q_3) & 0 & 94.6\sin(q_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \quad (29) $$

Cuarta articulación

$$ {}^{3}A_4 = \begin{vmatrix} \cos(q_4) & -\cos(0)\sin(q_4) & \sin(0)\sin(q_4) & 65\cos(q_4) \\ \sin(q_4) & \cos(0)\cos(q_4) & -\sin(0)\cos(q_4) & 65\sin(q_4) \\ 0 & \sin(0) & \cos(0) & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \quad (30) $$

$$ {}^{3}A_4 = \begin{vmatrix} \cos(q_4) & -\sin(q_4) & 0 & 65\cos(q_4) \\ \sin(q_4) & \cos(q_4) & 0 & 65\sin(q_4) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \quad (31) $$

A partir del producto de las cuatro matrices se obtiene la matriz de la ecuación 32 que indica la orientacion y posición del efector final con respecto al sistema de referencia de la base del robot.

$$ {}^{0}A_4 = {}^{0}A_1\ {}^{1}A_2\ {}^{2}A_3\ {}^{3}A_4 = \begin{vmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{vmatrix} \quad (32) $$

La ecuación 32 muestra el modelo cinemático directo de forma general, el cual se pone a prueba ingresando las coordenadas articulares como cero *(0)* que indican la posición

inicial del robot y la localización de su efector final como se observa en el diagrama de la Figura 6. De esta manera las coordenadas articulares toman el valor de la ecuación 33.

$$q_1 = q_2 = q_3 = q_4 = 0 \tag{33}$$

De esta manera la matriz que define la posición del efector final de acuerdo a las coordenadas anteriores está dada por la ecuación 34:

$${}^{0}A_4 = {}^{0}A_1\,{}^{1}A_2\,{}^{2}A_3\,{}^{3}A_4 \tag{34}$$

Reemplazando queda la ecuación 35.

$${}^{0}A_4 = \begin{vmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 135.5 \\ 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} 1 & 0 & 0 & 120.23 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} 1 & 0 & 0 & 94.6 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} 1 & 0 & 0 & 65 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{35}$$

El resultado se muestra en la ecuación 36.

$${}^{0}A_4 = \begin{vmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 279.83 \\ 0 & 1 & 0 & 135.5 \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{36}$$

En la Figura 7 se detalla de forma clara el cambio de base del sistema coordenado del robot hasta el efector final, que es como se resuelve la cinemática directa del robot antropomórfico Raplim de *4* grados de libertad.

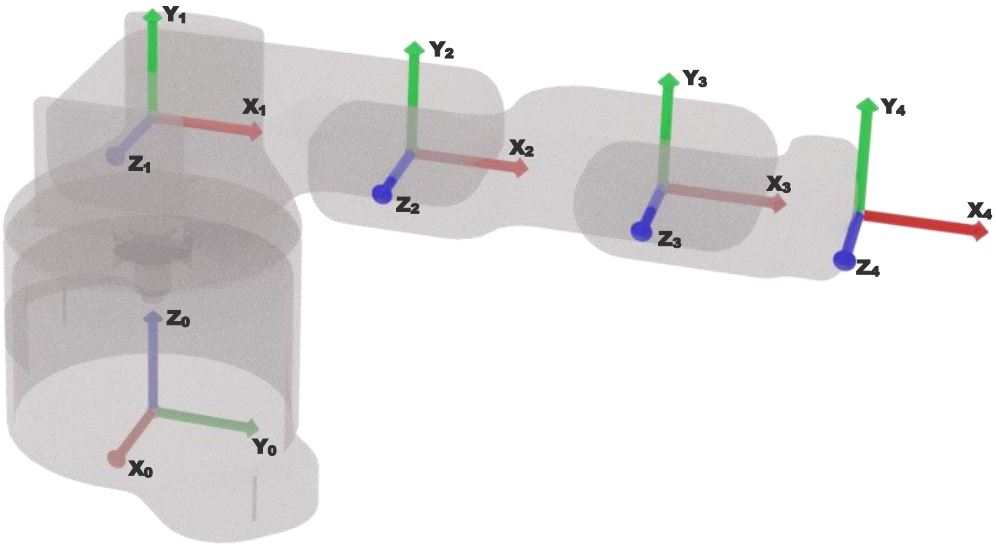


Figura 7. Eslabones en 3D del robot Raplim diseñado en Blender.

Ahora para corroborar cada coordenada articular se prueba ubicando el robot de forma vertical, dándole valores a las coordenadas articulares con la equivalencia que se le asigna en la ecuación 37.

$$q_1 = q_3 = q_4 = 0;\ q_2 = \frac{\pi}{2} \tag{37}$$

La matriz de transformación que representa la relación del efector final con respecto a la base del robot se muestra en la ecuación 38 y 39.

$${}^{0}A_4 = \begin{vmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 135.5 \\ 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 120.23 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} 1 & 0 & 0 & 94.6 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} 1 & 0 & 0 & 65 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{38}$$

$${}^{0}A_4 = \begin{vmatrix} 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 415.33 \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{39}$$

De acuerdo a las condiciones de la ecuación 39 el robot adopta una postura vertical desde su base hasta su efector final como se detalla en la Figura 8.

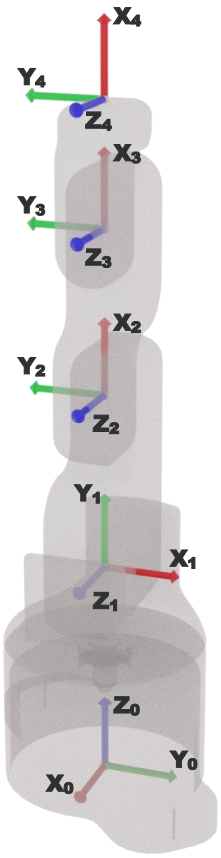


Figura 8. Postura vertical del robot Raplim en 3D.

3.2 Cinemática inversa

La cinemática inversa como su nombre lo indica, busca encontrar la forma que debe adoptar el robot a partir de la posición y orientación del efector final, es decir, lo contrario a la cinemática directa.

Para obtener los valores de las coordenadas articulares del robot a partir de la posición y orientación del efector final con respecto al sistema de referencia de su base, se considera resolver por el método geométrico los primeros 3 grados de libertad con los que cuenta el robot Raplim y el ultimo grado se resuelve por el método vectorial. Esta estrategia se usa debido a que el método geométrico se hace complejo cuando se resuelven más de 3 grados de libertad en un robot.

Partiendo de lo anterior se comienza a resolver la cinemática inversa de los primeros 3 grados de libertad del robot Raplim. En la Figura 9 se puede ver la vista superior del modelo simplificado para una posición del efector final $P = [P_x; P_y; P_z]$, con el cual se puede determinar la primera coordenada articular q_1.

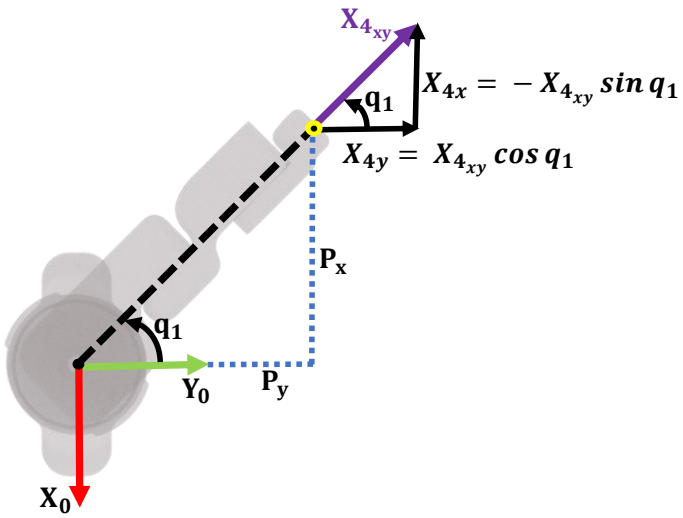


Figura 9. Diagrama para determinar la coordenada articular $\boldsymbol{q_1}$.

Con las ecuaciones 40 hasta la 42 se resuelve el arreglo para determinar el valor de la articulación q_1.

$$\sin q_1 = -P_x \tag{40}$$

$$\cos q_1 = P_y \tag{41}$$

$$q_1 = \text{atan2}\left(-P_x, P_y\right) \tag{42}$$

En la Figura 10 se ubica el punto de la muñeca *(P_m)* en el sistema 3 del robot a partir del cual se realizan los cálculos geométricos para determinar las articulaciones q_2 y q_3.

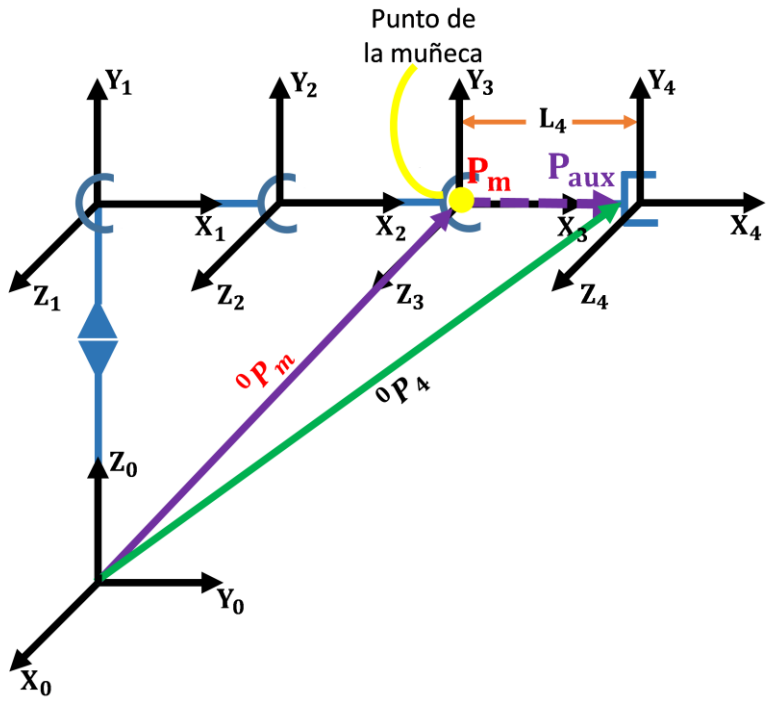


Figura 10. Representación punto de la muñeca.

Donde:

Pm = Punto de la muñeca

$Paux$ = Vector auxiliar

A partir de las indicaciones que se muestran en la figura se puede determinar las componentes del vector P_m mediante la suma de vectores descrita en las ecuaciones 43 a la 45.

$$P = P_m + P_{aux} \tag{43}$$

$$P_{aux} = X_4 L_4 \tag{44}$$

$$P_m = P - (X_4 L_4) \tag{45}$$

Con la ecuación 45 se hallan las componentes de P_m, pero antes se necesita determinar el valor del termino X_4. Con base en la Figura 11 se distinguen los términos que componen a X_4.

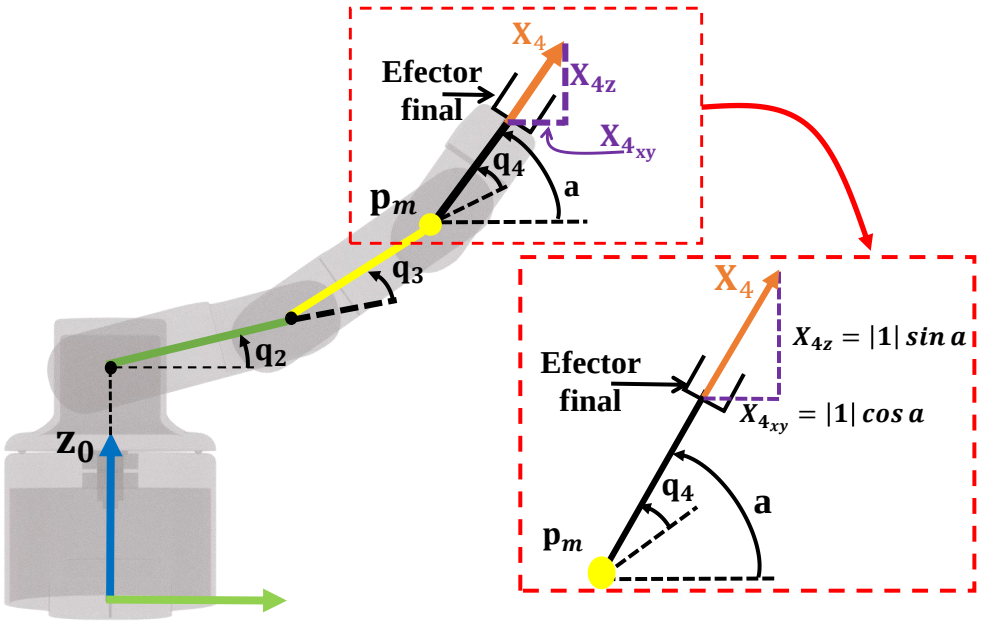


Figura 11. Diagrama para determinar las componentes de $\boldsymbol{X_4}$.

En la Figura 11 se detalla el ángulo a que indica los grados de giro del efector final con respecto al sistema de referencia de la base del robot. Este valor lo define el operador del robot y junto con las coordenadas cartesianas del efector final se calcula la cinemática inversa.

En la ecuación 46 se muestran las componentes de X_4.

$$\begin{bmatrix} X_{4_x} \\ X_{4_y} \\ X_{4_z} \end{bmatrix} = \begin{bmatrix} -\cos(a)\sin(q_1) \\ \cos(a)\cos(q_1) \\ \sin(a) \end{bmatrix} \tag{46}$$

Después de determinar X_4, se sustituye en la ecuación 45 para hallar el punto de la muñeca. Obtenido el valor de P_m, se comienza a analizar las variables para el cálculo de las articulaciones q_2 y q_3.

En la Figura 12 se muestra la vista superior del robot hasta el punto de la muñeca, vista que permite determinar la magnitud del segmento m entre el segundo sistema coordenado del robot y P_m.

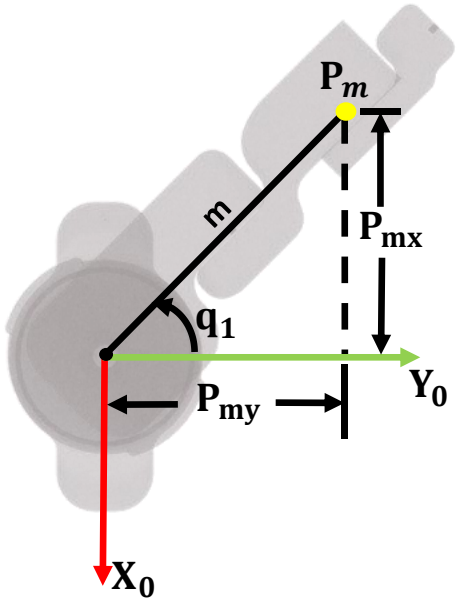


Figura 12. Vista superior de la primera articulación.

Aplicando el teorema de Pitágoras la ecuación 47 describe la solución del segmento m.

$$m = \sqrt{(p_{mx})^2 + (p_{my})^2} \tag{47}$$

La Figura 13 muestra las variables que hacen parte de la solución para determinar las coordenadas articulares q_2 y q_3.

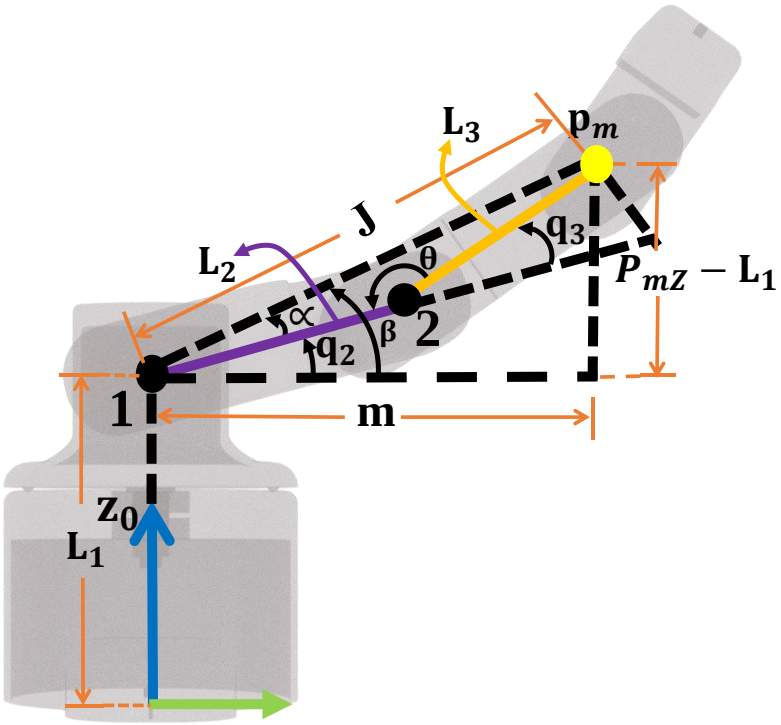


Figura 13. Vista lateral del robot y sus variables.

Para calcula q_3 se inicia hallando la distancia entre *1* y P_m (J), como se muestra en la ecuación 48, aplicando el teorema de Pitágoras.

$$J = \sqrt{(m)^2 + (p_{mz} - L_1)^2} \tag{48}$$

Otra forma de determinar el valor de J es usando el teorema de coseno para incluir en la solución el ángulo q_3 de la Figura 13. La ecuación 49 representa el teorema de coseno, la ecuación 50 describe la equivalencia de la variable θ, que se reemplaza en la ecuación 51.

$$J^2 = (L_2)^2 + (L_3)^2 - 2(L_2 L_3)\cos(\theta) \tag{49}$$

$$\theta = 180 - q_3 \tag{50}$$

$$J^2 = (L_2)^2 + (L_3)^2 - 2(L_2 L_3)\cos(180 - q_3) \tag{51}$$

Para despejar $cos\, q_3$ de la ecuación 51 se aplica *coseno* de la diferencia de ángulos dando como resultado las ecuaciones 52 a la 55.

$$\cos(180 - q_3) = \cos(180)\sin(q_3) + \sin(180)\sin(q_3) \tag{52}$$

$$\cos(180 - q_3) = -\cos q_3 \tag{53}$$

$$J^2 = (L_2)^2 + (L_3)^2 + 2(L_2 L_3)\cos(q_3) \tag{54}$$

$$\cos q_3 = \frac{J^2 - {L_2}^2 - {L_3}^2}{2L_2 L_3} \tag{55}$$

Utilizando la identidad trigonométrica de la ecuación 56 se obtiene el valor de $sin\, q_3$ que es necesario para hallar la articulación q_3 mediante el uso de la función trigonométrica $atan2$. Las ecuaciones 57 a la 60 demuestran el procedimiento matemático que se lleva a cabo para encontrar el valor de la articulación q_3.

$$cos^2\theta + sin^2\theta = 1 \tag{56}$$

$$sin\theta = \pm\sqrt{1 - cos^2\theta} \tag{57}$$

$$sin\theta = codo_{abajo}\sqrt{1 - cos^2\theta} \tag{58}$$

$$q_3 = \text{atan2}(\sin q_3 , \cos q_3) \tag{59}$$

$$q_3 = \text{atan2}\left(\sqrt{1 - \left(\frac{J^2 - {L_2}^2 - {L_3}^2}{2L_2L_3}\right)^2}, \frac{J^2 - {L_2}^2 - {L_3}^2}{2L_2L_3}\right) \tag{60}$$

La variable codo abajo tiene un valor de *-1* el cual indica la postura con la que se obtuvieron las ecuaciones geométricas para la **¡Error! No se encuentra el origen de la r eferencia.** cómo se describe en el capítulo 3 de este libro.

La coordenada articular q_2 se determina a partir de la diferencia entre β y $\propto$, este último esta expresado en función de q_3 cómo se detallan las ecuaciones 61 a la 64.

$$q_2 = \beta - \propto \tag{61}$$

$$\beta = \text{atan2}(p_{mz} - L_1, m) \tag{62}$$

$$\propto = \text{atan2}\big(L_3 \sin q_3 , L_2 + (L_3 \cos q_3)\big) \tag{63}$$

$$q_2 = \text{atan2}(p_{mz} - L_1, m) - \text{atan2}\big(L_3 \sin q_3 , L_2 + (L_3 \cos q_3)\big) \tag{64}$$

Finalmente, para obtener la coordenada articular q_4 se hace uso del vector X_4 previamente calculado y los vectores X_3 y Y_3 que se hallan mediante la matriz de transformación homogénea ${}^0A_3$ que se obtiene con solución de las 3 primeras coordenadas

articulares q_1, q_2 y q_3. La Figura 14 ilustra las variables que intervienen en el sistema a evaluar.

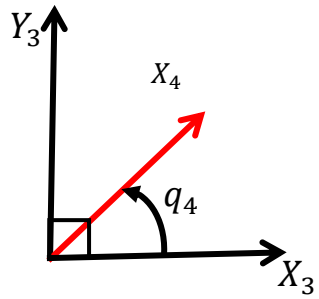


Figura 14. Vectores de la coordenada articular $\boldsymbol{q_4}$.

Para resolver q_4 se aplica el producto punto entre los vectores (X_4, X_3) y (X_4, Y_3), las ecuaciones 65 y 69 definen la solución del producto punto en función de q_4.

Producto punto del vector (X_4, X_3)

$$X_4 \cdot X_3 = |X_4||X_3|(\cos q_4) \tag{65}$$

$$X_4 \cdot X_3 = \cos q_4 \tag{66}$$

Producto punto del vector (X_4, Y_3)

$$X_4 \cdot Y_3 = |X_4||Y_3|(\cos 90 - q_4) \tag{67}$$

$$(\cos 90 - q_4) = \cos(90)\ \cos(-q_4) - \sin(90)\sin(-q_4) \tag{68}$$

$$X_4 \cdot Y_3 = \sin q_4 \tag{69}$$

De las ecuaciones 66 y 69 se despeja q_4 dando como resultado la ecuación 70.

$$q_4 = \text{atan2}(X_4 \cdot Y_3, X_4 \cdot X_3) \tag{70}$$

En resumen, las ecuaciones que se usan para resolver el modelo cinemático inverso del robot Raplim se encuentran definidas en la Tabla 3.

Tabla 3.
Fórmulas para el cálculo de las articulaciones del robot Raplim.

Articulación	Ecuaciones correspondientes
q_1	$\text{atan2}\left(-P_x, P_y\right)$
q_2	$\text{atan2}(p_{mz} - L_1, m) - \text{atan2}\left(L_3 \sin q_3\,, L_2 + (L_3 \cos q_3)\right)$
q_3	$\text{atan2}\left(\sqrt{1 - \left(\frac{J^2 - {L_2}^2 - {L_3}^2}{2L_2L_3}\right)^2}\,, \frac{J^2 - {L_2}^2 - {L_3}^2}{2L_2L_3}\right)$
q_4	$\text{atan2}(X_4 \cdot Y_3, X_4 \cdot X_3)$

Para comprobar la cinemática inversa se toma el valor de la posición del efector final (P) resuelto en la cinemática directa de la ecuación 36, donde el resultado de la matriz de transformación homogénea ${}^0A_4$ corresponde a la postura del robot vista en la Figura 7.

La ecuación 71 muestra las componentes del vector que representa la posición del efector final.

$$P = \begin{bmatrix} 0 \\ 279.83 \\ 135.5 \\ 1 \end{bmatrix} \tag{71}$$

Con las ecuaciones de la Tabla 3 se resuelven las diferentes coordenadas articulares presentes en el robot. La primera articulación q_1 se resuelve conociendo los valores de las variables que intervienen en la ecuación 42.

Donde:

$P_x = 0$

$P_y = 279.83$

Resolviendo la ecuación 42, la coordenada q_1 tomará el valor del resultado de la ecuación 72.

$$q_1 = atan2\left(\frac{0}{279.83}\right) \therefore \quad \boxed{q_1 = 0} \tag{72}$$

Ahora se debe calcular las componentes de X_4 de la ecuación 46 para determinar el valor del punto de la muñeca P_m y continuar con la solución de las articulaciones.

Con un ángulo $\propto = 0$ respecto al sistema de la base del robot se tiene la ecuación 73.

$$\begin{bmatrix} X_{4_x} \\ X_{4_y} \\ X_{4_z} \end{bmatrix} = \begin{bmatrix} \cos(0) & \sin(0) \\ \cos(0) & \cos(0) \\ \sin(0) & \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{73}$$

Se calcula ahora las componentes del punto de la muñeca definidas en la ecuación 74.

$$P_m = \begin{bmatrix} 0 \\ 279.83 \\ 135.5 \end{bmatrix} - 65 \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \therefore \quad \boxed{P_m = \begin{bmatrix} 0 \\ 214.83 \\ 135.5 \end{bmatrix}} \qquad (74)$$

Siguiente paso, calcular el valor de m en la ecuación 75.

$$m = \sqrt{(0)^2 + (214.83)^2} \therefore \quad \boxed{m = 214.83} \qquad (75)$$

Para obtener las coordenadas articulares de *q3* se determina el valor de J con la ecuación 76.

$$J = \sqrt{(214.83)^2 + (135.5 - 135.5)^2} \therefore \quad \boxed{J = 214.83} \qquad (76)$$

El resultado de J y m tienen el mismo valor porque su cálculo se basa en un movimiento articular conocido para el cual q_1 y q_2 son igual a cero, por lo tanto, al observar la **¡Error! No se encuentra el origen de la referencia.** se puede interpretar gráficamente e stos resultados.

Con los valores de la longitud del segundo eslabón L_2, el tercer eslabón L_3 y la magnitud del segmento J se calcula q_3 de la ecuación 77 a la 78.

$$q_3 = \text{atan2}\left(\sqrt{1 - \left(\frac{214.83^2 - 120.23^2 - 94.6^2}{(2)(120.23)(94.6)}\right)^2}, \frac{214.83^2 - 120.23^2 - 94.6^2}{(2)(120.23)(94.6)}\right) \qquad (77)$$

Resolviendo:

$$\boxed{q_3 = 0}$$

$$q_3 = \text{atan2}\left(\sqrt{1-1^2}, 1\right) \therefore \quad (78)$$

Con el valor de q_3 se calcula la coordenada articular q_2 con las ecuaciones 79 a la 80.

$$q_2 = \text{atan2}(135.5 - 135.5, 214.83) - \text{atan2}(94.6\sin(0), 120.23 + (94.6\cos(0))) \quad (79)$$

Resolviendo:

$$q_2 = \text{atan2}(0, 214.83) - \text{atan2}\big(0, 120.23 + (94.6)\big) \therefore \boxed{q_2 = 0} \quad (80)$$

Para calcular el valor de q_4 primero se deben encontrar los valores de X_3 y Y_3 para aplicar la ecuación 70. Las variables X_3 y Y_3 se calculan aplicando la matriz de transformación homogénea ${}^0A_3$ con las coordenadas q_1, q_2 y q_3 en la ecuación 81.

$${}^0A_3 = \begin{vmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 135.5 \\ 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} 1 & 0 & 0 & 120.23 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} 1 & 0 & 0 & 94.6 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \therefore {}^0A_3 = \begin{vmatrix} 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 350.33 \\ 0 & 0 & 0 & 1 \end{vmatrix} \quad (81)$$

Conociendo los de ${}^0\text{A}_3$ se pueden determinar las componentes de X_3 - Y_3 correspondientes a la primera y segunda columna de la matriz de la ecuación 81. Las componentes de X_3 - Y_3 se muestran en las ecuaciones 82 y 83 respectivamente.

$$X_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad (82)$$

$$Y_3 = \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} \tag{83}$$

Teniendo todas las componentes necesarias para calcular la coordenada articular q_4 aplicando la ecuación 69, se procede a reemplazar X_3, X_4, y Y_3 en la ecuación 84.

$$q_4 = \operatorname{atan2}\left(\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}\right) \therefore \quad \boxed{q_4 = 0} \tag{84}$$

Los resultados de aplicar la cinemática inversa a partir de la posición del efector final coinciden con los valores de las coordenadas articulares que se tomaron en la posición cero del robot.

La posición cero o inicio del robot Raplim para cada articulación equivale a cero *(0)* o está en reposo, así que los resultados de aplicar la cinemática inversa a este sistema se ven reflejado en la solución de las ecuaciones 72, 80, 78 y 84 que corresponde respectivamente a las articulaciones q_1, q_2, q_3 y q_4 de robot Raplim. De esta manera se demuestra que el modelo se calculó de manera correcta y se puede usar para determinar la posición de las articulaciones a partir de la localización del efector final.

Una vez definido el modelo cinemático se implementa en un programa matemático cada uno de los parámetros y ecuaciones que hacen parte de la solución del sistema, que permite evaluar eficientemente los resultados del modelo en poco tiempo. Además, el

programa permite representar de forma gráfica el diagrama de alambres para observar el comportamiento de cado uno de los eslabones al ingresar la posición del efector final o las coordenadas articulares del robot como se muestra en la Figura 15.

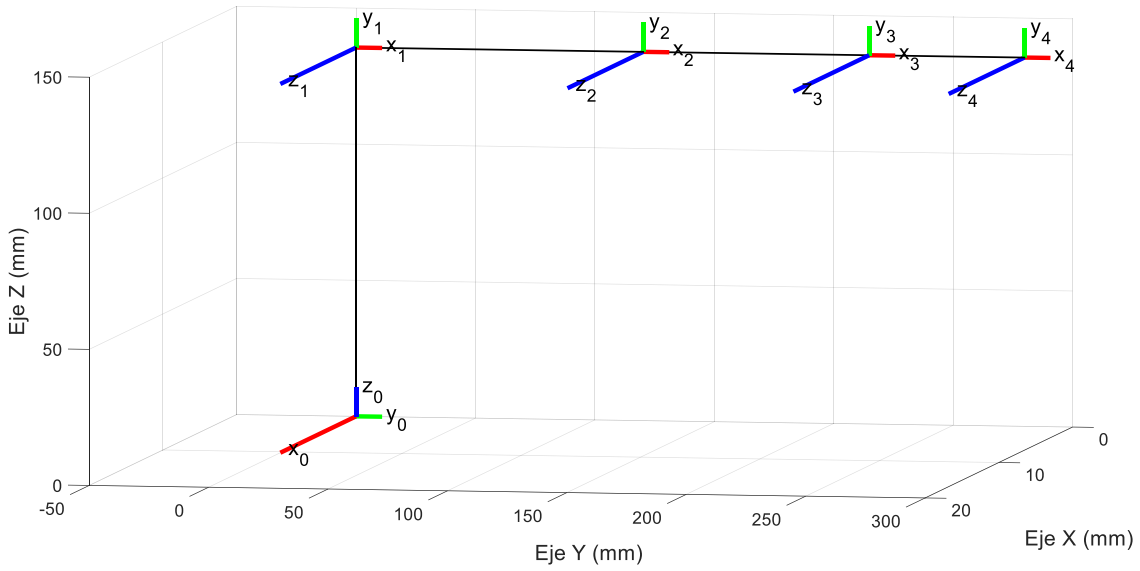


Figura 15. Diagrama de alambres del robot Raplim representando en el programa matemático.

Calculada y representada la cinemática del robot se procede al análisis del control por guiado en el siguiente capítulo.

4. PROGRAMACIÓN POR GUIADO GESTUAL

La programación por guiado se basa en mostrarle al robot los movimientos con los que se genera una trayectoria para que él pueda imitarla posteriormente de forma autónoma. La programación por guiado gestual se apoya en el anterior concepto sumado a la colaboración de las nuevas tecnologías para la captura de patrones de movimiento que serán evaluados mediante un algoritmo que respalde la captura del movimiento mediante la simulación para ser transformadas en trayectorias que el robot imite de forma segura. La Figura 16 muestra el diagrama de bloques que detalla de forma general el proceso para capturar los movimientos del operador a través del Kinect para la adquisición de datos. Para programar un robot por guiado gestual se deben llevar los datos capturados al espacio de la tarea mediante el uso de matrices de transformación homogéneas y los sistemas coordenados que intervienen en el entorno de trabajo, así como definir aquellos movimientos que se toman como referencia para iniciar y finalizar la adquisición de datos.

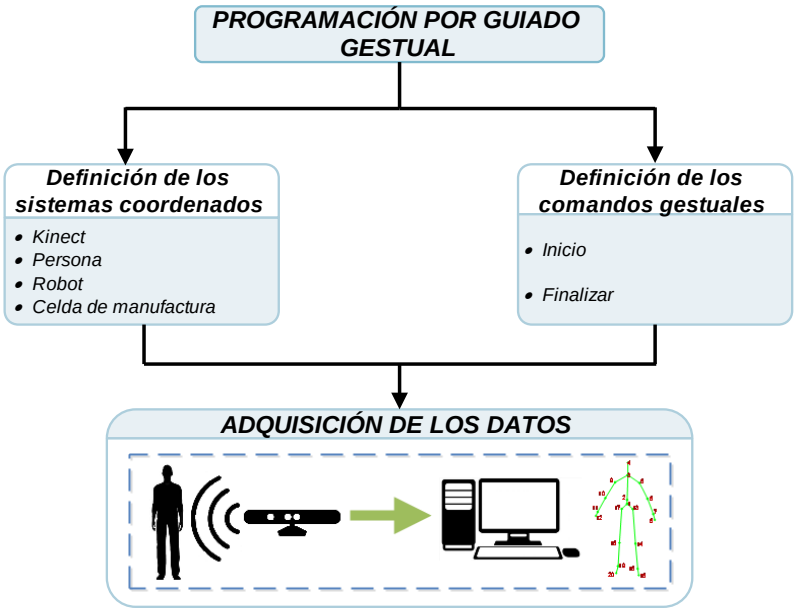


Figura 16. Diagrama de bloques de la captura del movimiento de un operador.

La visión artificial es una herramienta tecnológica que permite la programación de robots por guiado gestual a través de una cámara de luz estructurada que escanea el entorno (objetos, formas, personas, etc.) a través de un dispositivo (Kinect) utilizando un patrón de luz, capturando así los movimientos de un operador que pueden ser reproducidos por el robot al ser transformados en una trayectoria.

A continuación, se describe el uso de la técnica de visión artificial para la captura de los movimientos del operador.

4.1 Visión artificial

El control por guiado gestual para el robot antropomórfico Raplim basado en técnicas de visión artificial necesita de una cámara de luz estructurada capaz de reconocer gestos, movimientos, objetos e imágenes con las que se puede desarrollar una técnica de programación de trayectorias sin tener contacto directo con el robot o el ordenador.

Con ayuda de la cámara de luz estructurada **(**Kinect**)** se logran identificar las partes del cuerpo de cualquier operador para transformar sus movimientos en comandos, para luego ser reproducidos con ayuda de un algoritmo a través de una plataforma que sirva como interfaz de usuario. La cámara de luz estructurada que se usa es el sensor Kinect V1 de *Microsoft Research,* con el que se hace el seguimiento del cuerpo del usuario (esqueleto) o *Skeleton Tracking* en tiempo real.

Lo que el Kinect captura es una serie de puntos específicos que corresponden a cada articulación del cuerpo y con los que se logran asignar comandos cuando se registre el movimiento de alguna de las articulaciones del cuerpo, la Figura 17 ilustra el *Skeleton Tracking* que captura el Kinect y la silueta de fondo es para identificar las partes del cuerpo. Cabe aclarar que la Figura 17 se presenta desde la perspectiva del Kinect, es decir el operador se encuentra frente a él y la captura ilustra una imagen tipo espejo a la real.

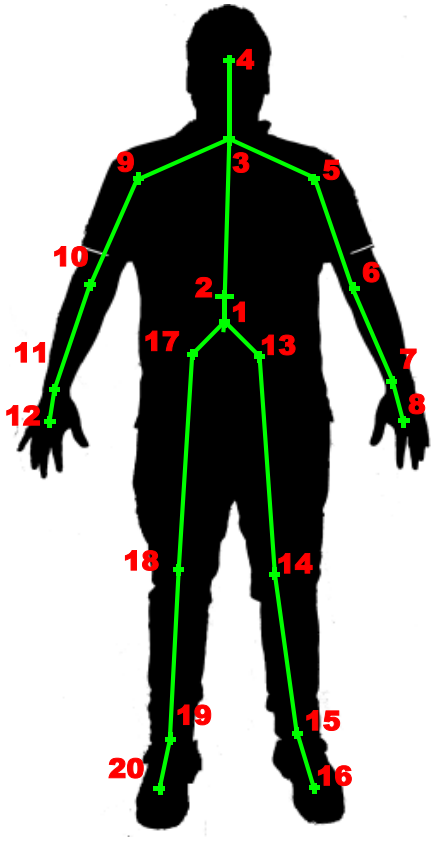


Figura 17. Skeleton Tracking.

En la Tabla 4 se especifica la equivalencia que existe entre los valores de los puntos reconocidos por el sensor Kinect de las articulaciones de una persona representadas en la

Figura 17, con los que se puede formar el esqueleto en un espacio tridimensional de la interfaz de usuario.

Tabla 4.
Datos de referencia del cuerpo generados por el Kinect.

Articulación	Dato de referencia
Centro de la cadera	**1**
Columna vertebral	**2**
Centro de los hombros	**3**
Cabeza	**4**
Hombro izquierdo	**5**
Codo izquierdo	**6**
Muñeca izquierda	**7**
Mano izquierda	**8**
Hombro derecho	**9**
Codo derecho	**10**
Muñeca derecha	**11**
Mano derecha	**12**
Cadera izquierda	**13**
Rodilla izquierda	**14**
Tobillo izquierdo	**15**
Pie izquierdo	**16**
Cadera derecha	**17**
Rodilla derecha	**18**
Tobillo derecho	**19**
pie derecho	**20**

Nota: los datos de la tabla son tomados del código publicado en el centro de ayudas de *MathWorks* para el reconocimiento del esqueleto por medio de *Kinect*.

En el control por guiado, es necesario que un operador le muestre al robot el camino que debe recorrer manipulándolo de forma directa como se define en la programación por guiado pasivo del capítulo 3. De acuerdo a lo anterior se propone homologar la metodología de programación por guiado, pero esta vez aplicando la visión artificial para la captura de los movimientos del operador y programarlos como trayectorias, sin que haya un contacto directo con el robot, es decir, que mediante poses especificas el operador sea capaz de iniciar, capturar y finalizar la toma de datos sin necesidad de interactuar con el robot o el equipo donde se procesa la información.

Los datos capturados a través del Kinect manejan las distancias en escala de metros (m) por lo que estas medidas deben ajustarse a los cálculos de la cinemática del robot previamente determinados en el capítulo 4 donde la escala se toma en milímetros (mm). Las matrices calculadas de la cinemática del robot, las constantes de trabajo y las coordenadas del Kinect se programan en el software matemático para su interpretación. La ecuación 85 escala las medidas de las coordenadas dadas por el Kinect a milímetros.

$$Punto\ leido = \left[(p_x * K), (p_y * K), (p_z * K)\right] \tag{85}$$

Donde:

$K = 1000$

4.2 Sistemas coordenados

Para simular la trayectoria del robot en un entorno virtual se deben considerar algunas características, como: las dimensiones del robot, la ubicación, la orientación de la persona, la posición y orientación del Kinect. Definido el sistema coordenado del espacio de trabajo (salón) o celda de manufactura *(S0)* como punto de referencia se ubican los objetos de interés como: el Kinect *(Sk)*, el robot *(SR)*, la persona *(SB)* y sus características para visualizar de manera virtual el ambiente de trabajo real como en la Figura 18.

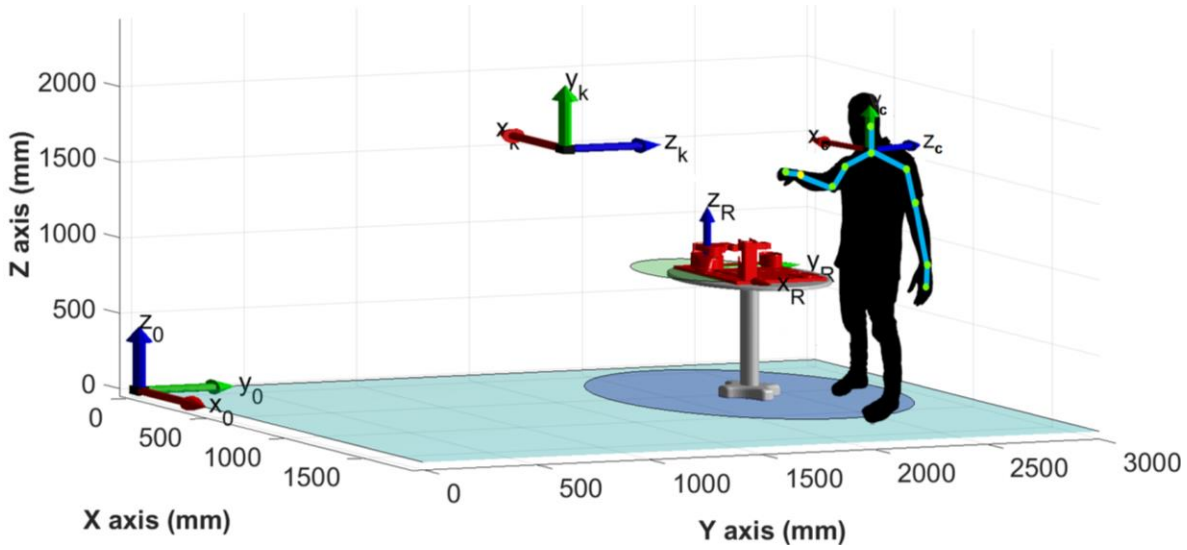


Figura 18. Entorno virtual con los objetos de interés.

4.2.1 Kinect

La Figura 19 muestra el esqueleto del operador visto por Kinect en el entorno virtual de manera horizontal (acostado) y debe verse vertical (de pie), por lo que se debe realizar una transformación de los puntos del cuerpo tomados por Kinect al sistema coordenado de la

celda de manufactura donde se encuentra ubicados los demás objetos para que se observe como en la

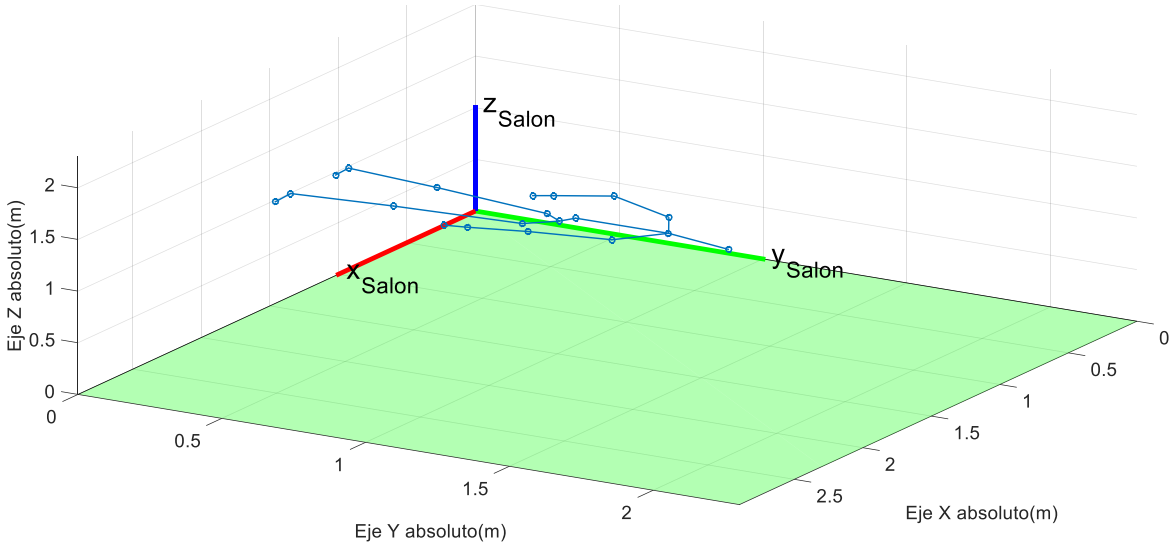


Figura 19. Cuerpo visto de forma horizontal.

La Figura 20 muestra una representación del sistema coordenado del *Kinect* con respecto al espacio de trabajo (celda de manufactura).

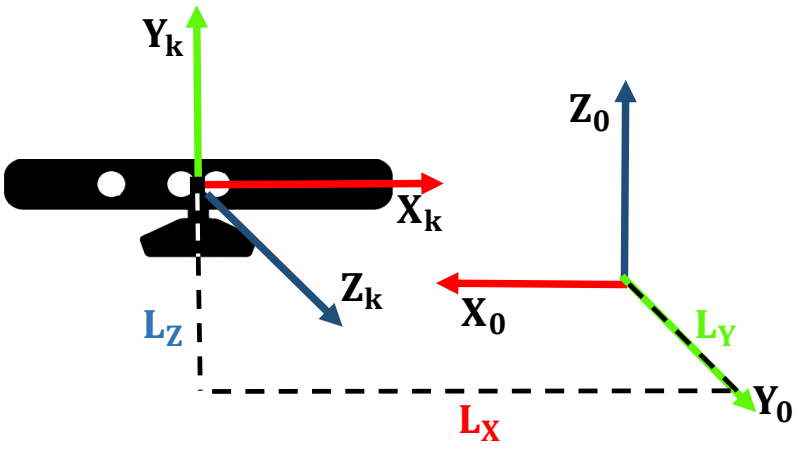


Figura 20. Sistema coordenado del Kinect.

La ecuación 86 muestra la matriz de transformación homogénea que lleva los puntos del cuerpo de la persona tomados por Kinect al sistema coordenado de la celda de manufactura.

$$ {}^{0}A_K = \begin{vmatrix} -1 & 0 & 0 & L_x \\ 0 & 0 & 1 & L_y \\ 0 & 1 & 0 & L_z \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{86} $$

4.2.2 Persona

En el cuerpo del operador debe ubicarse un sistema de referencia, definiendo como origen el punto en el centro de los hombros. Esta ubicación del sistema coordenado se escoge de forma estratégica para determinar con precisión la posición de las articulaciones (extremidades superiores) con las que se pretende generar la trayectoria tal como se observa en la Figura 21. Para la generación de la trayectoria se selecciona una articulación que el operador pueda manipular con comodidad y precisión, como los son los extremos de los brazos (muñecas), por lo que se escogen las muñecas derecha e izquierda como punto de referencia para generar la trayectoria y los comandos respectivamente.

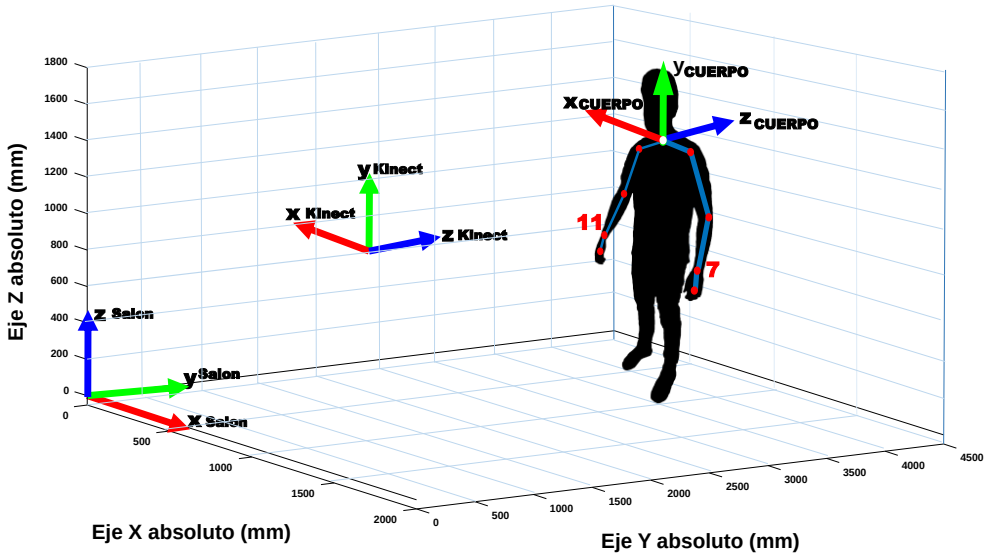


Figura 21. Sistema de referencia en el centro de los hombros.

El sistema coordenado del cuerpo está indicado por Kinect y tiene su origen en el centro de los hombros, a medida que el operador se mueve cambia la posición del sistema del cuerpo manteniendo la misma orientación del sistema coordenado del Kinect con respecto a la celda de manufactura (sistema coordenado).

En la Figura 22 se dibuja el sistema de referencia del cuerpo en el centro de los hombros con respecto a la celda de manufactura y se calcula su posición al aplicar la matriz de transformación ${}^{0}A_{k}$ al punto P_{k3} tomado por Kinect.

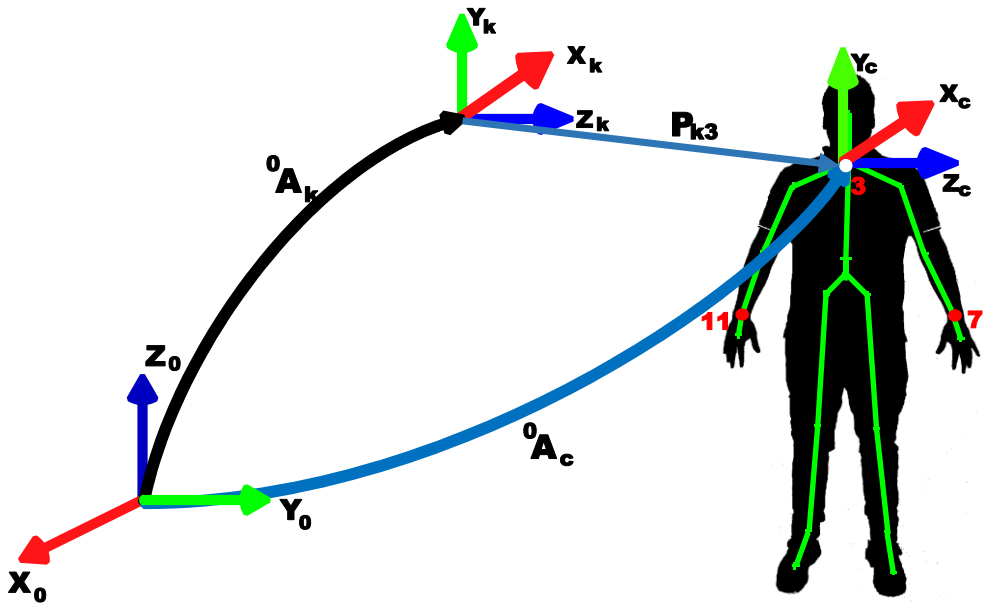


Figura 22. Sistema de referencia.

Las ecuaciones de la 87 a la 89 expresan matemáticamente la solución del sistema de referencia ubicado en el cuerpo con respecto a la celda de manufactura ilustrado en la Figura 23.

Se resuelve P_{ch} en función de la celda de manufactura y el punto del centro de los hombros con respecto a Kinect.

$$P_{ch} = {}^{0}A_{K} P_{k3} \tag{87}$$

El producto da como resultado un vector columna que representa la posición del centro de los hombros con respecto a la celda de manufactura.

$$P_{ch} = \begin{bmatrix} P_{ch(x)} \\ P_{ch(y)} \\ P_{ch(z)} \\ 1 \end{bmatrix} \tag{88}$$

En la ecuación 89 se define la orientación y la posición del sistema coordenado del cuerpo.

$$ {}^{0}A_C = \begin{vmatrix} -1 & 0 & 0 & P_{ch(x)} \\ 0 & 0 & 1 & P_{ch(y)} \\ 0 & 1 & 0 & P_{ch(z)} \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{89} $$

Donde:

P_{ch} = Es el vector que representan las componentes de posición del centro de los hombros con respecto al sistema de referencia del salón.

${}^{0}A_C$ = Sistema coordenado del cuerpo.

Definido el sistema coordenado del cuerpo, se verifica que los movimientos no se reproducen de forma natural, si el cuerpo está girando todo su sistema coordenado también debería hacerlo, por lo que se debe aplicar una matriz de transformación homogénea para hacer la rotación del sistema del cuerpo sobre el eje Y_c, pero antes se debe determinar la magnitud del giro. Esté ángulo de giro se calcula con las componentes del punto del hombro derecho con respecto a la celda de manufactura (P_{hd}) y el centro de los hombros (P_{ch}). La Figura 23 muestra el diagrama de referencia y los vectores del cuerpo desde la vista superior para determinar el ángulo de giro. El punto P_{ch} es el origen del sistema coordenado del cuerpo, los vectores X_c y Z_c representan los ejes de referencia para el cuerpo en reposo, los vectores X'_c y Z'_c representan los ejes de referencia cuando el cuerpo se gira un ángulo θ.

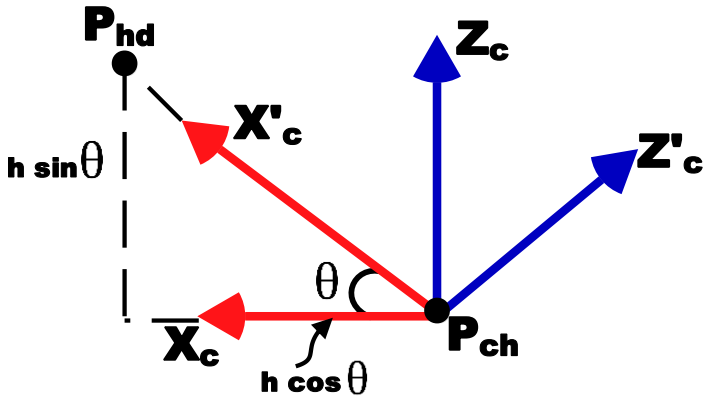


Figura 23. Diagrama de referencia.

La orientación del sistema coordenado del cuerpo se determina mediante el ángulo de rotación calculado a partir de la variación en las coordenadas del centro de los hombros y el hombro derecho del cuerpo de la persona dadas por el Kinect con respecto al sistema de referencia de la celda de manufactura, resueltas en las ecuaciones 90 a la 93.

Las coordenadas del hombro derecho con respecto a la celda de manufactura se representan con P_{hd}.

$$P_{hd} = {}^{0}A_K P_{k9} \tag{90}$$

Para obtener el ángulo θ se deben resolver las funciones trigonométricas sin y cos en función de P_{ch} y P_{hd}.

$$\sin\theta = P_{ch}(\mathrm{y}) - P_{hd}\ (\mathrm{y}) \tag{91}$$

$$\cos\theta = P_{ch}(\mathrm{x}) - P_{hd}\ (\mathrm{x}) \tag{92}$$

La solución del ángulo de giro se representa así:

$$\theta = tan^{-1}\left[\frac{\sin\theta}{\cos\theta}\right] \;\therefore\; \theta = tan^{-1}\left[\frac{P_{\mathrm{ch(y)}} - P_{\mathrm{hd(y)}}}{P_{\mathrm{ch(x)}} - P_{\mathrm{hd(x)}}}\right] \tag{93}$$

Donde:

P_{ch} = Posición del centro de los hombros con respecto a la celda de manufactura.

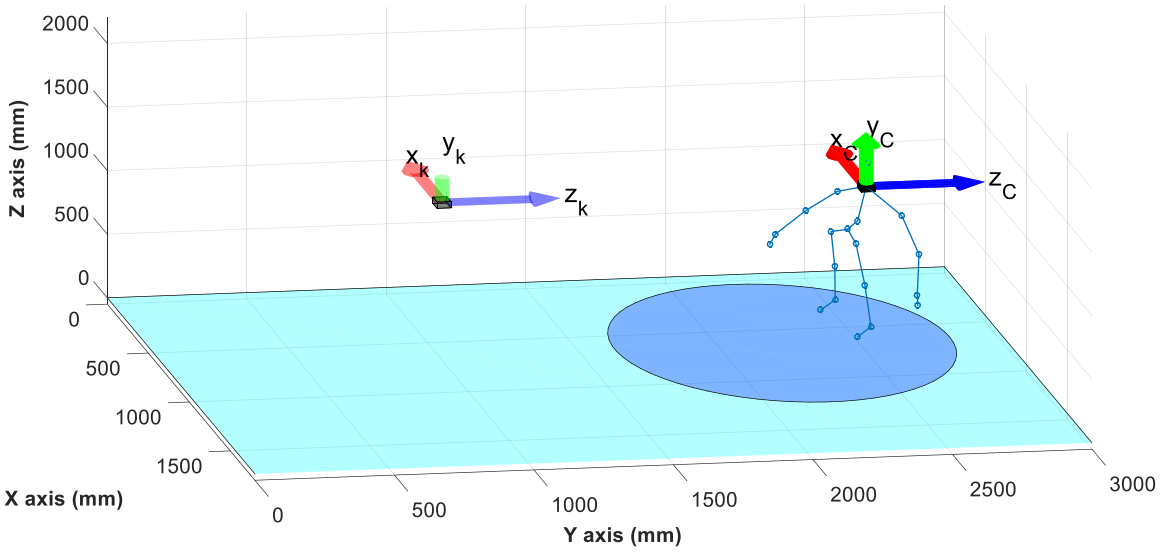
P_{hd} = Posición del hombro derecho con respecto a la celda de manufactura.

θ = Angulo de giro.

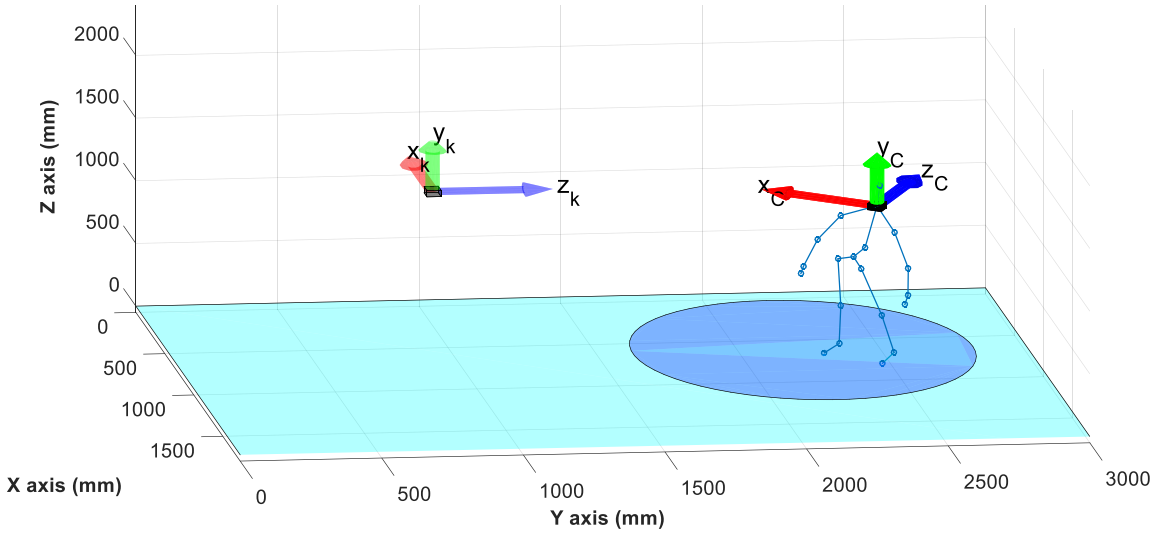
Una vez obtenido el ángulo de giro θ se modifica el sistema coordenado del cuerpo multiplicándolo por la matriz de rotación con respecto al eje Y_c $\left(Roty(\theta)\right)$. Este arreglo matemático representado en la ecuación 94 permite que a medida que se gire el cuerpo rote también el sistema de referencia.

$$^{0}A_{CR} = {}^{0}A_{C}Roty(\theta) \tag{94}$$

La Figura 24 muestra el sistema coordenado del cuerpo en el entorno virtual donde se evidencia a) Sistema sin rotación y b) Sistema rotado.



a)



b)

Figura 24. Sistema coordenado del cuerpo en el entorno virtual a) Sin rotar y b) Rotado.

Otro problema fue definir la posición del punto de la muñeca izquierda en el sistema coordenado del cuerpo, ya que este punto es tomado del sistema coordenado de Kinect. La solución se obtiene a partir de las matrices de transformación obtenidas en las ecuaciones 86

y 94 que corresponden a ${}^{0}A_{K}$ y ${}^{0}A_{CR}$ respectivamente, las ecuaciones 95 y 96 representan los cálculos que se deben aplicar para determinar el punto de la muñeca.

$$P_{mi} = {}^{0}A_{K} * P_{K7} \tag{95}$$

$$P_{Cmi} = ({}^{0}A_{CR})^{-1} * P_{mi} \tag{96}$$

Donde:

${}^{0}A_{CR}$= Matriz de transformación de la celda de manufactura al cuerpo

P_{mi} = Punto de la muñeca izquierda con respecto a la celda de manufactura.

p_{Cmi} = Componentes de la mano izquierda con respecto al sistema de referencia del cuerpo M_{c}.

P_{K7} = Punto de la muñeca izquierda con respecto a Kinect.

La muñeca izquierda es la encargada de indicar las posiciones con las que se generan las condiciones de los comandos, en la Figura 25 se muestra el punto de la muñeca izquierda con respecto al sistema de referencia del cuerpo.

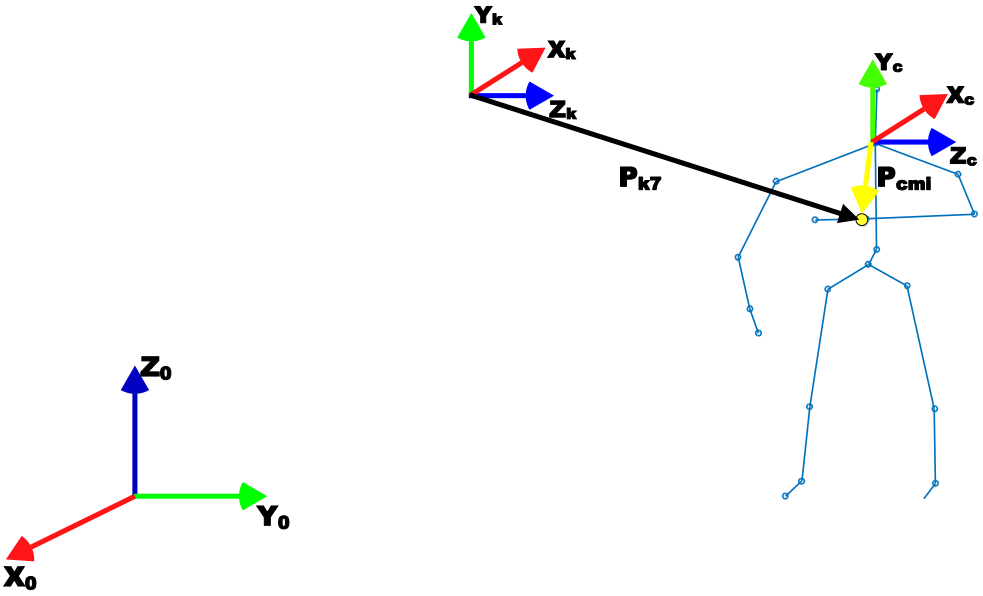


Figura 25. Punto de la muñeca izquierda.

Con estos arreglos se asegura que cualquier persona que este frente al Kinect pueda ejecutar los comandos a través de los movimientos de la muñeca izquierda sin que se necesiten hacer cambios en el programa (código fuente) cada vez que lo manipule un usuario diferente, ya que la condición de los comandos se programa para ajustarse automáticamente a la posición y orientación del cuerpo de la persona que lo va a operar.

4.2.3 Robot

Para generar la trayectoria se escoge como punto de referencia la muñeca derecha, ya que con la muñeca izquierda se indican los comandos de inicio, pausa y finalización del programa. Como el punto de la muñeca derecha esta referenciado al sistema coordenado del Kinect y con ella se va a generar la trayectoria del robot está debería referenciarse con

respecto al sistema del robot, la Figura 26 ilustra los cambios de base para generar esta transformación.

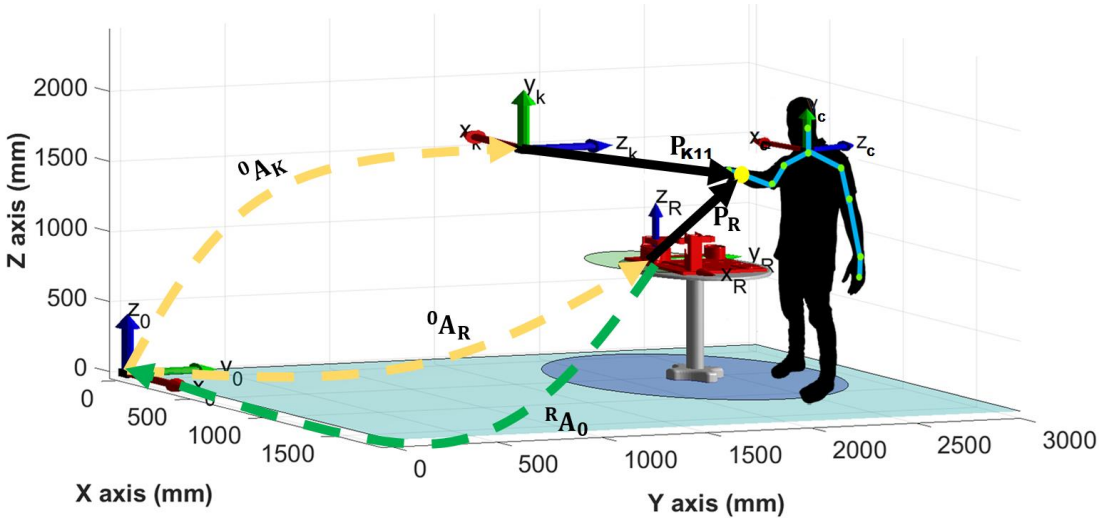


Figura 26. Sistema coordenado de la muñeca derecha.

La Figura 27 muestra el punto de la muñeca derecha P_R con respecto al robot y a P_{k11} que es el punto de la muñeca derecha referenciado al sistema del Kinect. Para llevar P_{k11} al sistema coordenado del robot hay que aplicar una matriz de transformación ${}^RA_k$ teniendo en cuenta que el sistema coordenado del Kinect y el del robot son sistemas fijos. Las ecuaciones 86 y 97 son las matrices de transformación que se utilizan para calcular ${}^RA_k$.

$$ {}^0A_R = \begin{vmatrix} 1 & 0 & 0 & L_{Rx} \\ 0 & 1 & 0 & L_{Ry} \\ 0 & 0 & 1 & L_{Rz} \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{97} $$

Donde L_{Rx}, L_{Ry} y L_{Rz} son las coordenadas que indican el origen del sistema coordenado del robot con respecto a la celda de manufactura.

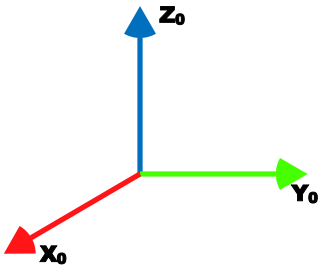
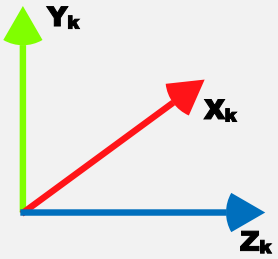
${}^{0}A_{R}$ es la matriz del robot con respecto a la celda de manufactura y su inversa es la que permite construir el camino del punto P_{k11} al sistema del robot. Las ecuaciones 98 y 99 se utilizan para determinar el punto P_R y la matriz de transformación ${}^{R}A_{k}$.

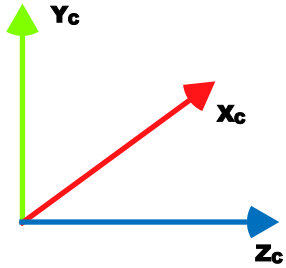
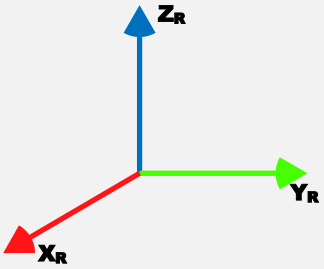
$$ {}^{R}A_{K} = {}^{0}A_{R}{}^{-1}{}^{0}A_{k} \tag{98} $$

$$ \mathrm{P}_{R} = {}^{R}A_{k}P_{k11} \tag{99} $$

Realizados los cálculos y transformaciones anteriores se crea la Tabla 5 que resume los sistemas coordenados de los objetos involucrados en el desarrollo de este proyecto.

Tabla 5.
Ecuaciones de los sistemas coordenados.

SISTEMA COORDENADO	DEFINICIÓN	MATRIZ DE TRANSFORMACIÓN
	Celda de manufactura o salón	$A_0 = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$
	Kinect	${}^{0}A_{K} = \begin{vmatrix} -1 & 0 & 0 & L_x \\ 0 & 0 & 1 & L_y \\ 0 & 1 & 0 & L_z \\ 0 & 0 & 0 & 1 \end{vmatrix}$

	Persona	$^{0}A_{CR} = {}^{0}A_{C}Roty(\theta)$
	Robot	$^{0}A_{R} = \begin{vmatrix} 1 & 0 & 0 & L_{Rx} \\ 0 & 1 & 0 & L_{Ry} \\ 0 & 0 & 1 & L_{Rz} \\ 0 & 0 & 0 & 1 \end{vmatrix}$

Nota: Los sistemas se toman con referencia a la celda de manufactura o salón.

4.3 Definición de los comandos gestuales

Un comando es una orden que se da para generar una acción donde los gestos son utilizados para programar condiciones que luego son ejecutadas, apoyándose en la característica que ofrece el Kinect de reconocer las partes y articulaciones del cuerpo. Para desarrollar este proyecto se elabora una estrategia que facilite la interpretación de poses para programar los comandos que inicien, detengan y finalicen la toma de datos, sin tener contacto directo con el robot ni el ordenador donde se procesan los datos que envía el Kinect para mostrar a través de la interfaz de usuario la simulación de la trayectoria.

Las poses para convertir los movimientos en comandos gestuales son posiciones cómodas para la persona que está ejecutando la acción y no tan usuales, como estar de pie, para evitar falsas lecturas o errores en medio del proceso. Estas precauciones se toman debido a que el registro de datos para generar la trayectoria del robot requiere un tiempo considerable. La Figura 28(a) muestra la pose que debe asumir la persona para dar inicio al sistema y la Figura 27(b) muestra la pose para dar fin a la ejecución del programa.

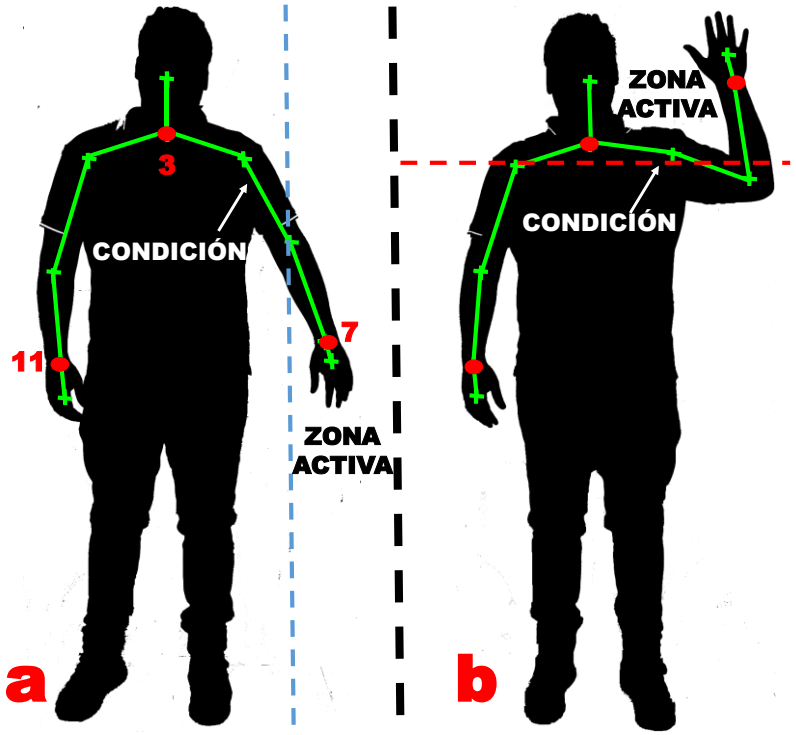


Figura 27. (a) Pose para el comando iniciar (b) Pose para el comando finalizar registro.

El comando de inicio tiene *2* condiciones, la primera es que la muñeca izquierda debe alejarse a una distancia considerable de la superficie del cuerpo, dando paso a la segunda condición, que es mantener esta posición mínimo durante 5 segundos para iniciar el registro de la trayectoria que valla a ejecutar la muñeca derecha, pero sin interrumpir la primera

condición, es decir, la posición de la muñeca izquierda no debe cambiar durante el registro de la trayectoria que está siendo ejecutada por la muñeca derecha. Si la posición de la muñeca izquierda cambia, se pausa el registro e inicia nuevamente en la posición en la que se encuentre la muñeca derecha.

Una vez ejecutado los movimientos para la trayectoria a través del movimiento de la muñeca derecha se debe finalizar el registro, esto se hace elevando la muñeca izquierda por encima de la cabeza, es decir, la muñeca izquierda debe moverse hacia la parte superior de la cabeza para finalizar el proceso de registro, ejecutando el comando de finalización.

Con ayuda de la interpretación de estas poses se consigue minimizar la interacción directa entre el operador, el robot y el equipo de cómputo utilizado para la adquisición, evaluación y simulación de la trayectoria antes de cargarla al robot real. La realidad aumentada también juega un papel muy importante en el desarrollo de este proyecto, pero de eso se habla más adelante.

En la Figura 28 se detalla el diagrama de flujo del algoritmo utilizado para la identificación de las poses a través del Kinect con las que se programan los comandos de inicio y finalizar para la toma de datos.

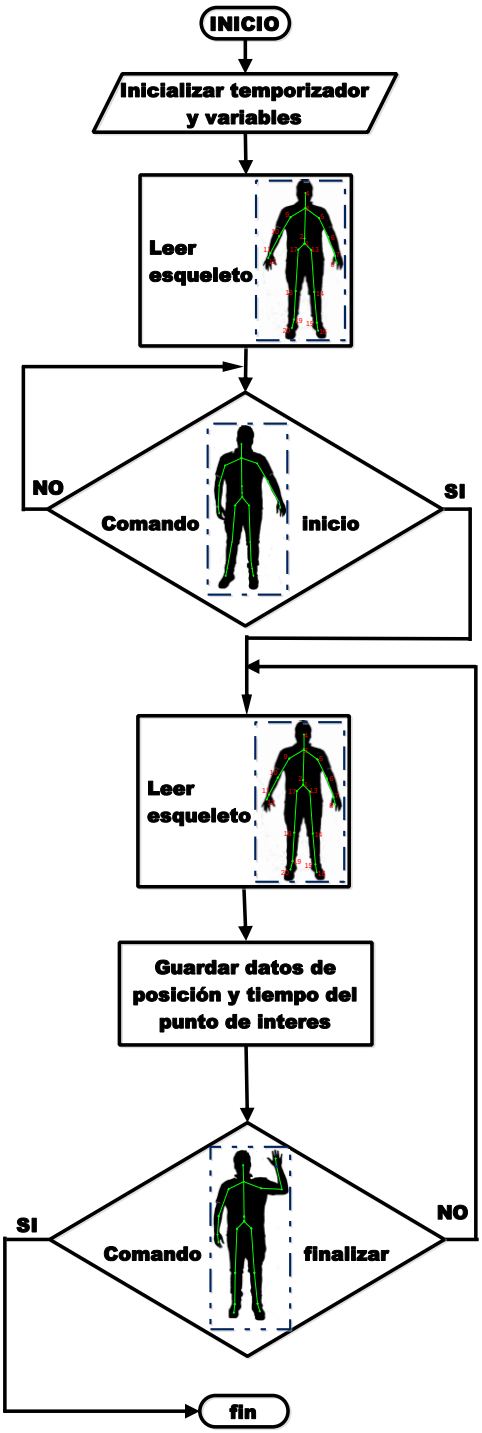


Figura 28. Diagrama de flujo para la adquisición de datos a través de comandos gestuales.

Para reducir la carga en el procesamiento de datos al momento de capturar los movimientos solo se leen los puntos de las articulaciones de la parte superior del cuerpo del operador, es decir, los brazos, los hombros y la cabeza, ya que el resto del cuerpo no aporta ningún dato relevante en el algoritmo de generación de la trayectoria. La Figura 29 muestra la silueta con los puntos que van a ser evaluados en el programa.

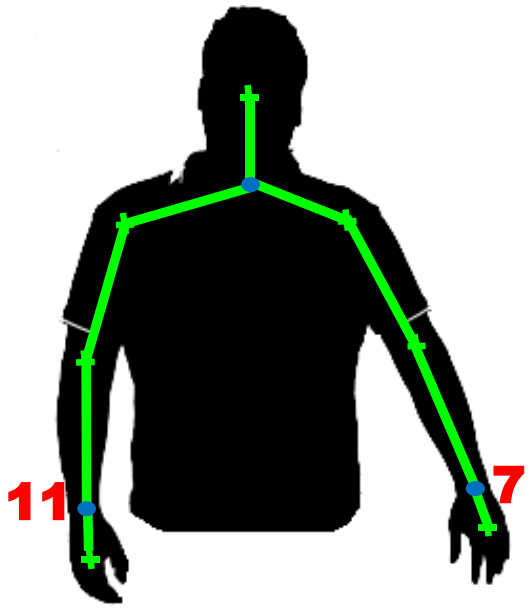


Figura 29. Puntos del cuerpo evaluados por el Kinect para desarrollar los comandos y su trayectoria.

5. INTERFAZ DE USUARIO

La interfaz de usuario permite la interacción entre la persona y el algoritmo, para facilitar la comunicación sin que se tenga un conocimiento técnico del lenguaje de programación o el manejo de la plataforma, es decir, es la parte visual e intuitiva que se comunica con el usuario. Para facilitar la interacción de terceros con el robot, se desarrolla una interfaz que permite visualizar los efectos producidos por las posturas y movimientos del operador en el comportamiento del robot y mostrar los resultados de la simulación.

La interfaz se desarrolla con un software matemático que cuenta con las herramientas necesarias para realizar la plataforma interactiva y a la par hacer los cálculos necesarios para dar solución al algoritmo de planificación de trayectorias. En el espacio de trabajo el usuario puede observar el entorno virtual, ajusta la posición del área para la toma de datos, seleccionar el rango de puntos usados en la trayectoria, iniciar y finalizar la adquisición de datos, ajustar la vista y dimensiones del entorno, conocer variables como cantidad de datos adquiridos, puntos que están fuera y dentro del espacio de trabajo del robot y los tiempos de simulación en la trayectoria, también puede implementar el algoritmo de control cinemático, optimizar la trayectoria, hacer la simulación del robot recorriendo la trayectoria y por último, puede guardar la trayectoria o cargar una previamente almacenada.

La Figura 30 presenta la interfaz de usuario principal donde se evidencian el área de trabajo para realizar las anteriores instrucciones.

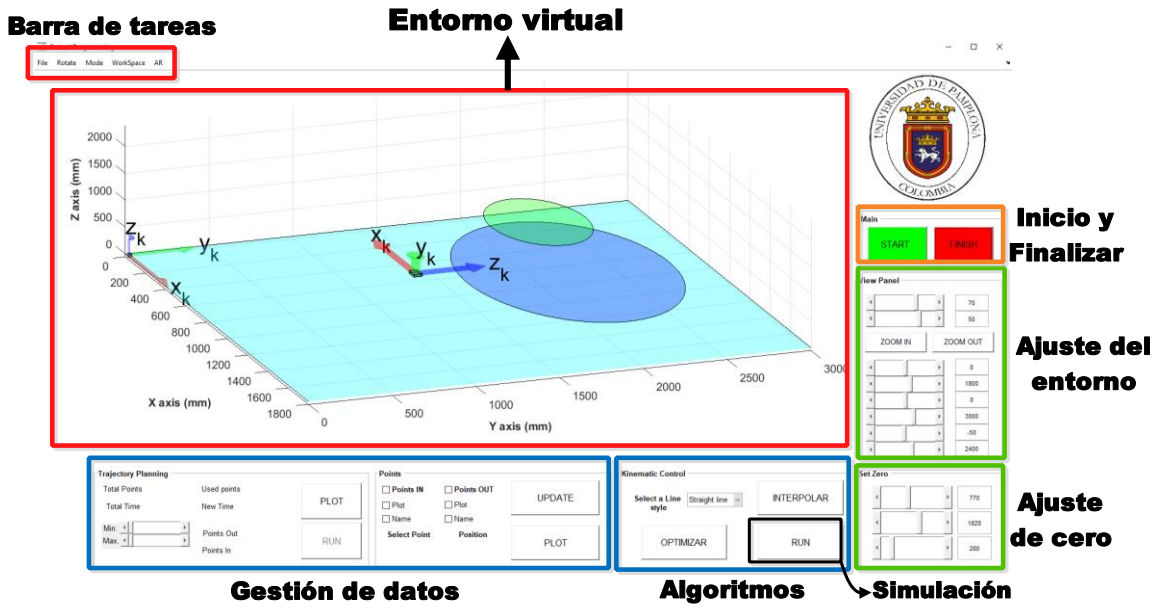


Figura 30. Interfaz de usuario principal.

A continuación, se explican algunas de las funciones de la interfaz de usuario principal.

5.1 Gestión de datos

En el manejo de datos se planea la trayectoria *("Trajectory Planning")*, es decir, se seleccionan los puntos específicos sobre los cuales se genera la trayectoria del robot, creados a partir de la captura del movimiento. Al finalizar la captura de datos con el botón rojo *"FINISH"* se genera una matriz que almacena las posiciones de la muñeca derecha (Trayectoria) y el tiempo de ejecución para aplicarle la cinemática inversa, teniendo en cuenta las restricciones en cada posición almacenada en la matriz con el objetivo de definir los puntos fuera y dentro del espacio de trabajo del robot Raplim. En la Figura 31 se detalla el diagrama de flujo que para el algoritmo que evalúa la matriz de datos.

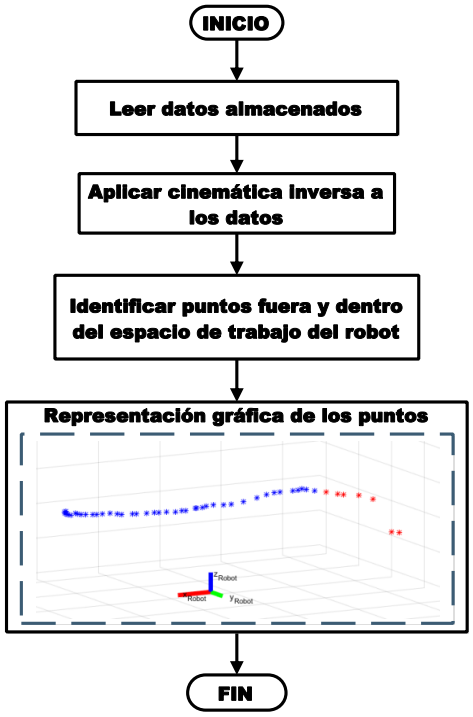


Figura 31. Diagrama de flujo para evaluar la trayectoria.

A la vez que se capturan los puntos que representan el movimiento del operador también se guarda el instante de tiempo de cada uno de ellos, el cual se utiliza en el control cinemático mencionado en el ítem 5.4 de este capítulo. La estructura de la matriz que almacena la posición con respecto al sistema coordenado del robot e instantes de tiempo de los puntos que pertenecen a los movimientos del operador se detalla en la ecuación 100.

$$M_d = \begin{bmatrix} P_{rx} \\ P_{ry} \\ P_{rz} \\ P_{rt} \end{bmatrix} = \begin{bmatrix} P_{rx1} & P_{rx2} & P_{rx3} & P_{rx4} & \dots & P_{rxn} \\ P_{ry1} & P_{ry2} & P_{ry3} & P_{ry4} & \dots & P_{ryn} \\ P_{rz1} & P_{rz2} & P_{rz3} & P_{rz4} & \dots & P_{rzn} \\ P_{rt1} & P_{rt2} & P_{rt3} & P_{rt4} & \dots & P_{rtn} \end{bmatrix} \quad (100)$$

Donde:

M_d = Es la matriz de los datos capturados.

P_r = Es el vector que almacena la posición con respecto al sistema coordenado del robot y el instante de tiempo para cada uno de los puntos.

Con el botón *"PLOT"* se visualiza en el entorno virtual el recorrido capturado en dos grupos de puntos, los azules indican las coordenadas que se encuentra dentro del espacio de trabajo del robot y los rojos aquellos que están fuera, de esta manera el usuario puede seleccionar la cantidad de puntos con los que desea simular la trayectoria. La Figura 32 muestra el entorno virtual donde se dibujan los conjuntos de puntos que están o no al alcance de los movimientos del robot.

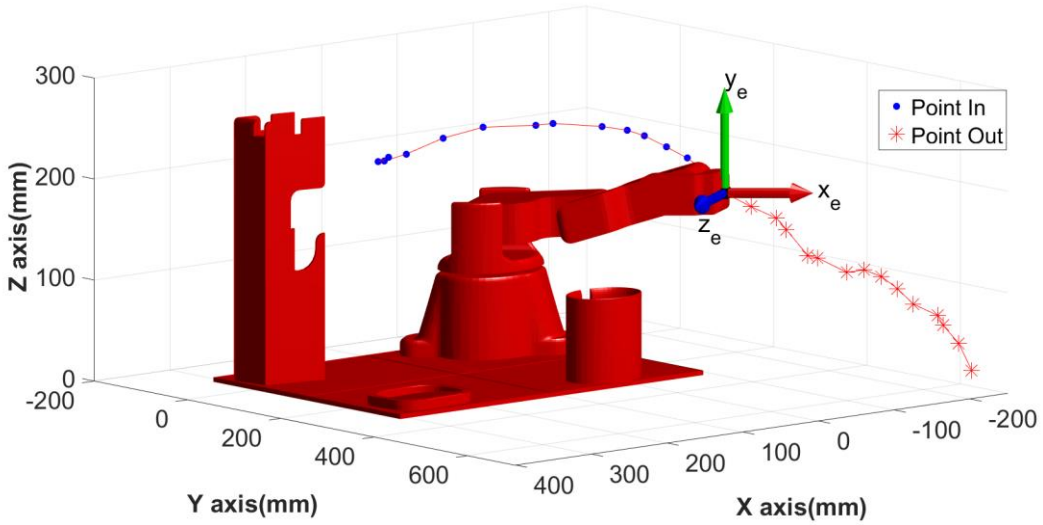


Figura 32. Entorno virtual con representación de los puntos que se encentran dentro y fuera del alcance del robot.
Nota: Los puntos azules representan el movimiento alcanzable por el robot y los puntos rojos son aquellos que no pueden ser alcanzable por el robot.

Definida la trama de puntos azules que representan la trayectoria que va a reproducir el robot real, se debe hacer como prueba la simulación en el entorno virtual dando *click* al

botón *"RUN"* habilitado en la plataforma. Con la simulación se descubre que dichos movimientos se reproducen de forma brusca, y para mejorarlos se propone generar automáticamente algunos puntos intermedios entre los ya definidos y evaluar de nuevo el recorrido. El cálculo de la cantidad de puntos intermedios varía dependiendo de la distancia entre cada par de puntos de tal forma que se ubique un punto adicional cada *5 milímetros* de distancia uno del otro aplicando la fórmula de la ecuación 101 llamada distancia euclidiana para dos puntos en R_3, este valor se determina basado en las gráficas de la Figura 38 que explican el comportamiento del robot al recorrer la trayectoria visto en el capítulo 5.3.

$$d = \sqrt{\left(P_{rxi} - P_{rx(i+1)}\right)^2 + \left(P_{ryi} - P_{ry(i+1)}\right)^2 + \left(P_{rzi} - P_{rz(i+1)}\right)^2} \qquad (101)$$

Donde *d* se reemplaza en la ecuación 102 para determinar la cantidad de puntos entre cada par de puntos.

$$n_{puntos} = \frac{d}{5} \qquad (102)$$

En la ecuación 102 el denominador representa una constante de distancia dada por el usuario para la generación del nuevo punto entre cada par de puntos, en este caso se define un valor de *5 milímetro*. Por ejemplo, si la medida de la diferencia entre dos puntos es de *1 centímetro* tendríamos *2* puntos intermedios adicionales y así sucesivamente para las diferentes medidas entre los puntos. En la Figura 34 se muestra la diferencia entre la trama original y la generada con mayor cantidad de puntos.

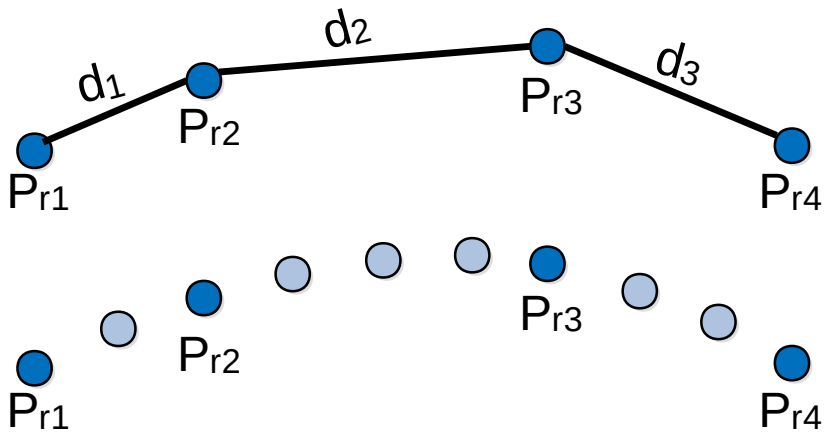


Figura 33. Puntos intermedios adicionales para la trayectoria.

Cabe destacar que el conjunto de puntos generados entre cada par de puntos de la trayectoria original debe ser evaluado nuevamente para asegurar que no estén fuera del espacio de trabajo del robot, si es así, el robot debe quedarse en la última posición alcanzable hasta que encuentre un punto en la trayectoria que este dentro de su espacio de trabajo para continuar con el recorrido.

5.2 Guardar y carga datos

Dentro de la interfaz se pueden archivar a través del botón *"Guardar"* los puntos de la trayectoria simulada para que al cerrar la aplicación estos no se pierdan y puedan ser revisados y analizados posteriormente. Los puntos se guardan en un documento con extensión *mat*.

Para cargar estos archivos *mat* se usa el botón *"Cargar"* del menú de la barra de tareas de la aplicación. En la Figura 34 se muestran los botones con los que se lleva a cabo cualquiera de las dos opciones.

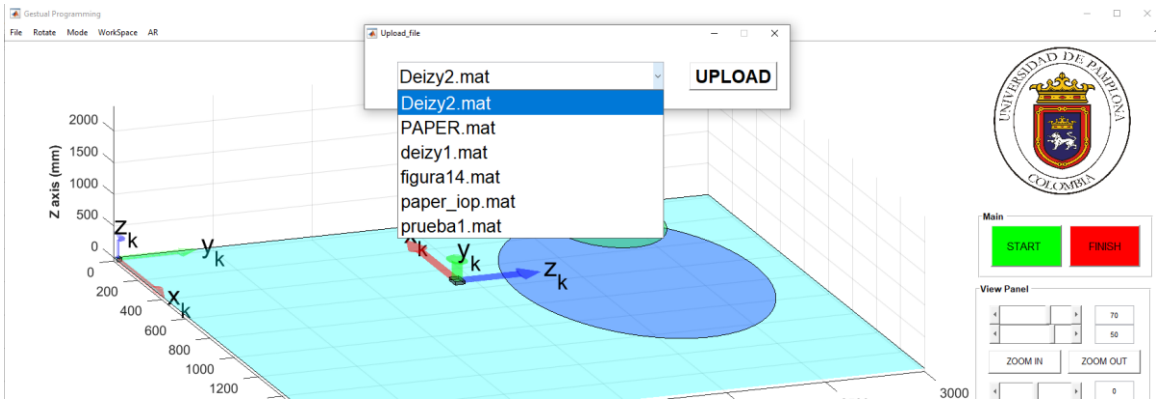


Figura 34. Interfaz para guardar o cargar un archivo *mat*.

5.3 Control cinemático

Con el control cinemático se pueden establecer límites en velocidad y aceleración para llevar a cabo los movimientos del robot, mejorando así la calidad de los resultados de la trayectoria. Para la aplicación del control cinemático se deben conocer las coordenadas articulares en cada punto de la trayectoria aplicando el modelo cinemático inverso. La matriz que se genera al aplicar el modelo cinemático inverso a los puntos seleccionados del movimiento capturado se define en la ecuación 103.

$$M_q = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \\ q_t \end{bmatrix} = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} & \cdots & q_{1n} \\ q_{21} & q_{22} & q_{23} & q_{24} & \cdots & q_{2n} \\ q_{31} & q_{32} & q_{33} & q_{34} & \cdots & q_{3n} \\ q_{41} & q_{42} & q_{43} & q_{44} & \cdots & q_{4n} \\ q_{t1} & q_{t2} & q_{t3} & q_{t4} & \cdots & q_{tn} \end{bmatrix} \tag{103}$$

Donde:

$M_q =$ Matriz que almacena las coordenadas articulares de cada uno de los puntos que representan el movimiento del operador.

$q =$ Vector que almacena las 4 coordenadas articulares del robot para un punto del movimiento capturado en un instante de tiempo q_t.

Para optimizar el movimiento de las articulaciones se aplica el interpolador splin cúbico como método de control cinemático, garantizando que se cumplan las restricciones de velocidad y aceleración máximas de los eslabones del robot Raplim dados por el usuario (restricciones del robot), con el fin de conseguir una trayectoria lo suficientemente suave y precisa. Un ejemplo del movimiento articular del primer eslabón se observa en la Figura 35.

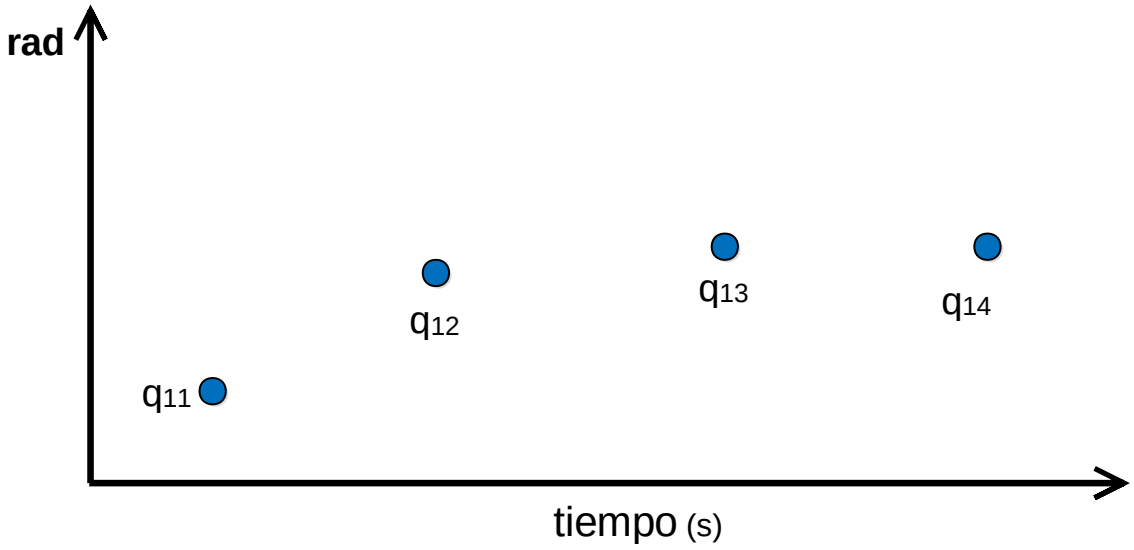


Figura 35. Movimiento articular eslabón 1.

Una vez aplicado el interpolador splin cúbico a la trayectoria articular de la Figura 35 se generan los puntos intermedios como se observa en la Figura 36 evidenciando el cambio en el movimiento articular del primer eslabón del robot Raplim.

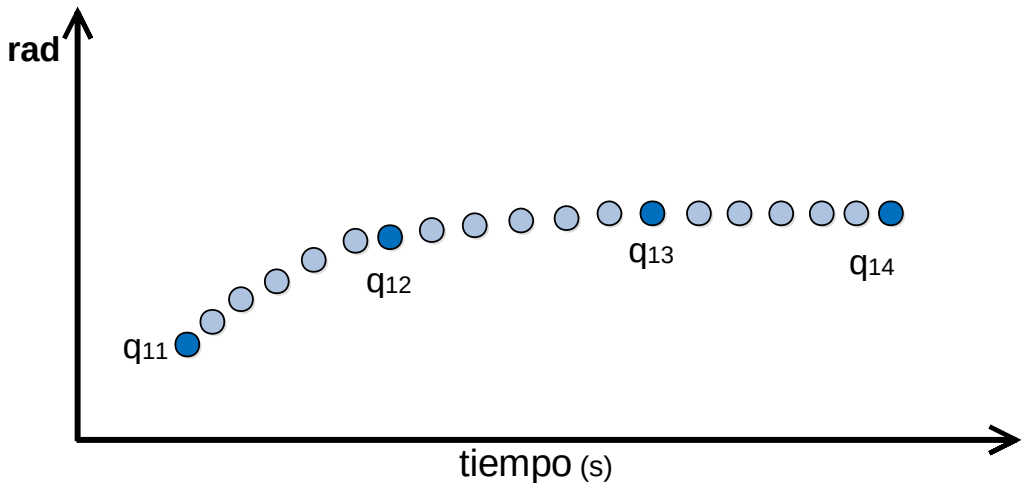


Figura 36. Cambio articular q1.

El algoritmo del interpolador splin cubico permite seleccionar la cantidad de puntos intermedios que se agregan entre dos coordenadas articulares, este valor se escoge dependiendo de qué tan suave se desea el movimiento del robot, en la Figura 37 se muestra la gráfica de la trayectoria que dibuja el robot en el plano cartesiano variando la cantidad de puntos intermedios en el interpolador y la línea recta *100 milímetros* indicada por *5* puntos con la que se evalúa el recorrido. En la imagen a) se representa la linea recta despues de aplicar el interpolador splin cubico con 2 puntos intermedios, en la b) se representa la linea recta con 5 puntos intermedios, La imagen c) representa la line arecta con 10 puntos intermedios y la d) representa la linea recta con 15 puntos intermedios.

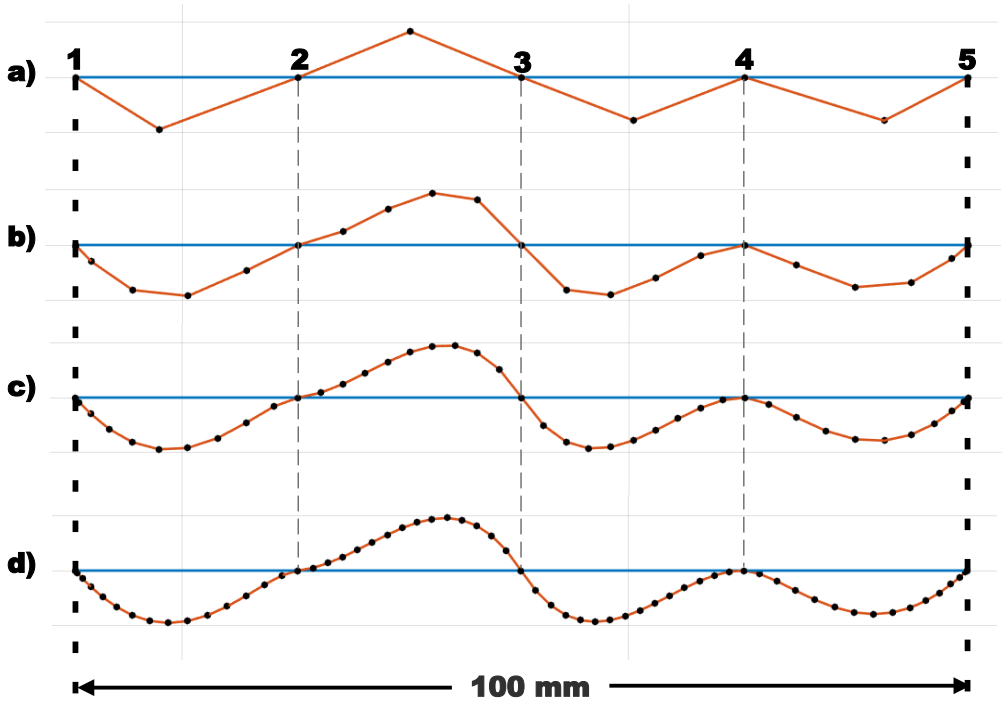


Figura 37. Variación de puntos de muestreo al aplicar el interpolador splin cubico. a) 2 puntos de muestreo, b) 5 puntos de muestreo, c) 10 puntos de muestreo y d) 15 puntos de muestreo.

Analizando las gráficas de la Figura 37 se evidencia que el recorrido que hace el robot al aplicar el interpolador con un muestro de 15 puntos representa un movimiento más uniforme, el cual fue seleccionado para generar las posteriores trayectorias de los resultados de este proyecto.

Una línea recta de 100 milímetros representada por solo *5* puntos en el espacio cartesiano no permite que el robot realice de forma aproximada el movimiento que se está evaluando al aplicar el interpolador, que para este caso es una línea recta de color azul y el movimiento del robot se ve representado en las curvas que se forman con la línea naranja por lo que se demuestra que el movimiento del robot no se parece a la línea recta. Para mejorar el movimiento del robot y representar mejor la línea recta, se aumentan los puntos de

muestreo en el plano cartesiano para representar la misma línea recta azul como se observa en las gráficas de la Figura 38. Donde a) representa la línea con 5 puntos (separados cada 20 milímetros), b) representa la línea con 7 puntos (separados cada 14.3 milímetros), c) representa la línea con 10 puntos (separados cada 10 milímetros) y d) representa la línea con 20 puntos (cada 5 milímetros).

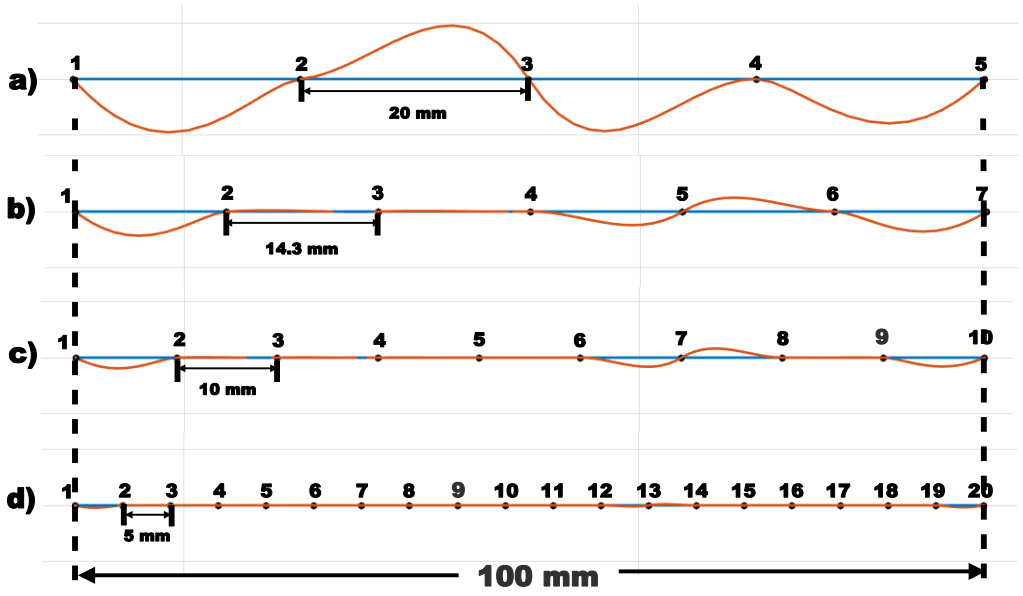


Figura 38. Variación de los puntos de muestreo en el plano cartesiano, a) 5 puntos separados cada 20 mm, b) 7 puntos separados cada 14.3 mm, c) 10 puntos separados cada 10 mm y d) 20 puntos separados cada 5 mm.

En la Figura 39 se observan las gráficas de posición, velocidad y aceleración obtenidas al aplicar el interpolador splin cúbico a cada una de las coordenadas articulares del robot Raplim necesarias para recorrer la trayectoria de línea recta en el espacio de la tarea de la Figura 38 grafica d).

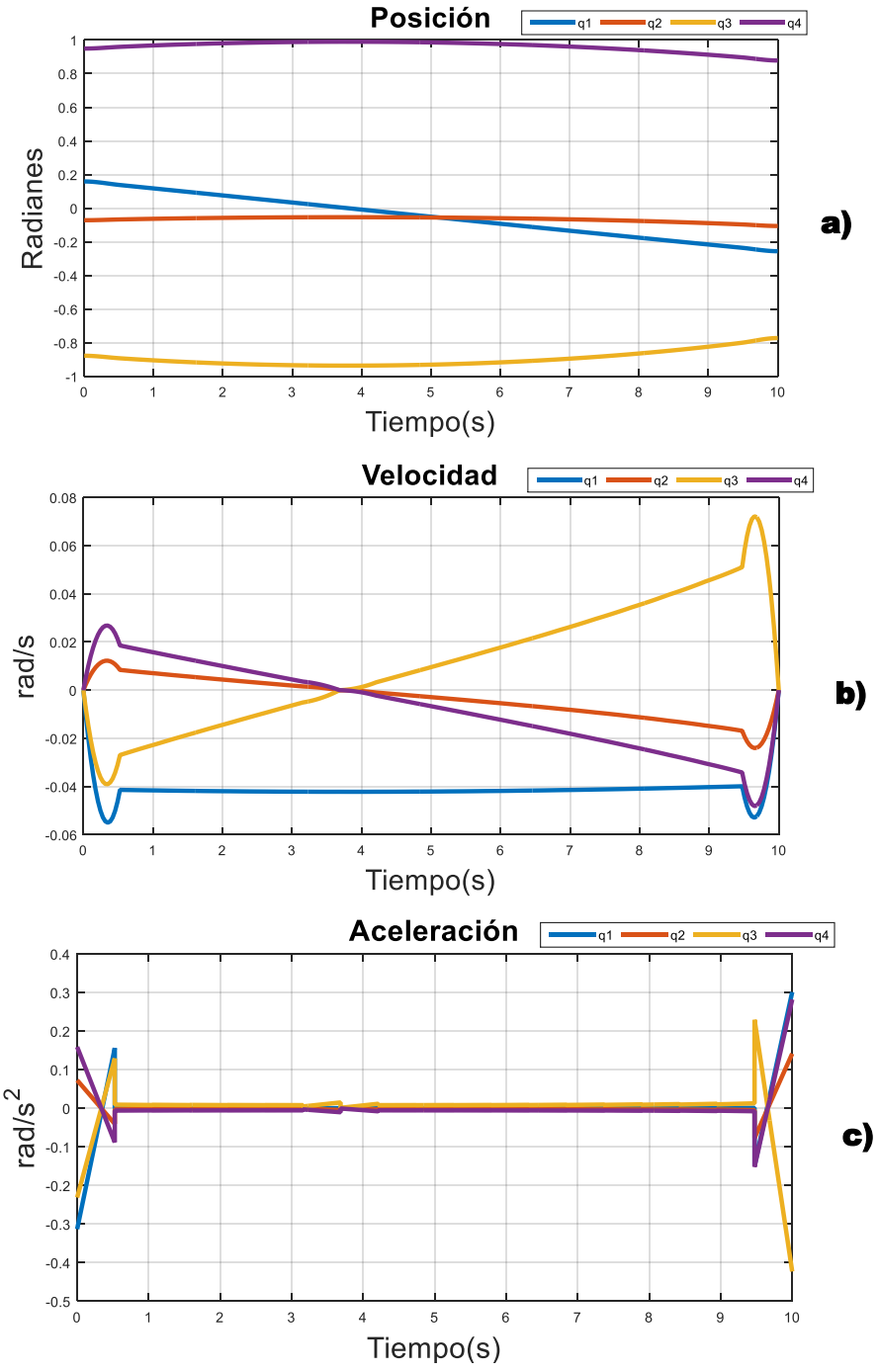


Figura 39. Graficas de las articulaciones, a) Posición, b) Velocidad y c) aceleración.

Los puntos obtenidos en el control cinemático se usan para realizar la simulación del movimiento del robot y observar que tan preciso es el movimiento generado, con el fin de ajustar los parámetros de entrada del algoritmo de control.

5.4 Algoritmo de optimización

A partir del movimiento del operador, el algoritmo captura a través del Kinect una gran cantidad de puntos, de los cuales algunos pueden no ser necesarios para definir la trayectoria, pero si generan gastos computacionales a la hora de realizar la simulación, por lo tanto, se deben seleccionar los puntos que describan mejor el movimiento minimizando el procesamiento de datos. Para esto se diseñan dos algoritmos (analítico y vectorial) que permitan analizar y seleccionar los puntos que mejor describen la trayectoria, el usuario tiene la libertad de aplicar alguno de ellos o ninguno según su conveniencia, pero siempre al final se aplicara el interpolador splin cúbico como parte del control cinemático. El diagrama de flujo que define este comportamiento se describe en la Figura 40.

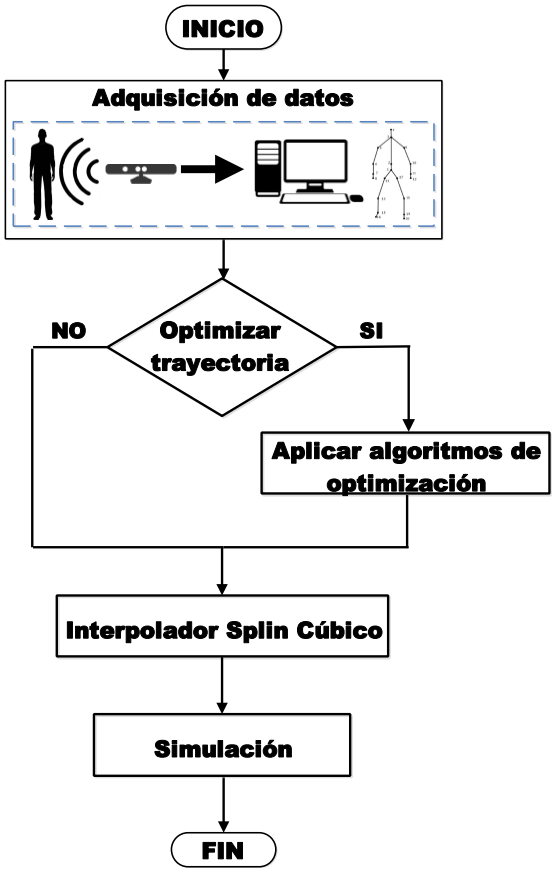


Figura 40. Diagramas de flujo para el análisis de los puntos.

5.4.1 Algoritmo de optimización vectorial.

Este método consiste en agrupar los puntos que mantienen una misma dirección en un vector acumulado que represente el movimiento capturado. Cuando la dirección de alguno de los puntos cambie se genera un segmento y se inicia un nuevo vector acumulado desde el punto donde se genera el cambio de dirección. Para este método no existe un límite de segmentos para describir la trayectoria. La Figura 41 muestra la dirección de cada uno de los vectores formados a partir de los puntos que representan el movimiento capturado.

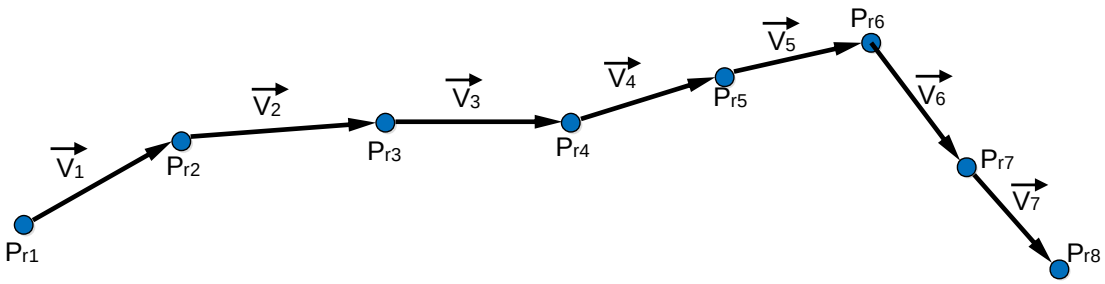


Figura 41. Método vectorial.

A continuación, se especifican con más detalle los pasos para el desarrollo del algoritmo de optimización vectorial determinando los vectores que se forman a partir de los puntos capturados con respecto al sistema de referencia del robot. Las ecuaciones 104 y 105 definen las componentes de cada una de los vectores.

El vector $\overrightarrow{V_1}$ se calcula a partir de los puntos representados en el espacio de trabajo del robot P_{r1} y P_{r2}.

$$\overrightarrow{V_1} = \left[(P_{rx2} - P_{rx1}), (P_{ry2} - P_{ry1}), (P_{rz2} - P_{rz1})\right] \quad (104)$$

De forma general un vector se representa de la siguiente forma:

$$\vec{V_i} = \left[(P_{rx(i+1)} - P_{rx(i)}), (P_{ry(i+1)} - P_{ry(i)}), (P_{rz(i+1)} - P_{rz(i)})\right] \quad (105)$$

Donde:

$\overrightarrow{V_1}$ = Es el vector formado a partir de la pareja de puntos (P_{r1}, P_{r2}).

$\vec{V_i}$ = Es la representación general para cualquier vector.

Como el algoritmo se basa en la dirección de los vectores es necesario calcular el vector unitario para cada uno de ellos utilizando las ecuaciones 106, 107 y 108.

$$\vec{V_i} = \left[V_{xi}, V_{yi}, V_{zi}\right] \tag{106}$$

$$\left|\vec{V_i}\right| = \sqrt[2]{{V_{xi}}^2 + {V_{yi}}^2 + {V_{zi}}^2} \tag{107}$$

$$\widehat{V_i} = \frac{\vec{V_i}}{\left|\vec{V_i}\right|} \tag{108}$$

Donde:

$\left|\vec{V_i}\right|$ = Magnitud del vector.

$\widehat{V_i}$ = Vector unitario.

Una vez obtenidos los vectores unitarios el algoritmo compara el vector unitario actual con el siguiente y así sucesivamente, de tal manera, que para un caso inicial tomaría el vector i y lo compara con el vector $i + 1$ aplicando el producto punto para determinar el ángulo entre los dos vectores como se describe en la ecuación 109.

$$\widehat{V_i} \cdot \widehat{V}_{(i+1)} = \left|\widehat{V_i}\right|\left|\widehat{V}_{(i+1)}\right| \cos\theta \tag{109}$$

A partir de la ecuación 109 se obtiene el ángulo que permite determinar el cambio de dirección en el movimiento capturado despejando θ como se muestra la ecuación 110.

$$\theta = \cos^{-1}\frac{\widehat{V}_i \cdot \widehat{V}_{(i+1)}}{\left|\widehat{V}_i\right|\left|\widehat{V}_{(i+1)}\right|} \tag{110}$$

La Figura 42 detalla el procedimiento grafico para determinar el ángulo que existe entre los vectores formados con los puntos de la trayectoria.

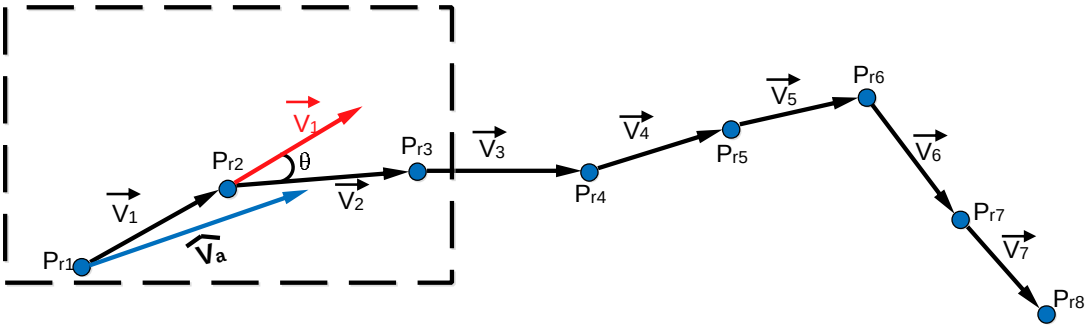


Figura 42. Procedimiento para determinar el ángulo entre dos puntos de la trayectoria.

El ángulo resultante de aplicar la ecuación 110 debe cumplir con la condición establecida por el programador para evaluar si se produce un cambio en la dirección de la trayectoria. Si la condición se cumple, se calcula un vector promedio entre estos dos vectores, al cual se le denomina vector acumulado. El vector unitario del vector acumulado debe compararse con el siguiente vector unitario $i + 2$ y así sucesivamente hasta acabar de recorrer todos los puntos generados en la captura del movimiento. Con la ecuación 111 se calcula el ángulo entre el vector acumulado y el siguiente vector del arreglo.

$$\widehat{V}_a \cdot \widehat{V}_{(i+2)} = \left|\widehat{V}_a\right|\left|\widehat{V}_{(i+2)}\right|\cos\theta \tag{111}$$

Donde:

$\hat{V}_a =$ Es el vector unitario acumulado.

La Figura 43 muestra el esquema general de los resultados del cálculo de la ecuación 111.

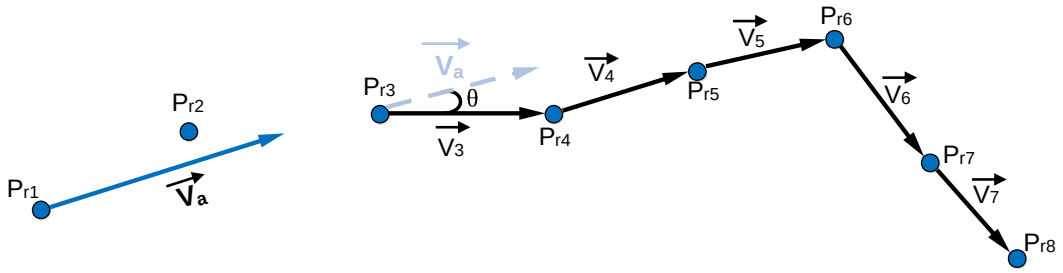


Figura 43. Vector acumulado unitario.

Si se sigue cumpliendo la condición, el vector acumulado debe tomar el valor del promedio de los vectores con los que está operando, así el valor del vector acumulado depende de la cantidad de vectores que ha agrupado (c), la ecuación 112 representa de forma matemática esta condición.

$$\vec{V}_a = \frac{\left(\hat{V}_a * c\right) + \hat{V}_{c+1}}{c + 1} \tag{112}$$

Donde:

$\vec{V}_a$ = Es el promedio de los vectores unitarios que mantienen la misma dirección.

c = Es la cantidad de vectores que se han acumulado.

$\hat{V}_{c+1}$ = El vector unitario actual que cumple la condición.

Ahora, sí el ángulo formado entre estas parejas de vectores unitarios no cumple la condición propuesta por el programador en el algoritmo como se observa en la Figura 44 a) entre los vectores $\overrightarrow{V_a}$ y $\overrightarrow{V_6}$, se debe comparar el vector acumulado $\overrightarrow{V_a}$ con un nuevo vector $\vec{V}_n$ formado a partir de los puntos P_{r6} y P_{r8} de la trayectoria visto en la Figura 44 b).

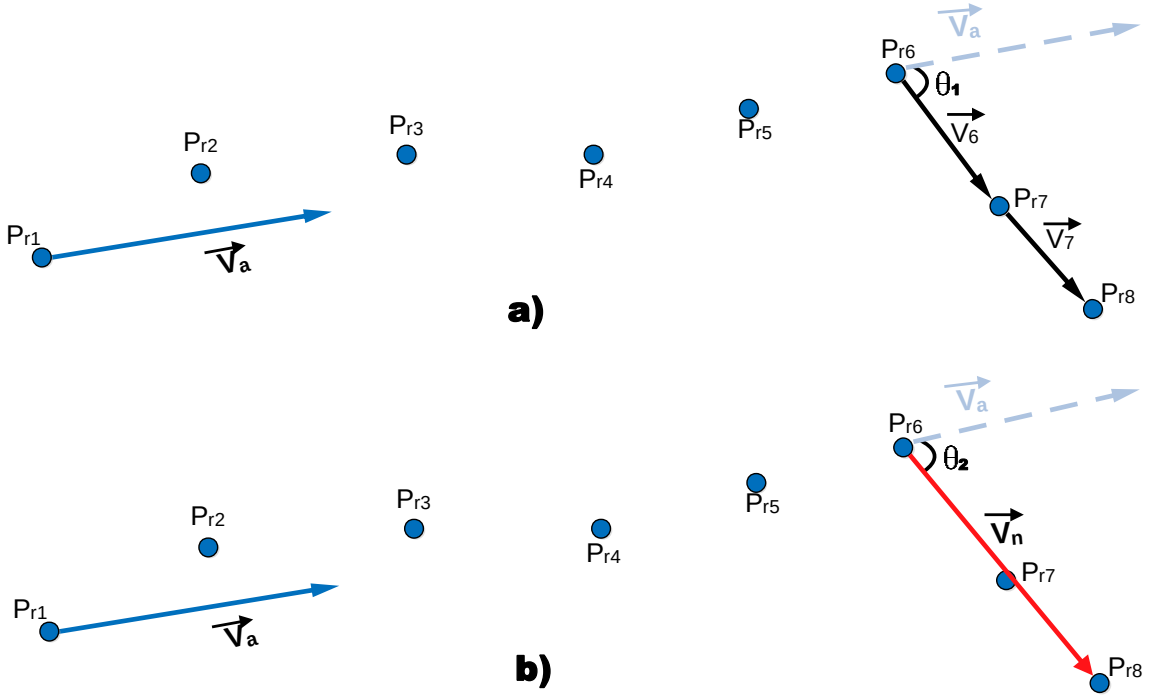


Figura 44. Análisis del sistema con el a) vector que no cumple con la condición y b) Nuevo vector unitario y su ángulo.

El procedimiento matemático para obtener el nuevo vector unitario $\vec{V}_n$ es el mismo que se aplica en las ecuaciones 106, 107 y 108.

Si con el nuevo vector unitario $(\hat{V}_n)$ se cumple la condición, se concluye que no existe cambio de dirección y se siguen analizando los demás vectores unitarios con el vector acumulado, de esta manera el vector acumulado se calcula tomando en cuenta que el vector

$\vec{V}_n$ lleva un valor ponderado de 2 ya que substituye 2 vectores $(\overrightarrow{V_6}, \overrightarrow{V_7})$ cómo se observa en la ecuación 113.

$$\vec{V}_{prom} = \frac{\hat{V}_a * c + \hat{V}_n * 2}{c + 2} \tag{113}$$

Donde:

$\vec{V}_{prom}$ = Es el promedio de los vectores unitarios que mantienen la misma dirección.

c = Es la cantidad de vectores que se han acumulado.

$\hat{V}_n$ = Es el nuevo vector unitario.

En caso opuesto que con el vector unitario $\hat{V}_n$ no se cumpla la condición, se asume que existe un cambio en la dirección de la trayectoria, por lo tanto, se crea un segmento $(\overline{S_1})$ desde el primer punto hasta el último punto que cumpla la condición, es decir, donde ocurre el cambio de dirección, se guardan esos dos puntos (P_1, P_2) para ser posteriormente utilizados en el algoritmo de control cinemático. Para continuar con el procedimiento en los vectores unitarios restantes de la trayectoria se reestablecen los valores del vector acumulado $\overrightarrow{V_a}$ y el contador de vectores c. La Figura 45 detalla el comportamiento general cuando ocurre el cambio en la dirección de la trayectoria.

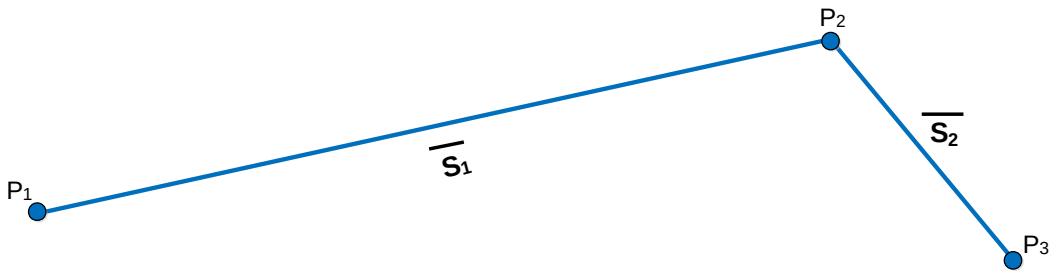


Figura 45. Segmentos que representan la trayectoria.

Al aplicar el algoritmo de optimización se debe conservar el primer y el último punto de la trayectoria así no exista un cambio de dirección, por ende, en la Figura 45 se observa que el segmento $\overline{S_2}$ se forma a partir de los puntos P_2 y P_3, ya que no existen mas puntos en la trayectoria.

5.4.2 Algoritmo de optimización analítico

El método analítico se basa en la medida del error cuadrático medio *(RMSE)* que mide el error absoluto del cuadrado de las desviaciones positivas y negativas para evitar que se cancelen entre sí. Para este método el operador debe especificar la cantidad de segmentos en los que quiere dividir la trayectoria y con esta información calcular el error medio de cada segmento. Con la ecuación 114 se calcula el *RMSE*.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(d_i)^2}{n}} \tag{114}$$

Donde:

d_i = Distancia perpendicular de un punto de la trayectoria a un punto del segmento.

n = Es la cantidad de puntos de la trayectoria original comprendidos entre los puntos que forman el segmento.

Una vez definida la cantidad de segmentos que se quieren evaluar en la trayectoria, se identifican los puntos que forman los segmentos donde el error promedio con respecto a la trayectoria original es el más bajo. La Figura 46 muestra un ejemplo de cómo se vería la selección de los puntos para 3 segmentos.

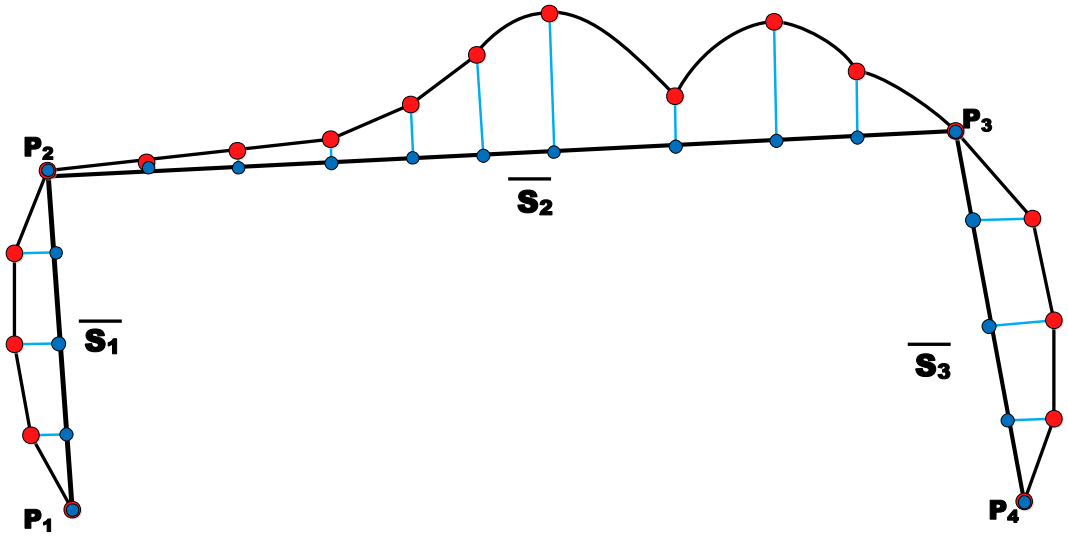


Figura 46. Método analitico.

En la Figura 46 se observan los puntos rojos de la trayectoria original y los puntos azules de los segmentos que forman la nueva trayectoria. El procedimiento para aplicar el método de optimización analítico consiste en hacer combinaciones de todos los puntos que pertenecen a la trayectoria para crear múltiples segmentos a partir de los cuales se calcule el *RMSE* y registrar este valor para cada combinación de puntos y finalmente determinar cuál es la combinación que tiene el menor error.

Los segmentos generados se componen de 500 puntos equidistantes entre cada par de puntos que pertenecen a la trayectoria y la combinación, para calcular el error medio de la primera combinación se determinan las distancias perpendiculares de los puntos de la trayectoria original con respecto a los segmentos generados por esta combinación. Esta distancia perpendicular se calcula para cada punto de la trayectoria comprendido entre los puntos que forman el segmento, es decir, si tomamos el primer punto de la trayectoria se deben determinar las distancias entre este punto y todos los puntos que pertenecen al segmento generado. La menor distancia entre el punto de la trayectoria y algún punto del segmento se considera como la "distancia perpendicular". En la Figura 47 se ilustra el procedimiento con el que se determina la distancia perpendicular de un punto de la trayectoria con respecto al segmento.

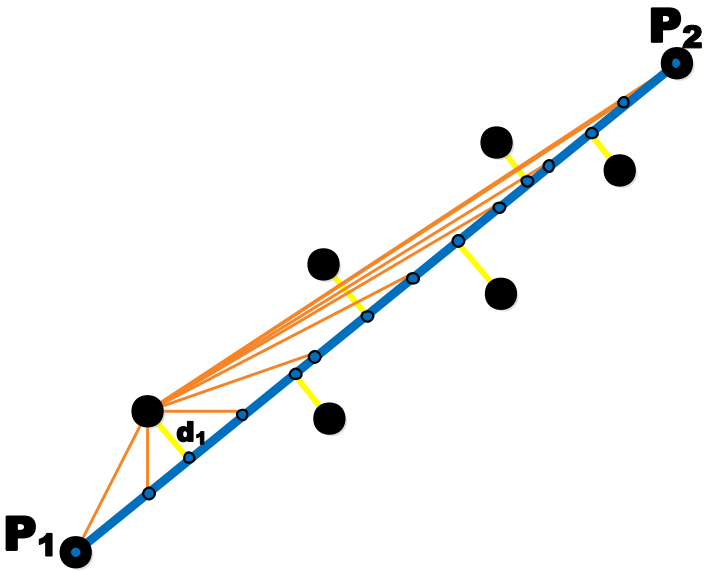


Figura 47. Distancia perpendicular.

El cálculo de la distancia entre un punto de la trayectoria original y un punto del segmento se detalla en la ecuación 115.

$$d_i = \sqrt{\left(P_{rxi} - P_{sxj}\right)^2 + \left(P_{ryi} - P_{syj}\right)^2 + \left(P_{rzi} - P_{szj}\right)^2} \quad (115)$$

Donde:

$d_i =$ Es la distancia entre el punto de la trayectoria y el punto del segmento.

$P_r =$ Es un punto de la trayectoria.

$P_s =$ Es un punto del segmento.

$i =$ Es el índice que indica la posición del punto de la trayectoria que pertenece al arreglo acotado por el segmento de la combinación.

$j =$ Es el índice que indica la posición del punto del segmento.

Calculadas las distancias perpendiculares de cada segmento se determina el error cuadrático medio de cada uno de los segmentos utilizando la ecuación 114, para finalmente obtener el error total de esta combinación a partir de la suma algebraica de los errores de cada segmento de la trayectoria de la Figura 46 aplicando la ecuación 116.

$$E_{T(n)} = E_{s1} + E_{s2} + E_{s3} \quad (116)$$

Donde:

$E_T =$ Error Total.

$E_s =$ Error cuadrático medio del segmento.

$n =$ Índice de la combinación.

El error total y los puntos recopilados para la combinación se guardan en una matriz para seguir haciendo el mismo procedimiento en las demás combinaciones. La cantidad de combinaciones depende del conjunto de puntos de la trayectoria y el número de segmentos que se quieran generar, la ecuación 117 presenta el cálculo de las posibles combinaciones ordinarias de forma general.

$$C_{p,s} = \frac{(p-2)!}{(s-1)!\,((p-2)-(s-1))!} \tag{117}$$

Donde:

C = Son el número de combinaciones.

p = Es el número de puntos de la trayectoria original.

$s =$ Es el número de segmentos de la nueva trayectoria.

El algoritmo de optimización analítico selecciona la combinación de puntos que menor error presenta con respecto a la captura del movimiento del operador de acuerdo a la cantidad de segmentos para representar la trayectoria con solo algunos puntos de la captura original.

Evaluados ambos métodos de optimización se puede decidir de acuerdo a sus resultados a cuál se le debe aplicar el control cinemático acondicionando la nueva trayectoria para simularla en el entorno virtual y en realidad aumentada.

5.5 Simulación

Para comprobar los cálculos de optimización de la trayectoria creada a partir de la lectura de los datos de Kinect se debe realizar el paso previo a la implementación que es, la simulación. En la Figura 48 se detalla el diagrama de flujo que describe el algoritmo para simular la trayectoria del robot.

En el primer bloque se leen los datos de posición de la captura del movimiento que hace el usuario con respecto al sistema coordenado del robot para generar la trayectoria. En el segundo bloque de procesos se crean los puntos de muestreo cada *5 milímetros* para hacer que el robot recorra con precisión y suavidad ala trayectoria. En el tercer bloque se aplica el control cinemático (cinemática inversa e interpolador splin cubico) para obtener las coordenadas articulares que representan el movimiento que debe ejecutar el robot al hacer la trayectoria. En el bloque cuatro se carga el modelo de robot *3D* para simular el comportamiento del robot al llevar a cabo la trayectoria. Por último, el algoritmo de simulación representa cada posición articular siempre y cuando este dentro del espacio de trabajo del robot, de no ser así el robot se mantendrá en la última posición alcanzable.

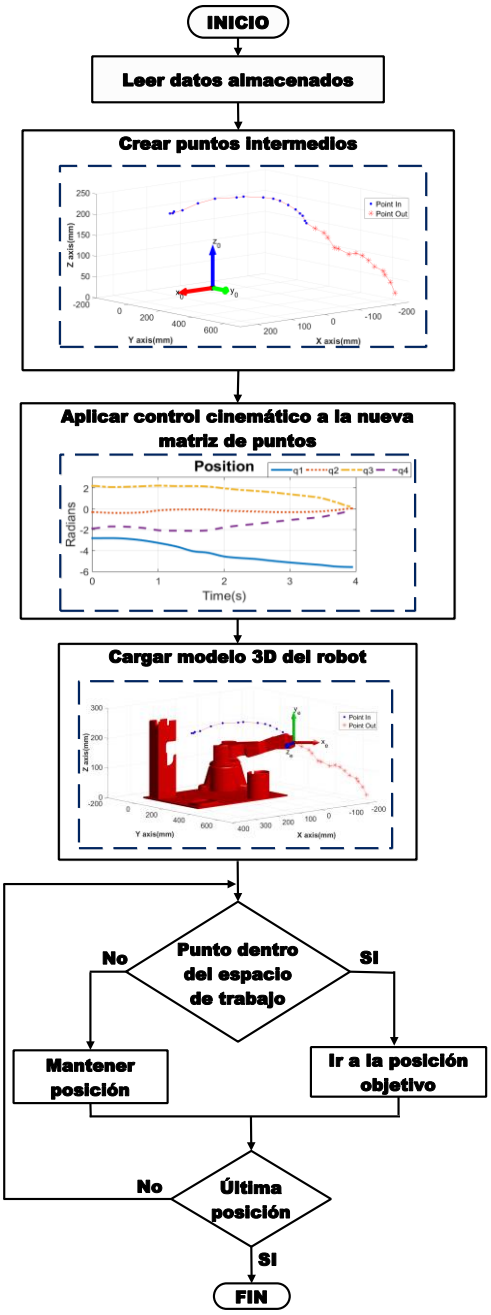


Figura 48. Diagrama de flujo para la simulación del recorrido del robot sobre los puntos.

La simulación permite comprobar el comportamiento del robot al recorrer la trayectoria, es decir, si el efector final es capaz de seguir el movimiento capturado por el operador con precisión y suavidad.

La simulación utiliza el modelo mecánico del robot importado por piezas con extensión *stl* al software matemático donde se desarrollan todos los algoritmos, estas piezas se crean en un programa de diseño mecánico provisto por la Universidad de Pamplona UP para trabajar con el modelo *CAD* del robot.

5.5.1 Importar diseño del robot al software matemático.

Al importar los archivos del modelo *CAD* con extensión *stl* que contienen los puntos correspondientes a la superficie de cada articulación se pueden ensamblar sus partes (eslabones) en el software matemático para mostrar la estructura *3D* del robot Raplim en el entorno virtual. A continuación, se detalla el procedimiento para generar el ensamble del robot en el entorno virtual.

El modelo *3D* de las piezas que hacen parte del robot Raplim se dividen en cinco partes la base y sus cuatro eslabones, la Figura 49 muestra la estructura completa del robot visualizada desde el programa Blender (programa de modelado 3D).

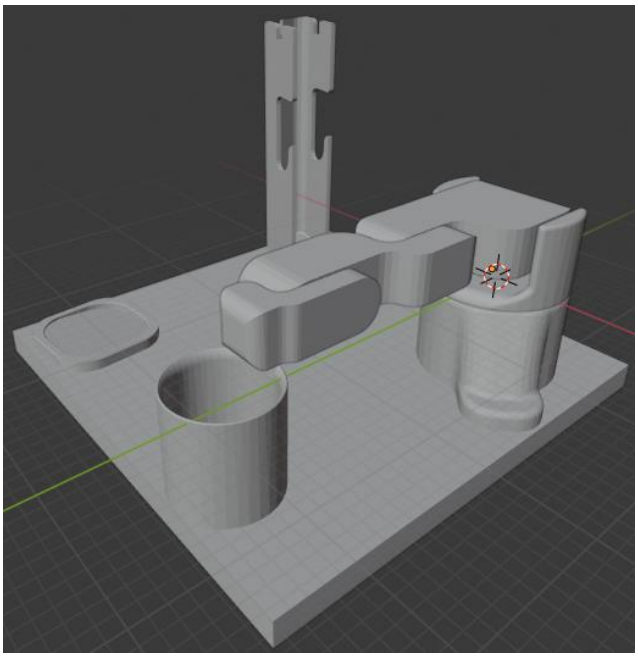


Figura 49. Estructura 3D del robot Raplim diseñada en el programa Blender.

Los archivos *stl* guardan la estructura de capa pieza del robot en pequeñas superficies triangulares indicadas por 3 puntos en el espacio tridimensional que permiten al software matemático interpretarlas y formar la estructura del robot en el entorno virtual. En ocasiones si el diseño no se hace de acuerdo al punto de origen donde deben estar ubicadas las articulaciones se deben ajustar los eslabones mediante el uso de matrices homogéneas de traslación y rotación para que al ensamblarse coincidan con la posición real de la estructura del robot.

La estructura del algoritmo que se usa en para cargar las piezas en el software mate matemático y ajustar los eslabones se define en el anexo 1.

Al ensamblar los eslabones del robot Raplim en el software matemático se define el modelo *3D* como se muestra en la Figura 50.

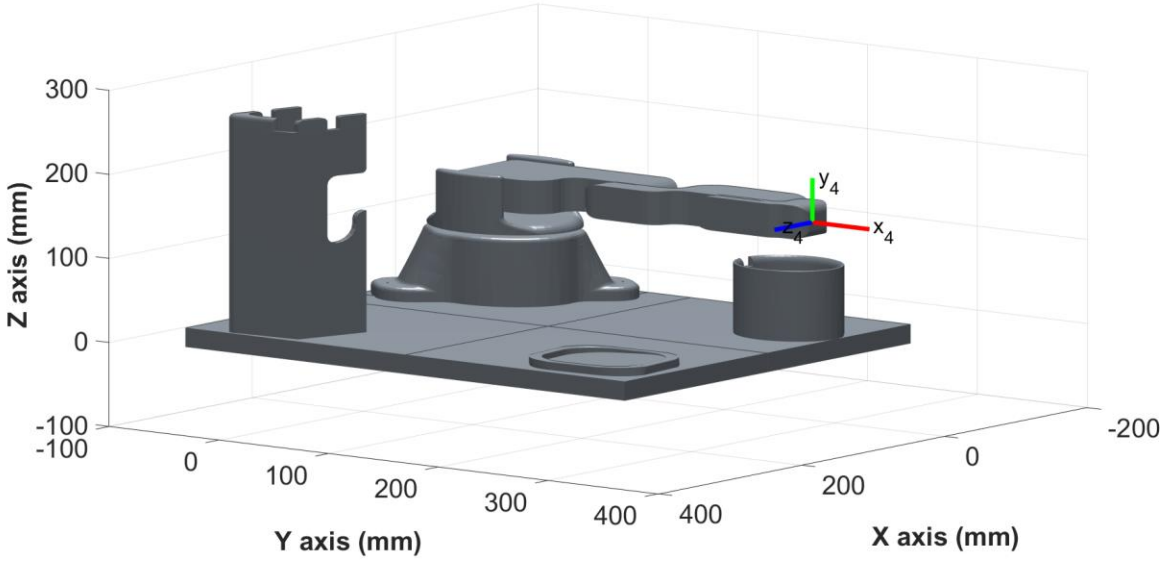


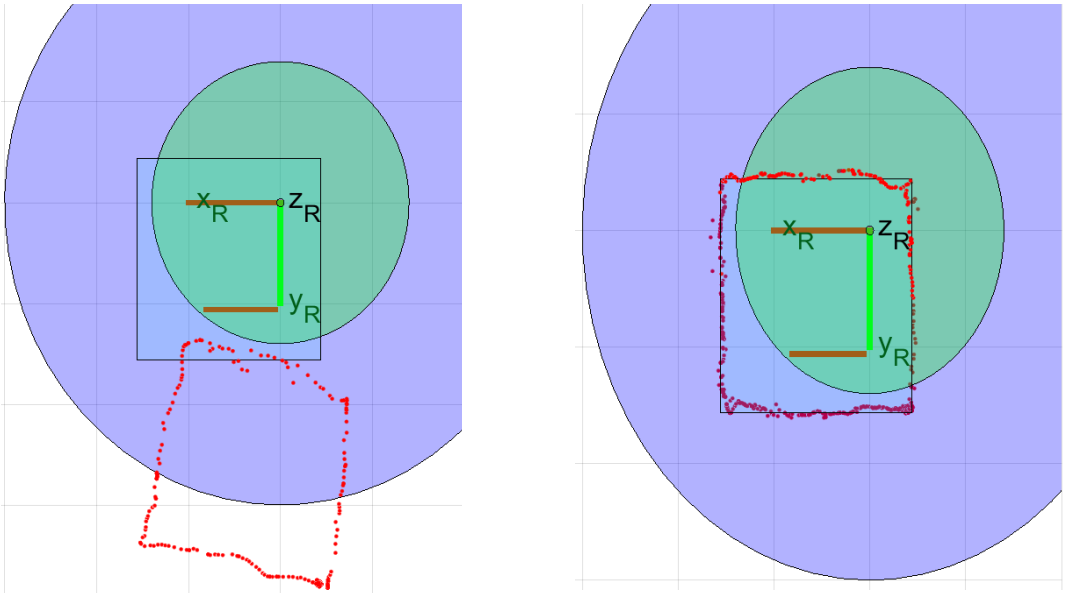
Figura 50. Modelo *3D* del robot Raplim.

El diseño *3D* en el entorno virtual se sobrepone al diagrama de alambres visto en la Figura 15, de esta manera se corrobora gráficamente que el robot se ajusta correctamente al entorno de simulación del software matemático, ya que el modelo *3D* y las distancias entre sus eslabones coinciden satisfactoriamente.

5.6 Calibración

Antes de hacer la captura de los movimientos se debe definir la ubicación del robot en lo que va a ser su zona de trabajo, definiendo la posición y orientación de su sistema

coordenado, sí no se hace de esta manera, las indicaciones que da el operador para generar la trayectoria no coincidirán con los movimientos del robot en el entorno virtual y en consecuencia la planificación de la tarea no se podrá gestionar. La Figura 51 a) y la Figura 51 b) representan un ejemplo del comportamiento de un sistema sin calibrar y calibrado respectivamente.



a) Toma sin calibrar b) Toma calibrada

Figura 51. Calibración del sistema a) Toma sin calibrar y b) toma calibrada.

El procedimiento de calibración se torna tedioso si se hace de forma Manual, debido a que la captura se debe hacer mediante el sensor Kinect y no siempre coinciden con las medidas que se toman de forma manual en el entorno físico, Adicionalmente se debe tener en cuenta la orientación del sistema coordenado del robot con respecto al sistema de Kinect.

Para mejorar la precisión en el proceso de calibración se desarrollan dos ventanas en la plataforma con las que se puede calibrar y validar el sistema coordenado donde se ubica el robot. Para llevar a cabo este procedimiento de forma automática se aprovechan las características estructurales del robot detalladas a continuación.

Como la base del robot es cuadrada se aprovechan sus cuatro esquinas como puntos de referencia para la calibración. De esta manera los límites de la calibración se hacen poniendo la muñeca derecha del operador sobre las cuatro esquinas de la base del robot donde se capturan un promedio de *100* mediciones de forma automática utilizando la interfaz de usuario. Para activar la medición en cada una de las esquinas a través de la ventana de calibración se debe elevar la muñeca izquierda 2 *segundos* y con retroalimentación visual la ventana indicara que la muestra ha sido registrada y se puede pasar a la siguiente esquina, hasta que recorra los cuatro puntos. La Figura 52 muestra la base del robot donde se toman las medidas para la calibración.

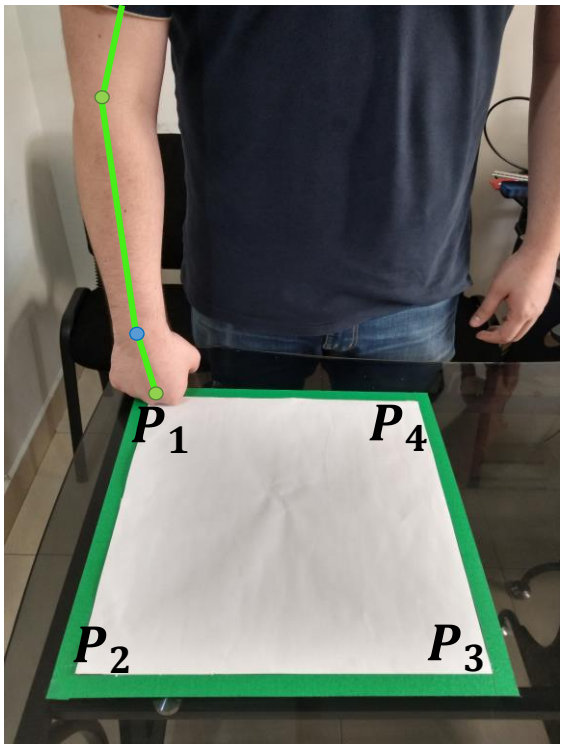


Figura 52. Espacio para la toma de datos.

Para la calibración se debe tener en cuenta la posición del Kinect debido a que influye directamente en las tomas, es decir, la rotación de la cámara en sus 3 ejes *(*R_{xk}, R_{yk}, R_{zk} *)* está directamente relacionada en la muestra.

La interfaz con la que interactúa el usuario para calibrar el sistema coordenado del robot y de Kinect se muestra en la Figura 53.

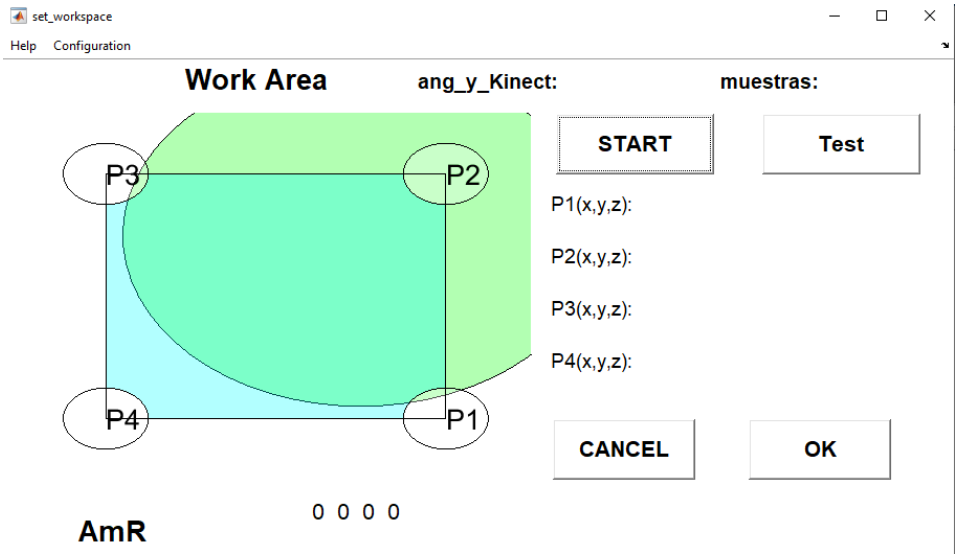


Figura 53. Entorno virtual de calibración.

Para este proyecto la ubicación del Kinect se mantiene fija y a una altura de 2 *metros* lo que genera una inclinación y rotación sobre los puntos capturados de los movimientos del operador. En la Figura 54 se observa la posición de los objetos en el espacio de trabajo real.

Figura 54. Espacio real de trabajo.

Como se observa en la Figura 55 el Kinect tiene una posición y una rotación de sus ejes fijas para que la cámara enfoque el espacio de trabajo del robot y al operador, por lo que se define la ubicación del instrumento ($X = 1.63\ m, Y = 0.77m, Z = 2.05m$) con respecto a la celda de manufactura para calibrar el sistema coordenado del robot.

El Kinect Internamente posee un acelerómetro con el que se puede determinar matemáticamente y automáticamente los grados de rotación según la inclinación a la que se vea sometido. El acelerómetro es un sensor que permite determinar la aceleración a la cual es sometido un objeto, en este caso en particular el Kinect se encuentra en reposo, por lo que la aceleración a la que él se encuentra sometido es la gravedad. El Kinect automáticamente hace los cálculos necesarios para entregar las medidas de cada eje coordenado entre *(-1* y *1)* que equivalen a estar en dirección opuesta y en la misma dirección de la gravedad respectivamente. En la Figura 55 se muestra al Kinect sin ninguna inclinación y el acelerómetro tiene los siguientes valores ($X = 0.0, Y = -1,\ Z = 0.0$).

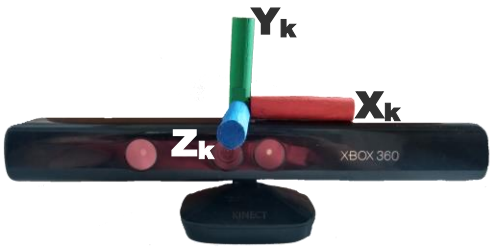


Figura 55. Kinect en reposo.

Teniendo en cuenta lo anterior mencionado se pueden determinar los ángulos de rotación de los ejes X y Z de Kinect con ayuda del acelerómetro interno. Para calcular la rotación en el eje X_k se usa la ecuación 118 según las variables involucradas en la Figura 56.

$$\propto = \operatorname{atan2}(g_z, -g_y) \tag{118}$$

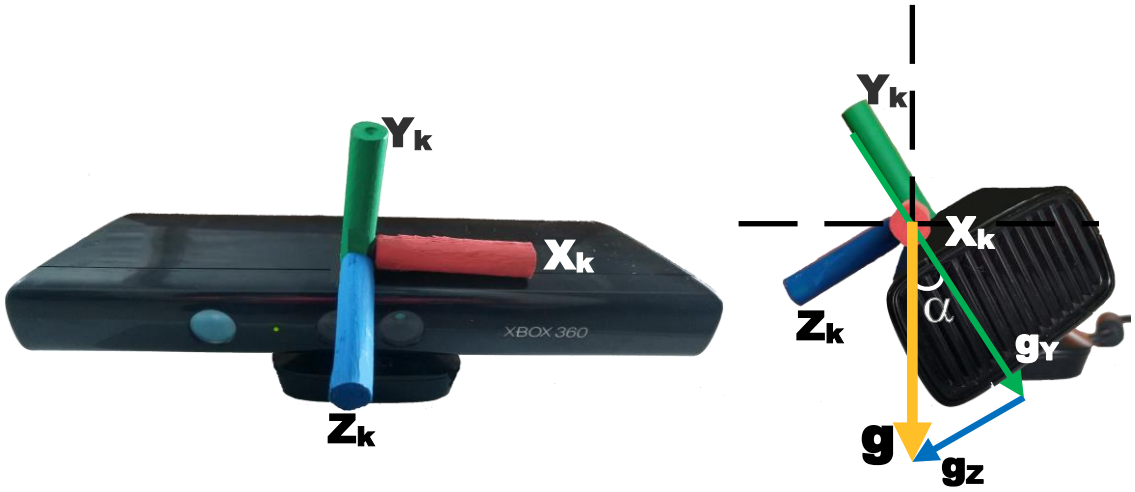


Figura 56. Variables para calcular el ángulo de rotación en el eje ***X***.

Para calcular la rotación en el eje X_k se usa la ecuación 119 según las variables involucradas en la Figura 57.

$$\beta = \operatorname{atan2}(-g_x, -g_y) \tag{119}$$

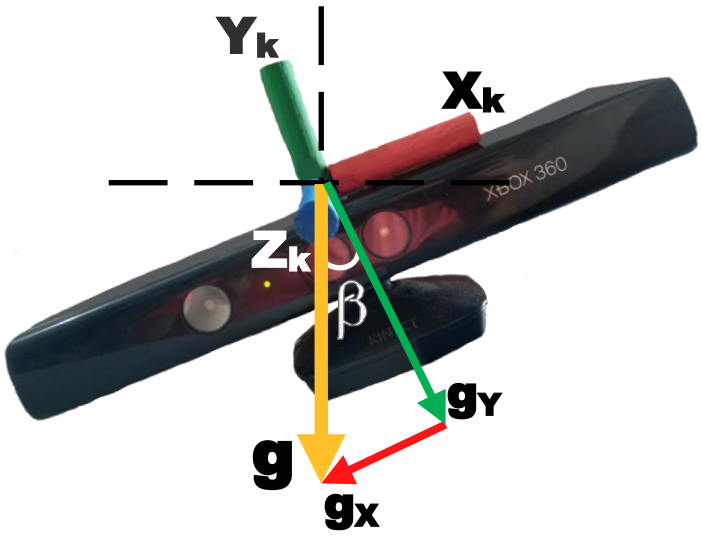


Figura 57. Variables para calcular el ángulo de rotación en el eje $\boldsymbol{Z}$.

En la Figura 58 b) se ilustra el ángulo de rotación con respecto al eje Y de Kinect (θ) que se determina tomando *2* medidas en dos de las esquinas de la base del robot que estén de frente al Kinect (P_2 y P_3) de la Figura 58 a) con las que luego se aplica la ecuación 120.

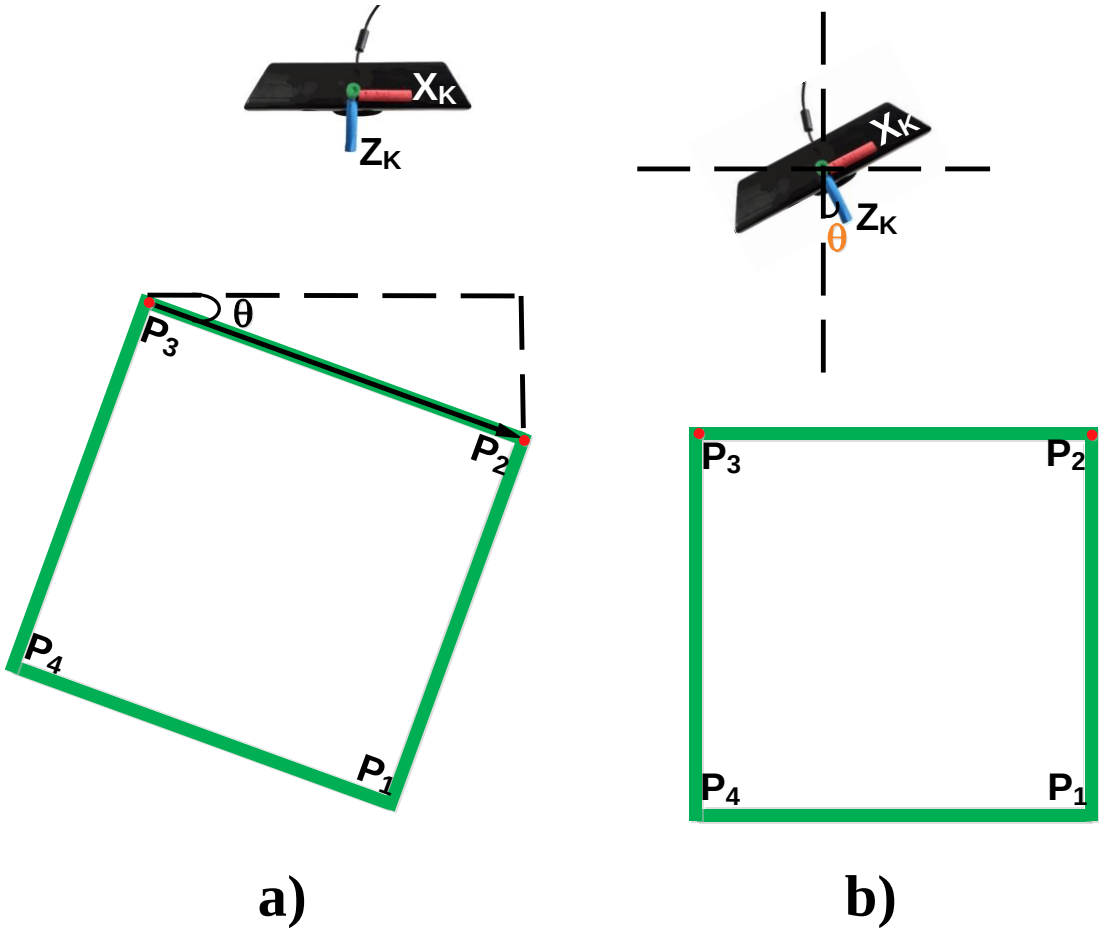


Figura 58. Rotación en el eje *Y* de Kinect. a) Vista de techo ángulo en la base del robot y b) Vista de techo ángulo en el Kinect.

$$\theta = atan2\left(P3_y - P2_y, P3_x - P2_x\right) \tag{120}$$

Teniendo las medidas de cada ángulo de rotación se hacen las transformaciones para el sistema coordenado de Kinect utilizando la ecuación 121.

$$^{0}A_{Kc} = {}^{0}A_K * Rot_x(\alpha) * Rot_y(\theta) * Rot_z(\beta) \tag{121}$$

Ajustado el sistema coordenado del Kinect se procede a realizar la calibración del sistema coordenado del robot donde solo se tendrán en cuenta las coordenadas *x, y, z* con respecto a la celda de manufactura.

En la calibración solo se tendrán en cuenta las cuatro esquinas de la base del robot por lo que determinar *x* en el sistema coordenado del robot requiere de promediar las componentes de los puntos P_1 y P_2. Para determinar ***y*** se promedian las componentes de P_2 y P_3. Por último, para calibrar el sistema coordenado de *z* se promedian las componentes de los puntos P_1, P_2, P_3 y P_4.

Las ecuaciones 122, 123, 124 y 125 son el resultado de la calibración del sistema coordenado del robot.

$$P_x = \frac{P1_x + P2_x}{2} \tag{122}$$

$$P_y = \frac{P2_y + P3_y}{2} \tag{123}$$

$$P_z = \frac{P1_z + P2_z + P3_z + P4_z}{4} \tag{124}$$

$$^0A_R = \begin{vmatrix} 1 & 0 & 0 & P_x \\ 0 & 1 & 0 & P_y \\ 0 & 0 & 1 & P_z \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{125}$$

Calibrado el sistema, se realiza una prueba capturando los puntos que se grafican con el movimiento de la muñeca derecha a través del Kinect y la interfaz de usuario con los que

el operador puede evaluar que tan precisa fue la toma de los datos. La Figura 59 muestra la plataforma virtual que se utiliza para las pruebas.

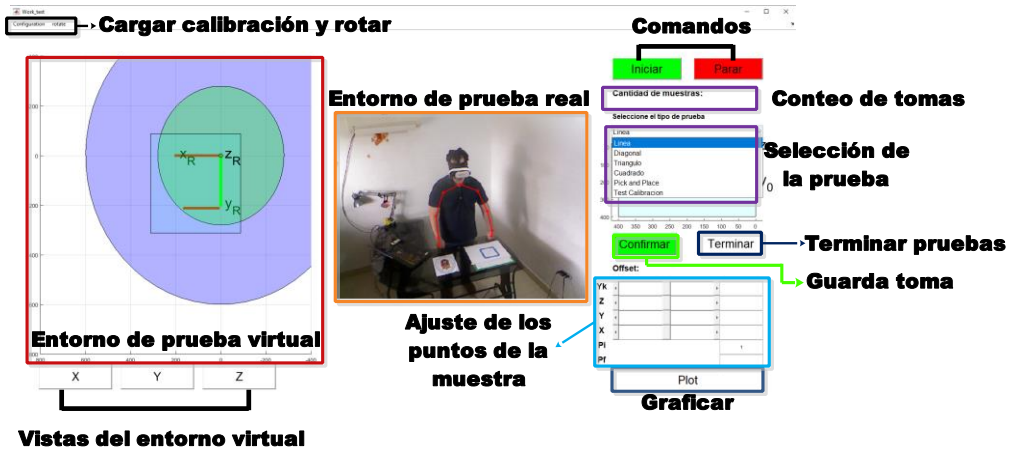


Figura 59. Plataforma de pruebas.

6. REALIDAD AUMENTADA

La realidad aumentada se usa en el desarrollo de este proyecto como herramienta visual para que el operador observe en tiempo real y a través de un dispositivo inteligente el movimiento del robot en un entorno virtual, mientras que el operador define la trayectoria en el un ambiente real utilizando la cámara de luz estructurada. El objetivo de utilizar la realidad aumentada es proveer al operador la capacidad de visualizar en tiempo real el comportamiento del robot en un ambiente virtual para no sacarlo de la línea de producción, además de predecir movimientos que lleven a colisiones al programar la trayectoria ya que permite observar el comportamiento que tiene el robot con los movimientos producido por el operador.

El programa que se usa para implementar la realidad aumentada es Unity motor de videojuegos multiplataforma utilizado en el desarrollo de videojuegos, pero en este proyecto se va a utilizar para el desarrollo de la aplicación para la visión artificial con la que se reconoce el marcador que activa el robot virtual.

A continuación, se describen lo necesario para llevar a cabo la realidad aumentada.

6.1 Marcador

Los marcadores se usan como indicadores que generan o activan la realidad aumentada, el marcador que se utiliza se crea en vuforia, donde se pueden generar

marcadores de tipo *2D* como imágenes planas o logos y *3D* como cubos, cilindros o figuras, siempre y cuando puedan ser reconocidos por una cámara. El marcador que se creo es de tipo *2D* y representa el logo de la Universidad de Pamplona, al generar el marcador se puede identificar características que definen el puntaje de reconocimiento por la cámara. En la Figura 60 a) se muestra el logo como imagen y en la Figura 60 b) las respectivas características como marcador creado en *vuforia* para cargar en *Unity*.

a) Logo de la UP

b) Logo como marcador

Figura 60. Marcador.

6.2 Modelo CAD en Unity

Una vez creado el marcador se debe cargar en *Unity* el modelo en *3D* del robot Raplim con ciertas características habilitadas en el diseño del modelo diseñado en *Blender* para ejecutar correctamente los movimientos del robot en la realidad aumentada.

En *Blender* de acuerdo al modelo cinemático del robot se restringen los movimientos en los ejes en los que el robot no deba moverse y se dejan libres aquellos donde si pueda rotar, adicional a esto se deben enlazar los eslabones, es decir, crear una jerarquía de tal forma

que los cambios en cualquier eslabón afecten el movimiento en las demás articulaciones. En la Figura 61 se muestra la jerarquía del diseño en *Blender* de los eslabones del robot según sus características mecánicas y las restricciones en los mismos dependiendo de la articulación donde se ejecute el movimiento.

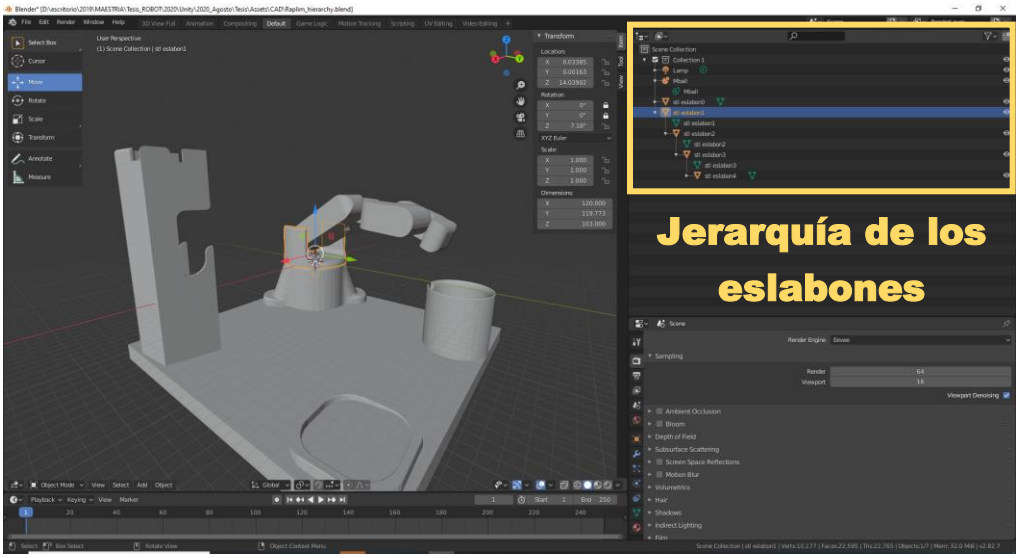


Figura 61. Jerarquía de los eslabones en *Blender*.

Ajustado el modelo CAD del robot en Blender se cargan los archivos a Unity para reconocer este modelo cuando se active el marcador creado en vuforia. Al reconocer el marcador a través de la cámara del dispositivo inteligente se puede visualizar paralelamente el modelo *3D* del robot en el área delimitada para hacer las pruebas de adquisición de datos, con el fin de que el operario no interrumpa la visualización del marcador y se pierda el robot virtual. La Figura 62 se muestra el marcador y el modelo *3D* del robot vistos desde la cámara del dispositivo inteligente.

Figura 62. Marcador visto desde el dispositivo inteligente.

Para reproducir movimientos en los eslabones del modelo 3D del robot se debe generar desde *Visual Studio* un archivo de código en lenguaje *c# (script)* utilizando la función *"transform.Rotate()"* dentro del comando *"MoveRobot"*. A continuación, se muestra un fragmento del código para generar los movimientos en los eslabones.

```
private void MoveRobot() //Función
{
  //========= ESLABON 1 ==========//
    q1_r = q1 - q1_m; // q1= coordenada articular deseada; ...
                      // q_1m = coordenada articular anterior;
                      //q1_r = ángulo obtenido para llevar el eslabón a la posición
                      //deseada
        Eslabon_1.transform.Rotate(0.0f, 0.0f, -q1_r);
        q1_m = q1; // guarda el valor anterior

        //========= ESLABON 2 ==========//
        q2_r = q2 - q2_m;
        Eslabon_2.transform.Rotate(q2_r, 0.0f, 0.0f);
        q2_m = q2;

        //========= ESLABON 3 ==========//
        q3_r = q3 - q3_m;
        Eslabon_3.transform.Rotate(q3_r, 0.0f, 0.0f);
        q3_m = q3;

        //========= ESLABON 4 ==========//
        q4_r = q4 - q4_m;
        Eslabon_4.transform.Rotate(q4_r, 0.0f, 0.0f);
        q4_m = q4;

}
```

6.3 Comunicación para la conexión de la realidad aumentada.

Para conectar la computadora con el dispositivo inteligente y comunicar la plataforma con la realidad aumentada de deben seguir los siguientes pasos:

1. El operador encargado de generar los movimientos que luego son transformados en trayectorias lleva consigo unas gafas de realidad aumentada donde se inserta el teléfono inteligente con el que se visualiza el marcador para mostrar el modelo *3D* del robot y retroalimentar el sistema. La Figura 63 muestra las gafas de realidad aumentada que utiliza el operador cuando está generando la trayectoria.

Figura 63. Gafas de realidad aumentada.

2. Los movimientos del operador son capturados por la cámara Kinect y convertidos en coordenadas *(X, Y, Z)* en un espacio de trabajo definido previamente para ser enviados a la computadora y procesarlos con el algoritmo de adquisición de datos.

3. En la computadora se procesan los datos mediante el uso del modelo cinemático del robot para obtener las coordenadas articulares enviadas mediante una trama de información al dispositivo inteligente a través del protocolo de comunicación *UDP*. La configuración de la trama queda de la siguiente manera:

```
function unity_send(q,efector_final,p_io,AR_bit)
q=q*180/pi;
q=round(q,0);
q=num2str(q);
q1=q(1,:);
q2=q(2,:);
q3=q(3,:);
q4=q(4,:);
efector_final=efector_final*0.001;%convertir medidas de mm a metros
x=num2str(efector_final(1,1));
y=num2str(efector_final(2,1));
z=num2str(efector_final(3,1));
punto_io=num2str(p_io);
AR_bit=num2str(AR_bit);%bit de la función de la interfaz en Realidad aumentada
trama=strcat(q1,',',q2,',',q3,',',q4,',',x,',',y,',',z,',',punto_io,',',AR_bit
);
judp('SEND',8050,'192.168.137.161',int8(trama));
```

4. Los datos recibidos por el dispositivo inteligente son interpretados a través de una aplicación desarrollada en Unity que incorpora un *Script* de escrito en leguaje *c#* que interpreta la trama recibida y la convierte en los datos necesarios para introducir a la función *MoveRobot()*. A continuación, se describe el código con el que se interpreta la trama:

```
byte[] data = client.Receive(ref remoteEndPoint);
string text = Encoding.UTF8.GetString(data); // transforma el dato en texto
char[] ident = { ',', ',', ',', ',', ',', ',', ',' ,','};// caracteres para
//separar el mensaje
Int16 cot = 8;//número de datos
string[] matriz = text.Split(ident, cot);// se convierte text en una matriz de
                                         //strings llamada matriz
q1 = float.Parse(matriz[0], CultureInfo.InvariantCulture);//obtener datos float
q2 = Convert.ToSingle(matriz[1]);
q3 = Convert.ToSingle(matriz[2]);
q4 = Convert.ToSingle(matriz[3]);
x_wrist = Convert.ToSingle(matriz[4]);
y_wrist = Convert.ToSingle(matriz[5]);
z_wrist = Convert.ToSingle(matriz[6]);
point_io= Convert.ToSingle(matriz[7]);
```

5. Entre los datos interpretados por el dispositivo inteligente se encuentran las componentes de la muñeca derecha encargada de generar los movimientos de la trayectoria y un dato de confirmación para saber si esta fuera o dentro del espacio de trabajo del robot, la aplicación cargada en el dispositivo inteligente le da al operador una retroalimentación de los movimientos reflejados en el robot para hacer las respectivas correcciones cuando el movimiento se encuentre fuera del espacio de trabajo marcándolo como un punto rojo en la trayectoria. La Figura 64 muestra la representación de los movimientos que se encuentran fuera y dentro del espacio de trabajo del robot vistos desde el dispositivo inteligente.

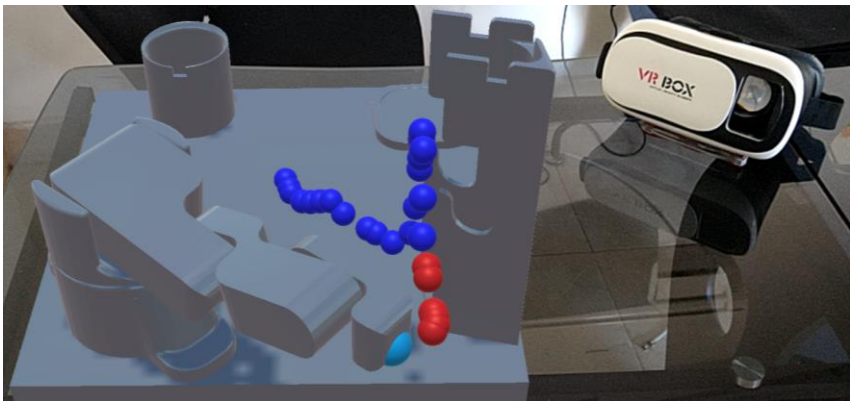


Figura 64. Simulación en realidad aumentada del robot ejecutando la trayectoria.

La Figura 65 detalla el diagrama de comunicaciones de los elementos involucrados en el control por guiado gestual usando la realidad aumentada. El Kinect captura los movimientos del operador y los transforma en coordenadas cartesianas, después el ordenador recibe mediante comunicación serial, procesa y guarda estos datos, que luego son enviados nuevamente al operador mediante comunicación UDP a la aplicación de realidad aumentada instalada en el dispositivo inteligente, que le permite al operador observar la trayectoria que

se está generando con sus movimientos. Con el software matemático se aplica el control cinemático para transformar las coordenadas cartesianas en articulares que se envían al robot o a la aplicación de realidad aumentada para observar la ejecución de la trayectoria.

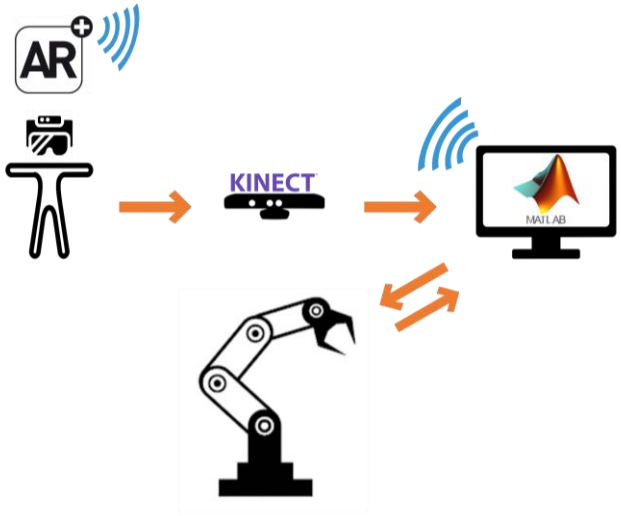


Figura 65. Comunicación para generar la realidad aumentada.

Cabe destacar que la visualización del movimiento no es solo para la retroalimentación a través del dispositivo inteligente utilizando la realidad aumentada, sino que también se puede visualizar en la interfaz desarrollada para el usuario a través de la computadora para ser supervisada por otro operador diferente al que está realizando los movimientos. Una vez terminada la captura de los movimientos se genera la trayectoria con el algoritmo y se ejecuta la simulación que se visualiza en ambas plataformas (la de realidad aumentada y la de la interfaz de usuario). La Figura 64 muestra la simulación de la trayectoria en el dispositivo inteligente y la Figura 66 muestra la simulación en el entorno de desarrollo de la computadora.

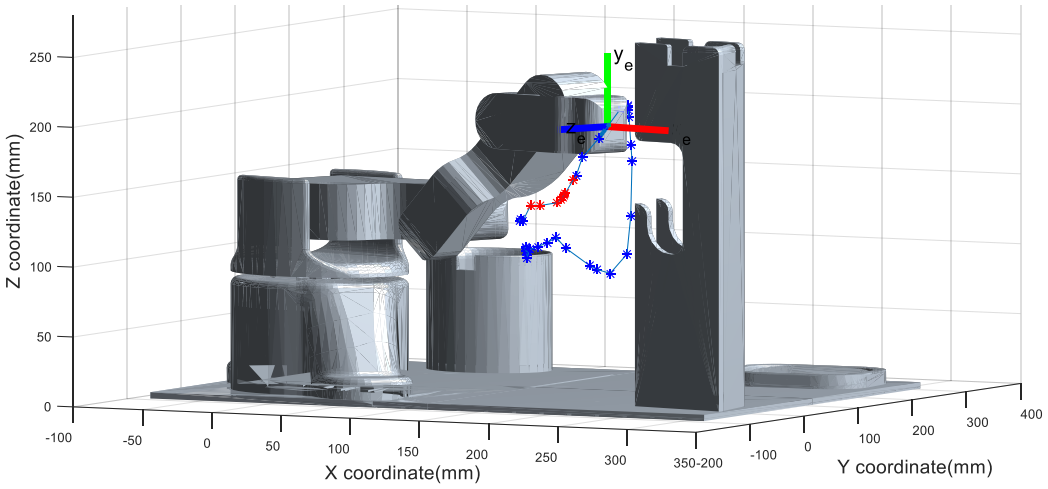


Figura 66. Simulación de la trayectoria en la computadora.

7. RESULTADOS

Para obtener los resultados de la implementación de este proyecto se van a realizar diferentes pruebas a las que se les aplicaran los algoritmos creados para la optimización de la trayectoria generada a partir del movimiento de un operador, en la Figura 67 a) se muestra la base elaborada por el usuario para las pruebas y calibración con sus respectivas medidas, diseñada a partir de las dimensiones del *CAD* de la base del robot *Raplim* detalladas en la figura b).

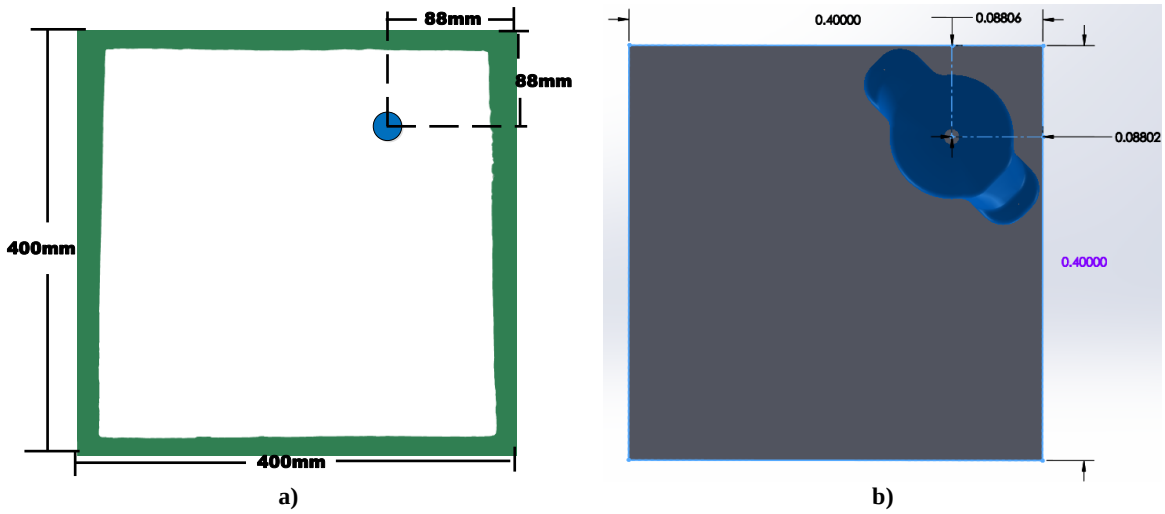


Figura 67. Dimensiones de la base del robot a) Base del robot elaborada para las pruebas y calibración y b) CAD de la base del robot en metros.

Las pruebas realizadas estuvieron sujetas a un modelo previamente establecido por el desarrollador del proyecto basado en la base del robot de la Figura 67 a) donde se ubican los modelos de prueba de la Figura 68 con sus respectivas medidas, es decir encima de la base del robot se ubican las formas de prueba en color azul que guían los movimientos del operador.

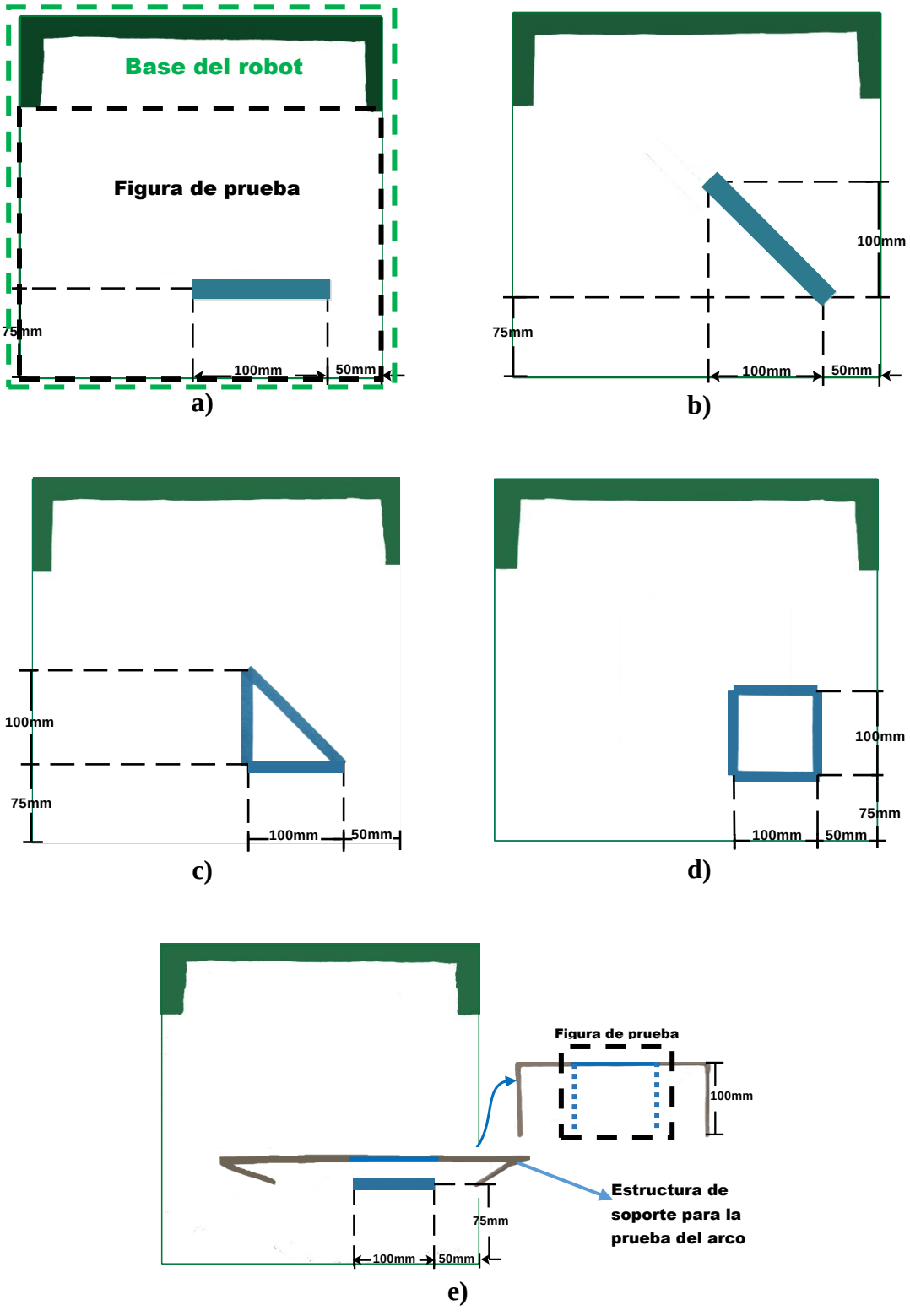
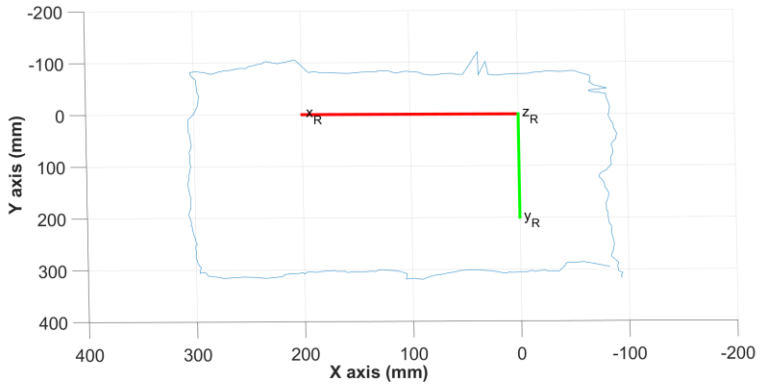
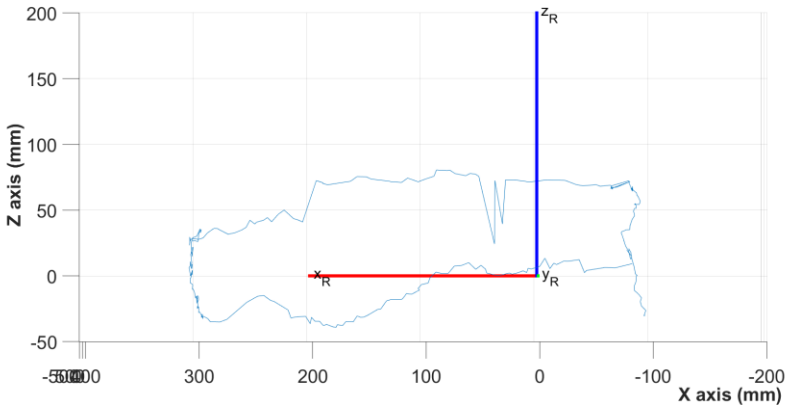


Figura 68. Figuras de prueba. a) Línea recta, b) Diagonal, c) Triangulo, d) Cuadrado, e) Arco.

Antes de realizar cada prueba se debe calibrar el sistema como se menciona en el capítulo anterior, una vez calibrado se realiza una prueba para verificar dicha calibración recorriendo la base del robot representada en la Figura 67 a). Si el recorrido no es tan preciso como al usuario le gustaría se pueden ajustar los puntos de la muestra en la interfaz de la Figura 59 hasta lograr una captura del recorrido satisfactoria. Durante las tomas se observan algunas variaciones de aproximadamente $50\ milimetros$ en el recorrido de la Figura 69 a) y b) hecho sobre la base del robot.



a)



b)

Figura 69. Recorrido de la base a) Vista de techo b) Vista lateral.

Como método para mejorar este comportamiento se plantea el uso opcional de tres filtros para corregir este error explicados con detalle a continuación:

- **Filtro mediana:** Agrupa de forma equitativa los datos de la muestra para evaluarlos y determinar el valor central, es decir, evalúa cada grupo de datos pertenecientes a la muestra y halla el valor central de forma consecutiva, para este caso se van a tomar grupos de 4 datos para dividir la muestra. La fórmula del cálculo de la mediana se presenta en la ecuación 126.

$$d_1 < d_2 < d_3 < d_4 \ \therefore \ M_c = \frac{d_2 + d_3}{2} \tag{126}$$

Donde d_1, d_2, d_3 y d_4 corresponden a los datos de cada grupo y M_c es el resultado del cálculo de la mediana para cada grupo.

- **Filtro no lineal:** Este procedimiento se aplica para encontrar valores pico en la muestra y reemplazarlos por el promedio de sus puntos adyacentes. Para este procedimiento se deben tener en cuenta varias condiciones que permitan encontrar los puntos irregulares que se encuentran en la muestra, se escogen grupos de 5 puntos consecutivos para determinar si el dato central tiene un valor pico para reemplazarlo por el promedio de 4 o 2 puntos del conjunto y equilibrar ese dato. La función mediante la cual se trabaja este filtro de datos es la siguiente.

```
function c = fun_eliminacionPicos(x)

%Determinar si el dato central es el mayor
if (x(3) > x(1)) & (x(3) > x(2)) & (x(3) > x(4)) & (x(3) > x(5))
    c = mean([x(1),x(2),x(4),x(5)]); %Se reemplaza por el promedio de los
datos adyacentes
%Determinar si el dato central es el menor
```

```
elseif (x(3) < x(1)) & (x(3) < x(2)) & (x(3) < x(4)) & (x(3) < x(5))
    c = mean([x(1),x(2),x(4),x(5)]); %Se reemplaza por el promedio de los
datos adyacentes
%Determinar si el dato central es mayor a los datos de los extremos
elseif (x(3) > x(1)) & (x(3) > x(5))
    c = mean([x(1),x(5)]); %Se reemplaza por el promedio de los datos de
los extremos
%Determinar si el dato central es mayor a los datos de los extremos
elseif (x(3) < x(1)) & (x(3) < x(5))
    c = mean([x(1),x(5)]); %Se reemplaza por el promedio de los datos de
los extremos
else
    c = x(3); %Si no se cumple ninguna de las anteriores se deja el dato
igual
end
```

- **Discretizar:** Otra alternativa para mejorar la muestra es discretizar los puntos a un valor determinado "Delta" que el usuario puede escoger para aplicarlo, este registro ayuda a que los puntos que estén por encima o por debajo de este valor alcancen un nivel estándar y se corrijan estas variaciones. El "Delta" que se escoge para realizar las pruebas tiene un valor de *5 milímetros* y se representa en la ecuación 127.

$$d_m = redondeo\left(\frac{dato}{Delta}\right)(Delta) \tag{127}$$

Donde d_m corresponde al dato muestreado, "$redondeo$" aproxima el resultado de la división a la parte entera más proxima, $dato$ es el valor correspondiente a la componente en Z de cada punto y $Delta$ es una constante de *5* milímetros.

Para cada tipo de prueba se realizan *50* muestras que son evaluadas en la interfaz de usuario para obtener resultados aplicando los filtros y los algoritmos de optimización. La interfaz de usuario está representada en la Figura 70.

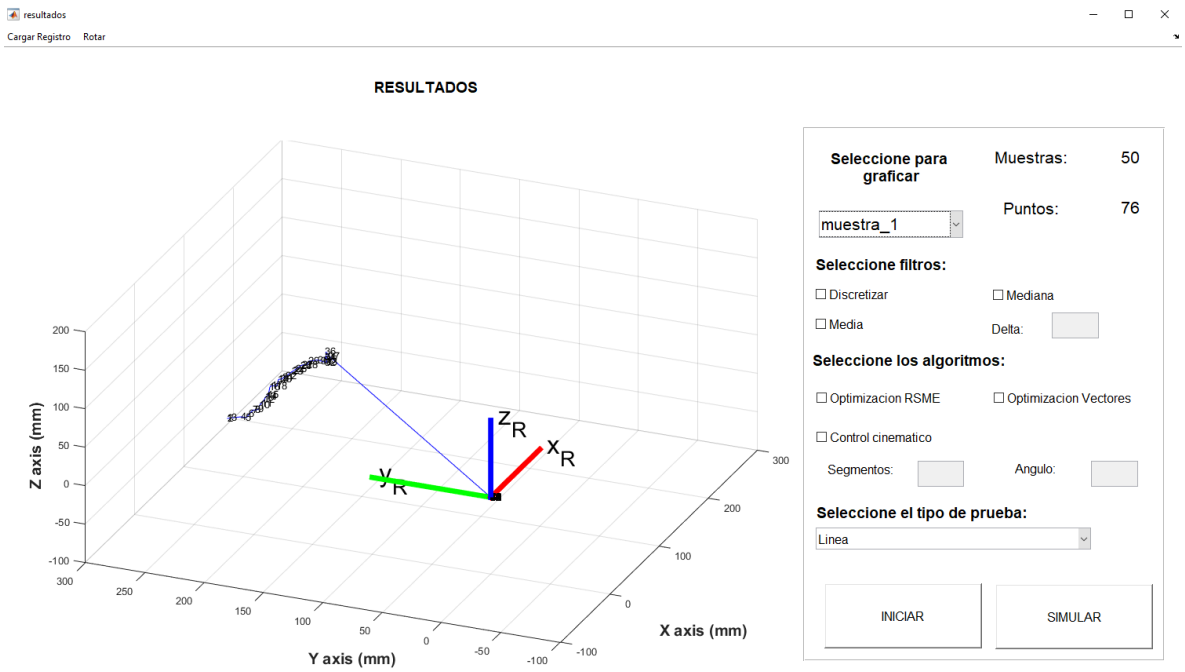


Figura 70. Interfaz de usuario para obtener los resultados.

La interfaz permite al usuario seleccionar el tipo de filtro y el algoritmo de optimización para obtener los resultados de las pruebas. Los resultados se obtienen aplicando todos los filtros a las *50* muestras de cada figura y dependiendo del orden en el que se apliquen varia la respuesta. En este proyecto se propone un orden para obtener los resultados basados en los ensayos previos con los que se obtienen los mejores resultados. El orden de los filtros para las muestras este dado de la siguiente manera:

1. Filtro no lineal (grupos de 5 datos).

2. Filtro mediana (grupos de 4 datos).

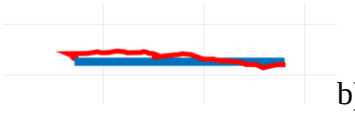
3. Discretizar (Delta de 5mm).

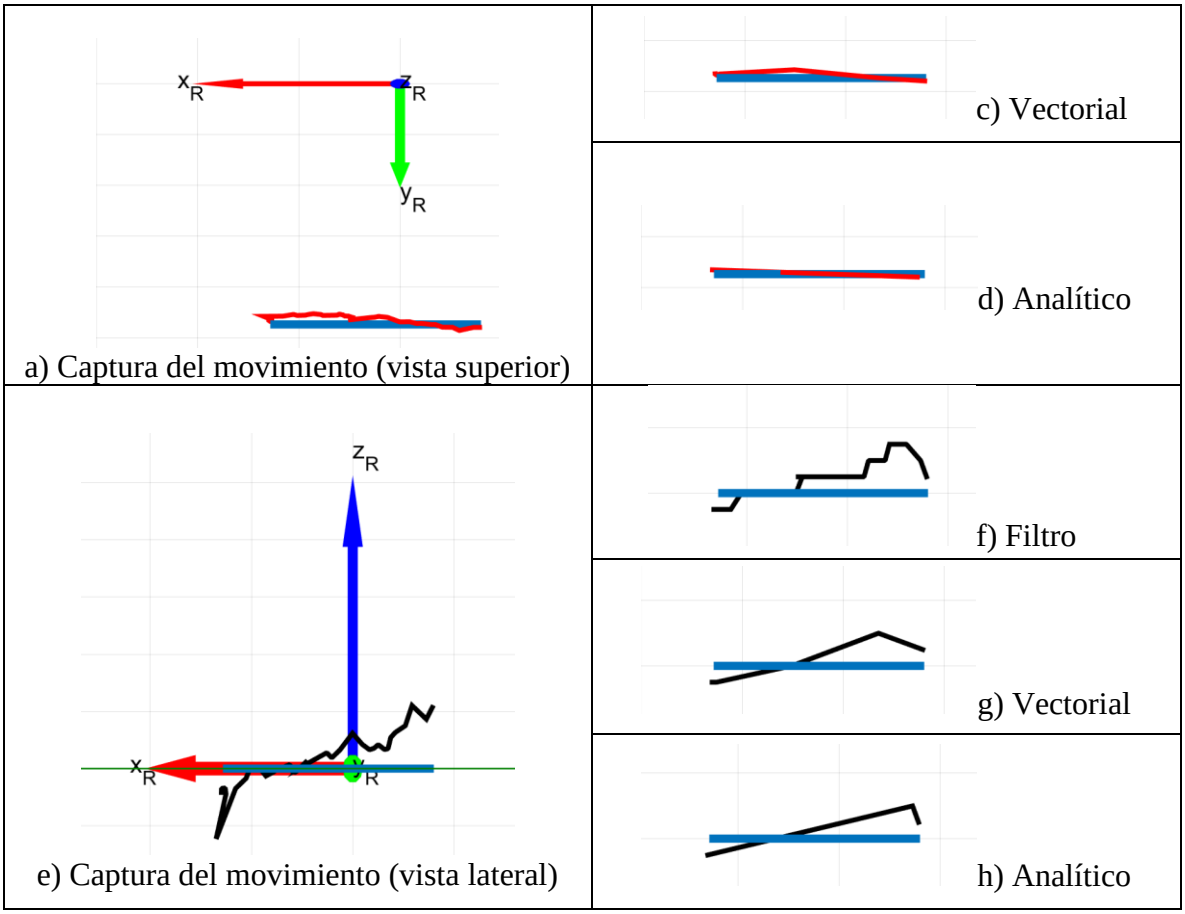
Para dar los resultados de cada prueba primero se aplican los filtros en el orden establecido los cuales entregan las muestras filtradas a las que posteriormente se le aplican los algoritmos de optimización e interpolador splin cúbico para cada una de ellas obteniendo *50* resultados por prueba en cada algoritmo.

Luego de aplicar todos los filtros y algoritmos a las muestras para obtener los resultados en el espacio de la tarea, estos se comparan con el modelo de prueba elaborado por el usuario de las formas de la Figura 68. Los resultados para cada prueba se presentan en formato de tabla, donde cada letra representa a) La captura del movimiento, b) Los resultados con el filtro, c) Los resultados con el algoritmo de vectores, d) Los resultados con el algoritmo lineal, todos los anteriores con vista superior. Los resultados con vista frontal están representados como e) La captura del movimiento f) Los resultados con el filtro, g) Los resultados con el algoritmo de vectores y h) Los resultados con el algoritmo lineal.

La Tabla 6 corresponde a los resultados de una de las 50 muestras realizadas para la prueba de la figura línea recta.

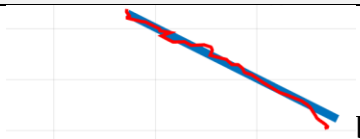
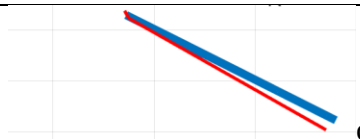
Tabla 6.
Registro de los resultados de una de las muestras correspondientes a la línea recta.

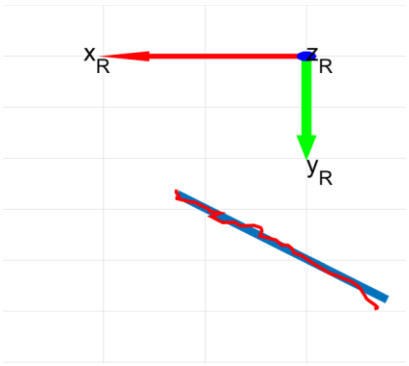
REGISTRO DEL MOVIMIENTO	**RESULTADOS**
	 b) Filtro



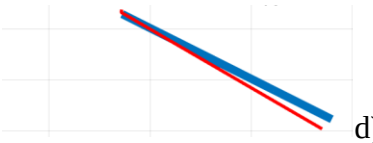
La Tabla 7 corresponde a los resultados de una de las 50 muestras realizadas en la prueba de la figura diagonal.

Tabla 7.
Registro de los resultados de una de las muestras correspondientes a la diagonal.

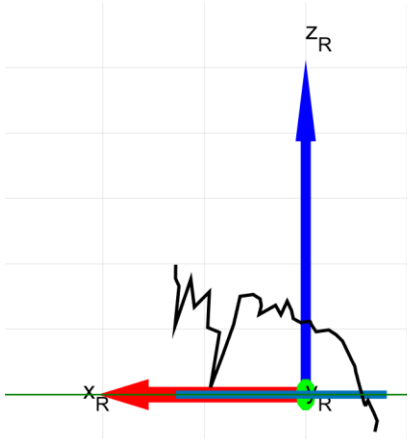
REGISTRO DEL MOVIMIENTO	RESULTADOS
	 b) Filtro
	 c) Vectorial



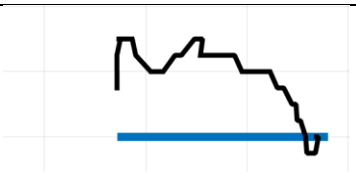
a) Captura del movimiento (vista superior)



d) Analítico



e) Captura del movimiento (vista lateral)



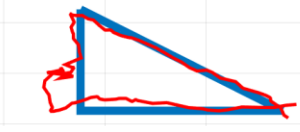
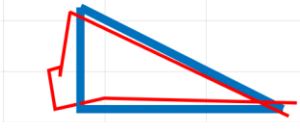
f) Filtro

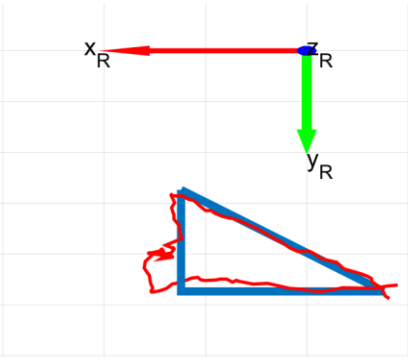
g) Vectorial

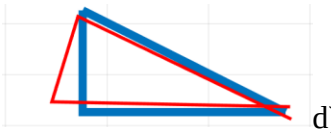
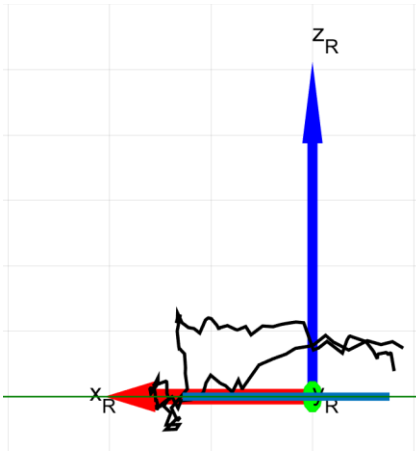
h) Analítico

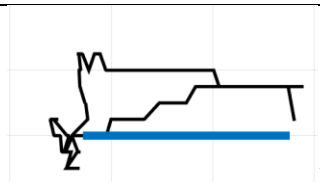
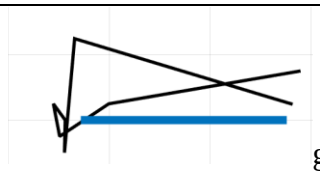
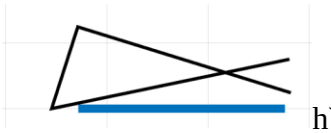
La Tabla 8 corresponde a los resultados de una de las 50 muestras realizadas en la prueba de la figura del triángulo.

Tabla 8.
Registro de los resultados de una de las muestras correspondientes al triangulo.

REGISTRO DEL MOVIMIENTO	RESULTADOS
	 b) Filtro
	 c) Vectorial

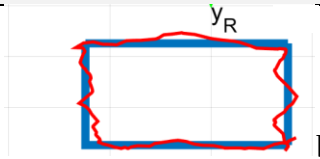
<table>
<tr>
<td> 

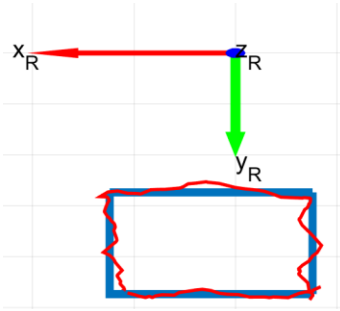
a) Captura del movimiento (vista superior)</td>
<td> 
d) Analítico</td>
</tr>
<tr>
<td rowspan="3"> 

e) Captura del movimiento (vista lateral)</td>
<td> 
f) Filtro</td>
</tr>
<tr>
<td> 
g) Vectorial</td>
</tr>
<tr>
<td> 
h) Analítico</td>
</tr>
</table>

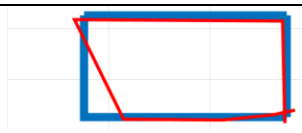
La Tabla 9 corresponde a los resultados de una de las 50 muestras realizadas en la prueba de la figura del cuadrado.

Tabla 9.
Registro de los resultados de una de las muestras correspondientes al cuadrado.

REGISTRO DEL MOVIMIENTO	RESULTADOS
	 b) Filtro

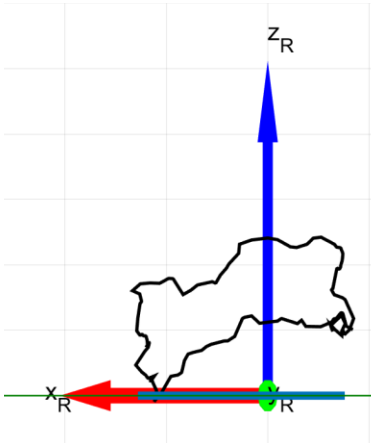


a) Captura del movimiento (vista superior)

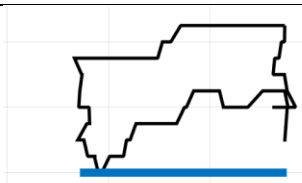


c) Vectorial

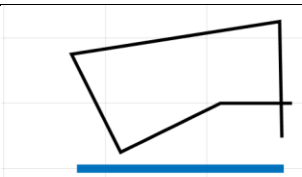
d) Analítico



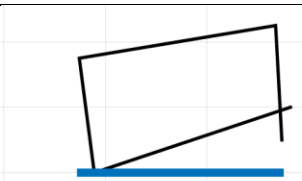
e) Captura del movimiento (vista lateral)



f) Filtro



g) Vectorial



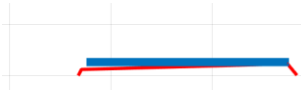
h) Analítico

La Tabla 10 corresponde a los resultados de una de las 50 muestras realizadas en la prueba de la figura del arco.

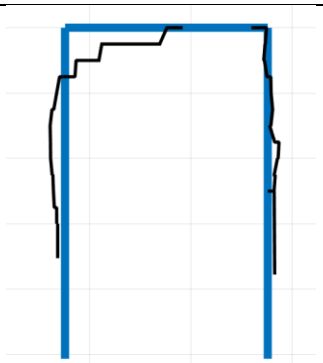
Tabla 10.
Registro de los resultados de una de las muestras correspondientes al arco.

REGISTRO DEL MOVIMIENTO	RESULTADOS
	 b) Filtro
	 c) Vectorial

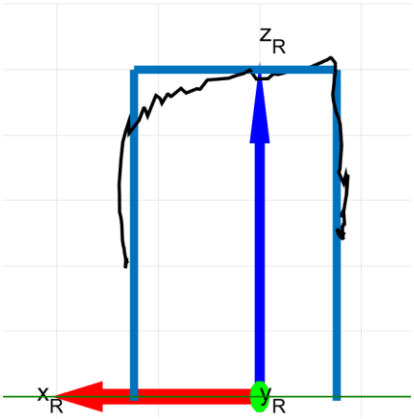
a) Captura del movimiento (vista superior)



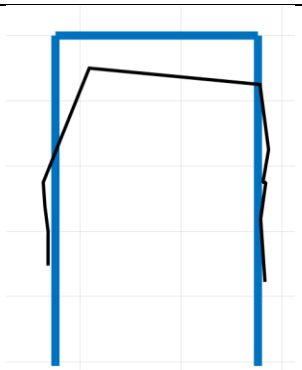
d) Analítico



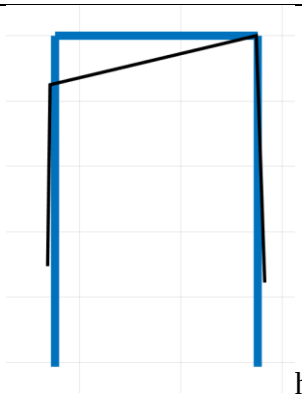
f) Filtro



e) Captura del movimiento (vista lateral)



g) Vectorial



h) Analítico

En las tablas de la 6 a la 10 se presentan los resultados de las muestras que menor error presentaron en cada una de las pruebas, para ilustrar los cambios que se generan a partir de aplicar los diferentes algoritmos.

Como en el algoritmo de optimización analítico se deben procesar varias combinaciones dependiendo del número de puntos y segmentos a calcular de acuerdo a la ecuación 117 presenta como consecuencia un tiempo elevado en el procesamiento de los datos, por lo tanto, para obtener los resultados se redujeron los puntos capturados a un máximo de *100* solo para este algoritmo mediante el proceso matemático de eliminación o promedio de puntos.

Estos procedimientos matemáticos solo se aplican cuando la cantidad de datos capturados supera los *100* puntos, es decir, si los puntos están entre *100* y *200* se agrupan de *2* en *2* si están entre *200* y *300* se agrupan de *3* en *3* y así sucesivamente, el método o procedimiento para la eliminación de datos conserva el primer dato del grupo y suprime los restantes y el método o procedimiento promedio de puntos promedia los puntos del grupo para convertirlo en uno solo.

Para dar los resultados al aplicar el algoritmo de optimización analítico se utiliza el método de eliminación para reducir la cantidad de puntos capturados en la muestra.

A cada muestra se le aplica el *RMSE* (*Root Medium Square Error*) para verificar la exactitud del recorrido, este cálculo matemático para determinar el error ayuda al operador a validar la muestra del registro con la definida en el entorno real para generar la trayectoria. Los resultados para cada algoritmo se dividen en dos, el error de la medida con respecto a la muestra *(Error de captura)* y el error de la medida con respecto al modelo o figura de prueba *(Error modelo)*. El promedio de error de los resultados de las *50* muestras para cada prueba teniendo en cuenta los *3* ejes coordenados X, Y y Z se representan en el formato de la Tabla 11, donde el *error de captura* representa el error del resultado obtenido con respecto a la muestra original y el *error de modelo* representa el error del resultado obtenido con respecto a la figura de prueba o modelo.

El error de captura le permite al usuario observar la exactitud con que se redujeron los puntos al comparar los resultados de los algoritmos de optimización con la captura de la muestra, es decir, si el error tiende a cero significa que el algoritmo interpreta un movimiento similar al del operador.

Los resultados obtenidos de los algoritmos de optimización son los puntos con los que se va a generar la trayectoria que al compararlos con la figura de prueba se obtiene el valor del error de modelo, con este valor se interpreta que tan exacto es el movimiento capturado con respecto a la intensión del desplazamiento por parte del usuario en el espacio de la tarea (figura de prueba).

Tabla 11.
Resultados de las pruebas en el plano XYZ.

PRUEBA	REGISTRO		FILTRO		VECTORIAL		ANALÍTICO	
	Error de captura (mm)	Error de modelo (mm)	Error de captura (mm)	Error de modelo (mm)	Error de captura (mm)	Error de modelo (mm)	Error de captura (mm)	Error de modelo (mm)
LÍNEA RECTA	0	30.43	4.17	29.70	12.24	20.28	8.27	24.80
DIAGONAL	0	44.89	3.66	43.76	10.57	36.51	7.69	40.09
TRIANGULO	0	42.35	2.43	42.14	8.85	38.88	8.86	38.67
CUADRADO	0	40.86	2.42	40.67	9.41	37.24	8.26	38.37
ARCO	0	12.31	2.48	12.20	8.91	17.31	6.32	14.58

Nota: El error de captura es *0* porque se está comparando con la misma muestra.

En la Tabla 12 se registran el tiempo invertido por el ordenador para ejecutar los algoritmos, el tiempo promedio por muestra y el tiempo total al analizar las *50* muestras que se toman para cada prueba.

Tabla 12.
Tiempo de ejecución de los algoritmos.

PRUEBA	VECTORIAL		ANALÍTICO	
	Tiempo promedio muestra (s)	Tiempo total(s)	Tiempo promedio muestra (s)	Tiempo total(s)
LÍNEA RECTA	0.0055	0.2772	0.2204	11.0180
DIAGONAL	0.0065	0.3258	0.3092	15.4587
TRIANGULO	0.0119	0.5968	13.5292	676.46
CUADRADO	0.0143	0.7143	251.8412	12592.06
ARCO	0.0186	0.9312	15.0407	752.04

Con los resultados de la tabla 6 a la 10 se evidencia gráficamente que las capturas presentan una menor precisión en el plano *XZ* que en el plano *XY*, por esta razón en la Tabla 13 se muestran los resultados numéricos ajustando el *RMSE* para determinar el error en *R2* del plano *XY*, es decir, sin tener en cuenta la componente en *Z*.

Tabla 13.
Resultados de las pruebas en el plano XY.

PRUEBA	REGISTRO		FILTRO		VECTORIAL		ANALÍTICO	
	Error de captura (mm)	Error de modelo (mm)	Error de captura (mm)	Error de modelo (mm)	Error de captura (mm)	Error de modelo (mm)	Error de captura (mm)	Error de modelo (mm)
LÍNEA RECTA	0	5.038	0	5.038	3.121	5.018	2.913	4.768
DIAGONAL	0	5.045	0	5.045	3.754	4.433	3.487	4.533
TRIANGULO	0	4.457	0	4.457	4.411	4.398	4.144	3.940
CUADRADO	0	4.658	0	4.658	5.983	6.085	4.265	4.497
ARCO	0	6.979	0	6.979	3.322	6.367	2.811	6.599

Nota: El error de captura es 0 porque se está comparando con la misma muestra, en este caso el filtro también toma el valor de 0 ya que solo se aplica en el eje Z lo cual no genera cambios en los ejes restantes.

La Tabla 14 muestra el registro de los resultados obtenidos de las *50* muestras para la prueba de la figura de la línea recta que se usó para determinar los parámetros de esta misma figura que se encuentran en la Tabla 11, donde:

Puntos capturados (P.C.): Se refiere a la cantidad de puntos capturados del movimiento del operador al representar la figura de prueba.

Error de captura (E.C.): Representa la similitud de los puntos que componen cada muestra y los obtenidos de cada algoritmo con respecto a los puntos del movimiento capturado.

Error de modelo (E.M.): Representa la similitud de los puntos que componen cada muestra y los obtenidos de cada algoritmo con respecto a la figura de prueba.

Puntos algoritmo (P.A.): Se refiere a la cantidad de puntos obtenidos de cada algoritmo de optimización.

Puntos control cinemático (P.C.C.): Se refiere a la cantidad de puntos generados automáticamente cada 5 milímetros en el espacio de la tarea para cada uno de los segmentos formados a partir de los puntos obtenidos de cada algoritmo de optimización. A estos puntos se les aplica el control cinemático para generar las trayectorias del robot.

Tabla 14.
Resultados de la figura de prueba línea.

Toma #	REGISTRO			FILTRO		VECTORIAL				ANALÍTICO			
	P. C.	E.C. (mm)	E.M. (mm)	E.C. (mm)	E.M. (mm)	P.A.	P.C.C.	E.C. (mm)	E.M. (mm)	P.A.	P.C.C.	E.C. (mm)	E.M. (mm)
1	36	0	59.27	5.45	58.42	3	25	15.90	47.49	3	26	11.95	48.21
2	45	0	16.80	2.38	16.31	3	23	6.92	17.28	3	23	8.90	15.18
3	37	0	23.86	3.42	23.01	3	23	7.80	16.53	3	25	6.70	19.87
4	34	0	22.88	2.83	22.85	2	23	10.04	14.47	3	23	4.62	21.52
5	45	0	30.64	4.31	30.29	3	24	11.00	19.60	3	24	11.06	19.64
6	36	0	12.97	2.64	12.63	4	21	5.10	10.43	3	22	4.09	11.26
7	38	0	11.43	3.28	9.43	6	22	4.99	7.23	3	21	6.43	5.70
8	35	0	35.11	5.30	33.46	3	21	18.02	18.14	3	25	11.91	25.03
9	30	0	28.01	3.37	27.70	2	21	13.51	15.83	3	24	6.08	24.46
10	30	0	19.24	3.22	18.54	5	21	8.24	15.04	3	24	7.59	13.35
11	28	0	24.44	3.17	24.08	2	22	13.78	11.86	3	23	7.09	19.00
12	29	0	38.62	7.10	36.73	3	23	18.11	21.10	3	24	14.66	26.88
13	33	0	25.47	3.71	25.45	4	24	6.49	23.21	3	26	7.44	22.50
14	32	0	31.41	5.13	30.37	2	22	18.78	13.18	3	25	8.64	25.40
15	43	0	29.26	3.22	29.27	3	24	11.00	20.84	3	28	6.55	24.89
16	39	0	24.80	2.96	24.86	4	23	12.33	13.46	3	25	8.57	16.83
17	35	0	23.78	4.28	23.53	3	22	10.41	16.95	3	25	8.08	18.05
18	30	0	16.60	2.52	15.84	3	21	9.79	8.52	3	23	7.50	11.17
19	37	0	30.17	2.89	29.89	2	24	11.14	17.99	3	25	7.35	24.15
20	40	0	36.45	6.18	35.14	3	25	12.99	27.71	3	29	10.75	30.75
21	39	0	28.39	4.70	27.36	4	21	14.11	17.09	3	24	9.39	20.92
22	38	0	19.64	3.10	19.56	3	21	9.65	10.68	3	24	4.60	17.92
23	36	0	31.28	3.31	31.05	4	21	10.64	23.98	3	25	6.53	27.92
24	33	0	28.44	4.88	27.90	2	22	15.78	13.92	3	23	8.12	21.95
25	38	0	25.00	2.71	24.78	3	23	10.79	14.95	3	23	5.98	21.45
26	36	0	36.00	2.93	34.72	3	24	9.37	28.76	3	24	6.67	29.37
27	31	0	23.13	2.20	22.48	2	23	9.91	14.29	3	23	5.06	18.88
28	35	0	25.08	2.74	24.76	4	21	5.96	19.96	3	24	5.40	21.22
29	42	0	21.14	2.55	21.34	3	22	8.58	14.58	3	23	6.70	16.90
30	36	0	29.39	4.56	28.34	3	21	17.08	13.70	3	24	8.17	23.83
31	30	0	16.10	2.34	15.55	3	22	8.44	9.07	3	23	4.35	14.07
32	42	0	36.10	4.05	35.18	2	24	15.73	23.96	3	28	6.53	33.58
33	43	0	38.12	4.98	37.72	2	23	18.12	21.19	3	26	11.24	30.78
34	40	0	52.38	6.34	51.37	2	22	27.18	26.65	3	27	15.19	41.28
35	31	0	35.86	5.79	34.11	2	22	17.30	16.63	3	25	10.50	28.73
36	37	0	17.21	3.83	16.81	5	21	10.82	8.82	3	23	7.11	11.91

37	30	0	58.11	5.77	57.12	3	24	14.18	47.40	3	25	11.31	48.03
38	39	0	15.37	2.14	15.25	4	23	5.15	13.02	3	23	4.55	13.04
39	42	0	51.98	5.87	50.91	3	26	15.58	37.89	3	26	15.61	37.90
40	38	0	55.43	4.77	54.50	3	27	8.42	51.32	3	27	8.36	51.31
41	43	0	29.80	6.52	28.45	4	23	11.70	22.71	3	26	10.30	24.91
42	36	0	29.06	4.71	27.24	3	23	9.09	25.09	3	25	7.74	26.34
43	30	0	26.11	3.60	25.26	2	21	14.65	12.24	3	22	6.69	21.05
44	28	0	23.21	3.15	22.83	3	22	6.83	18.58	3	23	5.56	20.34
45	27	0	24.72	5.30	24.59	3	21	10.81	17.48	3	23	7.99	20.88
46	32	0	62.42	12.03	59.87	3	26	25.48	41.56	3	31	20.89	46.43
47	38	0	25.23	2.60	25.00	2	22	9.76	16.01	3	24	3.98	23.44
48	33	0	31.21	3.51	31.03	2	22	12.11	22.28	3	24	4.89	28.62
49	33	0	49.55	6.75	48.07	3	22	19.66	30.06	3	26	12.48	40.55
50	37	0	34.75	3.36	34.30	2	21	12.97	23.24	3	24	5.40	32.46

Nota: El valor de “0” en la columna de Error Captura (E.C.) en el registro se da porque se está comparando con respecto a la misma muestra. En el algoritmo filtro no se agregó la columna P.A. y P.C.C. ya que los puntos que se obtuvieron y usaron para el cálculo del control cinemático son los mismos capturados de cada muestra en el registro.

La toma *7* de la Tabla 14 subrayada en verde es la muestra que menor error presenta de las *50* tomas, con ella se graficaron los resultados de la Tabla 6. Los resultados de las muestras para cada una de las figuras se encuentran en el anexo *2* de este libro.

Como recurso computacional para realizar todo el procesamiento de datos se utilizó un equipo de cómputo portátil de marca ASUS con un procesador AMD Ryzen 7 3750H, 16 GB de RAM, tarjeta de video NVIDIA GeForce GTX1650 y sistema operativo Windows 10 de 64 bits.

Como ultimo procedimiento para poder generar las trayectorias y simular los resultados se deben obtener las coordenadas articulares aplicando el control cinemático visto en el capítulo *5.3* de este libro.

Obtenidas las trayectorias resultantes de cada prueba se cargan en la interfaz de usuario de la Figura 71 para simularlas y observar el comportamiento de las articulaciones del robot en un entorno virtual.

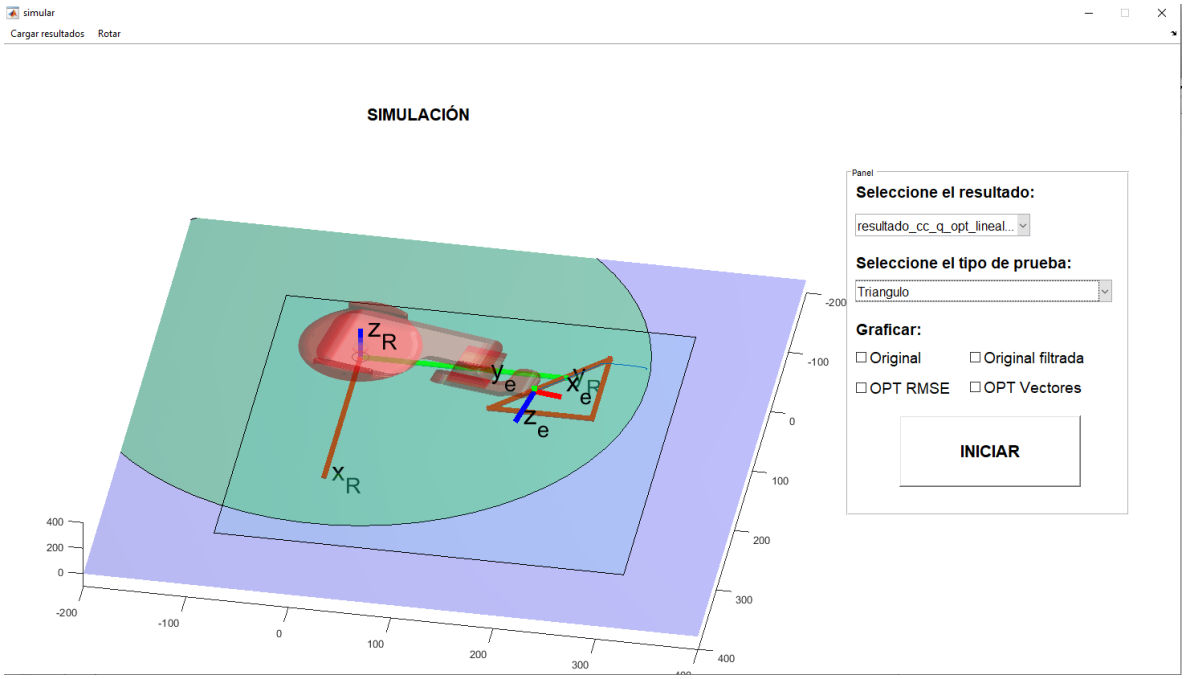


Figura 71. Entorno de simulación de trayectorias.

Con la simulación de las trayectorias se evidencia si el robot es capaz o no de realizar dichos movimientos, ya que al encontrarse con un punto no alcanzable se quedará inmóvil dando a entender que por sus características antropomórficas no le es posible seguir con ese tipo de movimiento.

Los movimientos del robot en el entorno virtual están representados por la línea verde punteada que se observa en la Figura 72, donde se evidencia que si realiza todo el recorrido salvo por una pequeña interrupción en uno de los vértices de la forma triangular a causa de las limitaciones mecánicas del robot. El recorrido representado por la línea negra punteada de la Figura 72 pertenece a las figuras d) y h) de la Tabla 8 que muestra el resultado del algoritmo analítico del cual se obtuvieron 4 puntos, donde el punto *3* no es alcanzable, por lo que el robot iría del punto *2* al *4* directamente, para evitar esto se agregaron cada *5 milímetros* puntos intermedios entre cada par de puntos. De esta manera se recorren segmentos donde alguno de sus puntos no sea alcanzable por el robot. Este procedimiento también se aplica para el algoritmo de optimización vectorial.

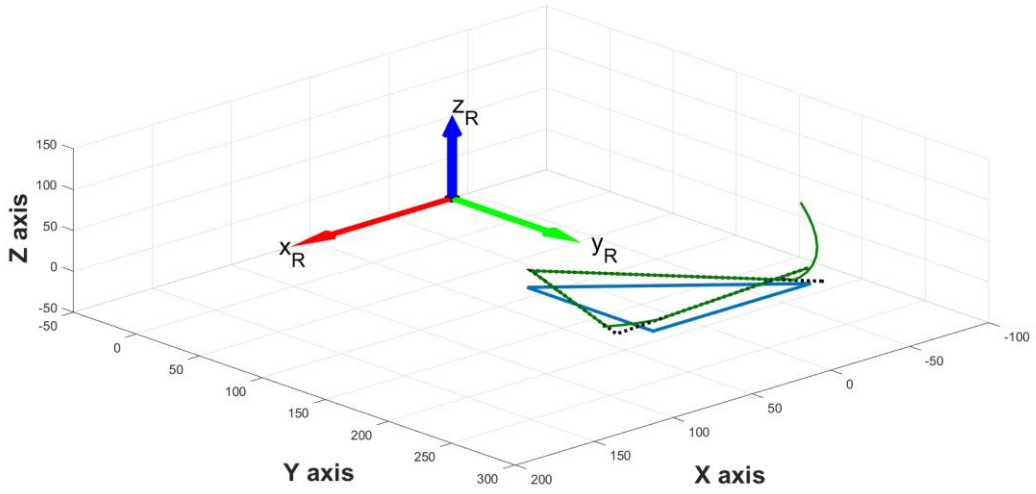
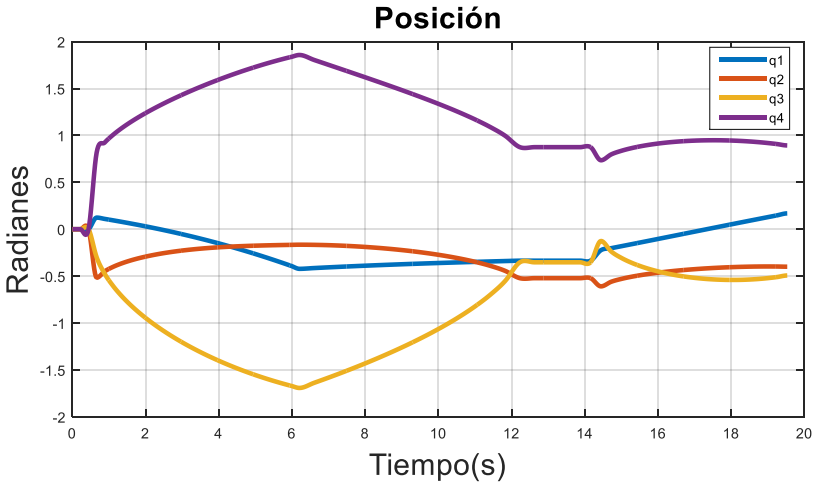


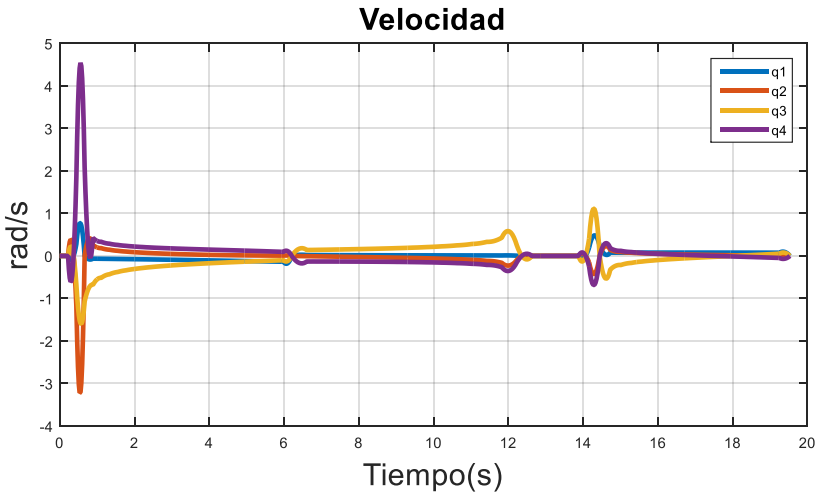
Figura 72. Representación gráfica de la trayectoria ejecutada por el robot (línea verde) con respecto a la línea negra punteada resultante de aplicar el algoritmo de optimización analítico.

En la Figura 73 se muestran las gráficas de posición articular, velocidad y aceleración de cada una de las articulaciones Vs tiempo (segundos) que realiza el robot para poder llevar a cabo la trayectoria punteada de color verde de la Figura 72, que describe el resultado de la

muestra capturada para la figura triangular después de aplicar el interpolador splin cúbico a las coordenadas articulares obtenidas de la cinemática inversa.



a)



b)

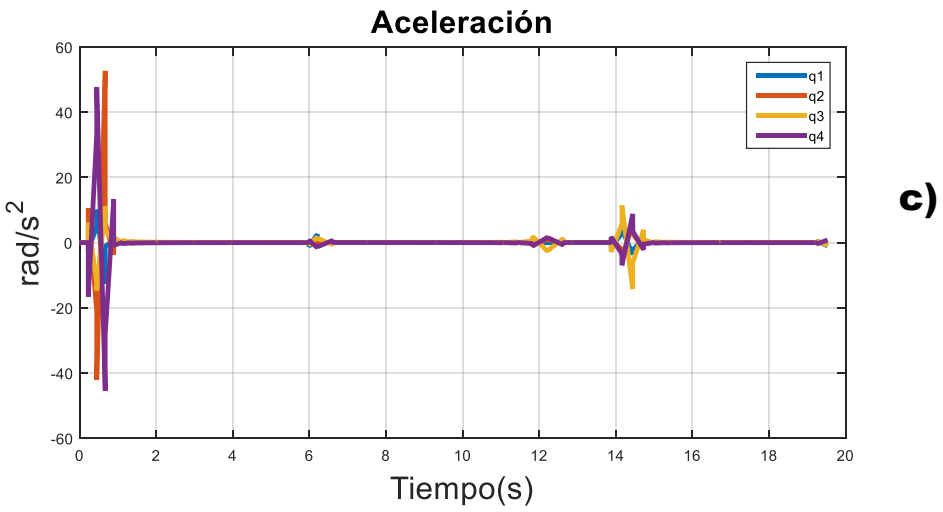


Figura 73. Representación gráfica del triángulo. a) Posición articular Vs tiempo, b) Velocidad Vs tiempo y c) Aceleración Vs tiempo.

Las trayectorias obtenidas al aplicar el interpolador splin cúbico a cada uno de los resultados de cada prueba correspondientes a las tablas de la *6* a la *10* se muestran en el anexo *3* de este libro.

Para mostrar los resultados en realidad aumentada de las trayectorias de cada prueba se utiliza un dispositivo inteligente con la aplicación de Unity, donde al identificar el marcador muestra el robot virtual ejecutando la trayectoria. En la Figura 74 se observa a través de una cámara el marcador y el robot virtual simulando la trayectoria de una de las pruebas.

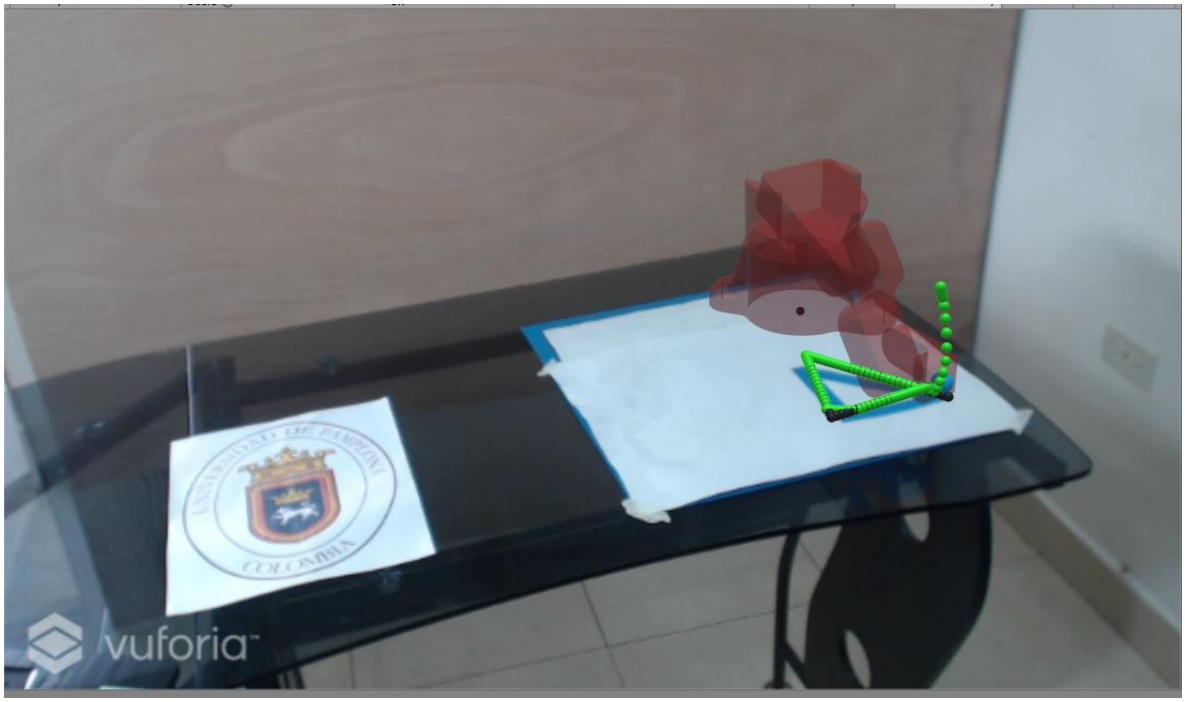


Figura 74. Simulación de una de las trayectorias en realidad aumentada.

8. CONCLUSIONES Y RECOMENDACIONES

A diferencia de los tipos de programación comunes (textual y por guiado) este sistema le facilita al usuario generar un trayectoria sin necesidad de hacer cálculos matemáticos, ya que el operador puede indicarla con una parte de su propio cuerpo, reduciendo la complejidad en el uso de robots industriales, además se podrían eliminar algunos componentes de los robots como lo son los sensores de fuerza, esenciales para la programación por guiado, disminuyendo los costos de fabricación y por ende los costos de adquisición de equipos robóticos.

Para la toma de datos se debe mantener una postura ideal con el fin de que el algoritmo de reconocimiento identifique correctamente las articulaciones y evitar sobreponer una articulación sobre la otra para no causar irregularidades en la lectura de los movimientos del operador y fallas en la activación de los comandos y captura de los movimientos, adicionalmente la lectura por parte del sensor Kinect va a ser más exacta.

La cámara de luz estructurada tiene una alta sensibilidad al capturar los movimientos registrando hasta los más leves e involuntarios como las vibraciones, lo que puede generar picos en la trayectoria dañando la muestra, para evitar esto, el usuario debe tener movimientos suaves, firmes, con perspectiva y teniendo en cuenta su postura para que al registrar dichos movimientos en la interfaz de pruebas cumplan con lo necesario para ser guardados.

Al aplicar los algoritmos de optimización se logra interpretar en un menor número de puntos en el movimiento que el usuario ejecuta, reduciendo la complejidad de la trayectoria del robot y el error que presenta la captura original con respecto al modelo de la muestra ya que los movimientos del usuario capturados por el Kinect generan una gran cantidad de datos que aumenta la carga computacional.

La selección del algoritmo de optimización por parte del usuario depende del tipo de trayectoria que se va a analizar por ejemplo, el algoritmo analítico presenta mejores resultados siendo menos sensible a los picos de error que se presentan en la captura de la muestra aunque para las trayectorias de más de *2* segmentos aumenta considerablemente el tiempo en el procesamiento de los datos, 0.31s para dos, 14s para 3 y 252s para 4 segmentos, en cambio el algoritmo vectorial es capaz de analizar trayectorias complejas en menor tiempo, máximo 0.02s para cualquier trayectoria, presentando resultados favorables pero con mayor error ya que pueden generar segmentos adicionales en los picos que hacen que la trayectoria no se vea tan definida.

En las pruebas se evidencian variaciones en la identificación de la articulación seleccionada para la captura del movimiento por parte del Kinect, provocando cambios de más de *50mm* en la componente *Z* aunque la muestra se realizaba sobre una superficie plana sin elevar la articulación (muñeca derecha) y algunas muestras se graficaban por debajo de esta superficie por lo que no era posible que la muestra fuera simulada. Para evitar este efecto

en la muestra se recomienda emplear una versión actualizada del sensor Kinect (Kinect V2) u otro dispositivo que entregue lecturas más estables y permita identificar más articulaciones, por ejemplo, si el operador pudiese realizar sus movimientos con su dedo índice tendría mayor control y precisión al momento de indicar la trayectoria.

Durante las simulaciones de los resultados se evidencia que al robot se le dificulta seguir las trayectorias por su área de trabajo reducida debido a su tamaño y los pocos grados de libertad, restringiendo el tipo de pruebas que se pueden realizar. Si se aplicara este proyecto a otro tipo de robot, se recomienda utilizar un robot tipo SCARA o un antropomórfico con mayores grados de libertad para observar el alcance del desarrollo de este proyecto.

A causa de la nueva normalidad debido a la pandemia del Covid-19 no se pudieron realizar pruebas en el robot real, ya que este se encuentra dentro de las instalaciones de la Universidad de Pamplona, por tanto, la influencia de las restricciones por parte del robot se observa en el anexo 3 a través de los recorridos de la trayectoria ejecutada por el robot en simulación. De esta manera se pretende en un futuro programar estas trayectorias en el robot real con el fin de reproducir el movimiento capturado y validar las trayectorias generadas con respecto a la simulación.

Para fututos trabajos se puede adquirir una cámara Intel RealSense (D435) que permite capturar las articulaciones con un error de 2 milímetros y detectar objetos a una

distancia de 10 metros, además puede reconocer las articulaciones de los dedos que permitiría al usuario trazar una trayectoria con mayor comodidad y precisión, de esta manera se podrían evaluar nuevamente los algoritmos desarrollados en este proyecto validando la importancia de los dispositivos de captura para conseguir programar las trayectorias de un robot utilizando la programación por guiado gestual.

PUBLICACIONES Y PONENCIA

Producto del desarrollo de este trabajo se lograron dos publicaciones en revistas categoría B y una ponencia en el congreso internacional de electrónica y tecnología de avanzada del año 2020 (CIETA) de las cuales se presentan las siguientes evidencias.

El artículo titulado "*Gestual Guidance Control for an Anthropomorphic Robot Based on Artificial Vision Techniques*", escrito por Andrés Felipe Padilla Matilla, César Augusto Peña Cortés y Gonzalo Guillermo Moreno Contreras fue publicado en el año 2020 en la revista *International Journal of Advanced Science and Technology (IJAST)*, categoría B en el año de publicación.

Abstract: This article presents the development and implementation of an algorithm that allows evaluating the movements of an operator, which are captured by a structured light camera, generating a series of points that indicate the trajectory that the robotic arm must execute, this procedure is based on artificial vision techniques. The results are displayed through a user interface that allows the operator's movements to be graphically replicated through points that indicate whether or not they belong to the robotic manipulator's workspace, the corroboration is done through the simulation of the robot traversing the path obtained. The captured movement shows fluctuations in the plotted trajectory, showing points with a slight margin of error in the segments generated from them. To smooth this movement out, intermediate points are added between each pair of points, improving the relationship between speed and precision without ruling out those generated by the variation in the involuntary movements of the operator when executing the trajectory.

International Journal of Advanced Science and Technology
Vol. 29, No. 5, (2020), pp. 14164 - 14177

Gestual Guidance Control for an Anthropomorphic Robot Based on Artificial Vision Techniques

Ing. Andrés Felipe Padilla Mantilla[1]*, PhD. César Peña[1], PhD. Gonzalo Moreno[2]

[1]*University of Pamplona*
Department of Mechanical, Mechatronics and Industrial Engineering
Research Group Automation and Control.
E-mail: {andres.padilla, cesarapc}@unipamplona.edu.co.

[2]*University of Pamplona*
Department of Mechanical, Mechatronics and Industrial Engineering
E-mail: gmoren@unipamplona.edu.co.

Abstract

This article presents the development and implementation of an algorithm that allows evaluating the movements of an operator, which are captured by a structured light camera, generating a series of points that indicate the trajectory that the robotic arm must execute, this procedure is based on artificial vision techniques. The results are displayed through a user interface that allows the operator's movements to be graphically replicated through points that indicate whether or not they belong to the robotic manipulator's workspace, the corroboration is done through the simulation of the robot traversing the path obtained. The captured movement shows fluctuations in the plotted trajectory, showing points with a slight margin of error in the segments generated from them. To smooth this movement out, intermediate points are added between each pair of points, improving the relationship between speed and precision without ruling out those generated by the variation in the involuntary movements of the operator when executing the trajectory.

Keywords: *Gesture guidance, trajectory, algorithm, robot, interface, structured light camera.*

1. Introduction

Industrial robots were made to collaborate with physically demanding work, repetitive and high-risk tasks for man, but today they are endowed with the ability to replace the human being almost completely in any activity, even those that require impeccable precision like the area of medicine where human life is put at risk. [1].

The structures of these robots present certain similarities to the human being, some in their entirety and others only to their extremities, as is the case of robotic arms, although their physical structure does not depend on how closely they resemble known physiognomies, but rather the ability to carry out a certain effort or task and ensuring that the results are equal to or better than those of a person [2].

Most of the large industries are forced to design intelligent machines that solve specific problems in different work areas, some of the manufacturing companies of this type of technology are ABB® and KUKA®, which have worldwide recognition in the

[1]*Andrés Felipe Padilla Mantilla

El artículo titulado "*Advances in industrial robots programming applying gestural guidance techniques*", escrito por Andrés Felipe Padilla Matilla, César Augusto Peña Cortés y Gonzalo Guillermo Moreno Contreras fue publicado en el año 2020 en la revista Journal *of Physics: Conference Series (JPCS)*, categoría B en el año de publicación.

Abstract: In this article, artificial vision is presented as an application for the generation of trajectories through the capture of an operator's movements using a structured light camera (Kinect) and applying a recognition algorithm to the body's joints, a three-dimensional representation is made of the skeleton, taking as points of interest the articulation of the left hand to generate the gestural commands and the right hand to indicate the path that the robot must execute. Simulation results are obtained through a development environment where the kinematic model of the robot is incorporated, this indicates that it is possible to generate a trajectory through the human body using artificial vision.

XIV International Congress of Electronics and Advanced Technologies (XIV CIETA) IOP Publishing
Journal of Physics: Conference Series **1704** (2020) 012001 doi:10.1088/1742-6596/1704/1/012001

Advances in industrial robots programming applying gestural guidance techniques

A F Padilla[1], **C A Peña**[1], **and G G Moreno-Contreras**[1]
[1] Grupo de Automatización y Control, Universidad de Pamplona, Pamplona, Colombia

E-mail: andres.padilla@unipamplona.edu.co, cesarapc@unipamplona.edu.co

Abstract. In this article, artificial vision is presented as an application for the generation of trajectories through the capture of an operator's movements using a structured light camera (Kinect) and applying a recognition algorithm to the body's joints, a three-dimensional representation is made of the skeleton, taking as points of interest the articulation of the left hand to generate the gestural commands and the right hand to indicate the path that the robot must execute. Simulation results are obtained through a development environment where the kinematic model of the robot is incorporated, this indicates that it is possible to generate a trajectory through the human body using artificial vision.

1. Introduction

Industrial robots have been around since the late 1950s and have evolved to suit the needs of modern industry. The first generation of robots consisted of complex mechanical systems that had to be manipulated by an operator and had a simple control system, the second generation brought learning robots, which repeat a sequence of movements previously executed by means of a code, the third generation introduces sensory devices that help the robot to interact with the work environment, and finally mobile or intelligent robots belong to the fourth generation, bringing us closer to artificial intelligence [1]. An industrial robot can adapt to various tasks in a company by reprogramming itself.

The programming of a robot is defined as the process by which it is indicated the sequence of actions that it must carry out during the performance of a certain task [2]. Therefore, programming is the tool that the user must program the robot to perform a specific task. The physical characteristics of the robot and the programming possibilities are directly related [2]. The two types of programming mostly used today are defined below. The first is guided or learning programming, which consists of manipulating the desired trajectory or points in which the robot is desired to move at the same time as adapted configurations are recorded for later automatic reproduction [3]. The second is textual programming, which uses a specific programming language to edit orders in the robot's tasks, which are subsequently executed [4].

In the development of this project, the guided programming method is used, including new technologies such as the structured light camera with the application of artificial vision techniques, to program the trajectory without the need to move the robot out of the operating line. This project is based on a user interface platform in which the trajectory is defined from the movements captured by Kinect under a reference system that simulates the trajectory of the robot, which must later be adjusted and saved for programming. For this application the Raplim robot was used, which is anthropomorphic and has 4 degrees of freedom, this robot was taken as a test model to implement the programming method

En el XIV congreso internacional de ingeniería electrónica y tecnologías avanzadas se realizó la ponencia que llevo a la publicación del articulo titulado "*Advances in industrial robots programming applying gestural guidance techniques*", Como se evidencia en el certificado que se muestra a continuación.

La República de Colombia, y en su nombre

UNIVERSIDAD DE **PAMPLONA**

FACULTAD DE INGENIERÍAS Y ARQUITECTURA

CERTIFICA QUE

ANDRES FELIPE PADILLA MANTILLA

CC . 1094269726

PONENTE

XIV CONGRESO INTERNACIONAL DE INGENIERÍA ELECTRÓNICA Y TECNOLOGÍAS AVANZADAS. X CONGRESO INTERNACIONAL EN SISTEMAS, INFORMÁTICA E INGENIERÍA DEL CONOCIMIENTO . X CONGRESO EN TELECOMUNICACIONES. IX CONGRESO EN INGENIERÍA ELÉCTRICA. V CONGRESO INTERNACIONAL DE INGENIERÍA MECATRÓNICA. II CONGRESO INTERNACIONAL DE INGENIERIA INDUSTRIAL

Realizado los días 28,29 y 30 de octubre de 2020

Con una intensidad de 24 horas

JORGE LUIS DIAZ RODRÍGUEZ
Decano Facultad de Ingenierías y Arquitectura

ALDO PARDO GARCÍA
Coordinador

55034

REFERENCIAS

Arce, C. (2014). *Realidad aumentada.* Paraguay.

Artopoulos, A., Bambozzi, L., Bastos, M., Beiguelman, G., Brissac Peixoto, N., Kyong Chun, W. H., . . . Yeregui, M. (2010). *Nomadismos Tecnológicos.* Buenos Aires: Ariel.

AUTOMÁTICA INDUSTRIAL. (26 de Diciembre de 2020). *AUTOMÁTICA INDUSTRIAL*. Obtenido de https://automaticaindustrial.wordpress.com/robotica/programacion-de-robot/

Barrientos, A., Peñin, L., Balaguer, C., & Aracil, R. (2007). *Fundamentos De Robotica.* Madrid: Mc Graw Hill.

Bayani, S., Rastegari, R., & Cheraghpour Samavati, F. (2015). Design of Hyper Redundant Robot using Ball screw . *IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, 1273.

Bermudez, G. (2002). Robot móviles. Teoria, aplicación y experiencias. *5*(10).

Cañas, J. M., Matellán, V., & Montúfar, R. (2006). Programación de robots móviles. *3*(2).

Chang Herrera, A. N. (2014). *Diseño y simulación de un robot articular con seis grados de libertad utilizando el toolbox robotics de Matlab para fortalecer las clases teoricas realizando practicas de laboratotio con el software presentado en este proyecto.* Quito: Escuela Politécnica Nacional.

Checa, D., Luna, D., & Mosquera, V. (2009). *Simulador de un robot SCARA de 4 grados de libertad basado en realidad virtual.* Cauca: Cuarto congreso Colombiano de computación.

Crus Lara, R. J. (2018). *Hibridación de algoritmo de programación matemática con algoritmos evolutivos en problemas de optimización de diseño mecatronico.* Veracruz: Lania.

Fernandez Yanez, L. A., & Sotomayor Reinoso, L. F. (2016). *Analisis Cinematico inverso y directo del robot paralelo.* Quito: Escuela Politécnica nacional.

Figueroa de godos, A. (2015). Diseño y verificación del sistema de interpolación de trayectorias de un mecanismo de cinemática paralela. (TFG 2014-17).

Garcia Calandín, L. I. (2008). *Modelado cinemático y control de robots móviles con ruedas.* Valencia, España.

García Luna, F., Rodrígues Ramírez, A., & Luviano Cruz, D. (2019). *Diseño de librería con enfoque didáctico para control de robots manipuladores en sistemas embebidos.* Mexico: Pistas educativas.

García Monsálvez, J. C. (2017). Python como primer lenguaje de programación textual en la enseñanza secundaria. *18*(12).

Garcia, J. M. (2015). Robótica Educativa. La programación como parte de un proceso educativo . (46).

Giraldo, L. F., Delgado, E., & Castellanos, G. (2006). Cinemática inversa de un brazo robot utilizando algoritmos genéticos. *3*(1).

Guaraca Medina, P. J., & Ochoa Ochoa, J. L. (2015). *Estudio de la programación y operación de los robots industriales Kuka KR16-2 y KR5-2 ARC HW.* Cuenca: Univerisdad politécnica salesiana sede cuenca.

Hernández Herrera, V. (2012). *Localización de un robot móvil empleando memoras asociativas ALFA-BETA.* México, D. F.

Hernandez Matías, J. C., & Vizán Idoipe, A. (2015). *Sistemas de automatización y robótica para las pymes españolas.* Madrid: Fundación EOI.

Hernández Méndez, S. (2011). *Determinación y localización espacial de objetos geométricos simples para la manipulación por un robot móvil autónomo.* Xalapa, Veracuz: Universidad de Veracruz.

Jara Bravo, C. A. (2020). *Aportaciones al aprendizaje constructivo y colaborativo en internet. Aplicaciones a laboratorios virtuales y remotos de robotica industrial.* Alicante: Universidad de Alicante.

Knasel, M. (1986). Mobile Robotics - State of the Art Review. *Elsevier Science Publishers B.V.*, 149 - 155.

León, O. (2012). *Robot industrial para la automatización del proceso de conformado de piezas en prensas troqueladoras.* Maracaibo: Universidad Rafael Belloso Chacín.

Loaiza Bernal, J. (2012). *Diseño y simulación de un prototipo de protesis de mano bioinspirada con cinco grados de libertad.* Bogotá: Universidad Nacional de Colombia.

López García, H., & González Librán, R. (1996). *´Programación de robots industriales .* Oviedo, Asturias : Universidad de Oviedo, 1996.

MathWorks. (30 de Diciembre de 2020). *Robótica*. Obtenido de https://www.mathworks.com/discovery/robotica.html

MathWorks. (30 de Diciembre de 2020). *Robotica y sistemas autónomos*. Obtenido de https://la.mathworks.com/solutions/robotics.html#:~:text=Los%20investigadores%20y%20los%20ingenieros,con%20los%20algoritmos%20que%20desarrolle.

MathWorsk. (30 de Diciembre de 2020). *Robotics System toolbox*. Obtenido de https://es.mathworks.com/products/robotics.html

Mendoza Pérez, M. A., Cruz Flores, R. G., Vilbalba Hernández, A. A., Calderón Rodríguez, J., & Arreola Patiño, E. (2017). APLICACIÓN DE REALIDAD AUMENTADA PARA LA ENSEÑANZA DE LA ROBÓTICA. *CITEC*, 14.

Meza, R. D., & Mamaní, G. A. (2018). *Implementación de la robotica educativa en la escuela: Un enfoque didactico para el diseño, construcción y programación de robots con alumnos de primaria.* Salta: XII Congreso nacional de tecnología en educación y educación en tecnología.

mister.D. (26 de Diciembre de 2020). *Blog MasterD*. Obtenido de https://www.masterd.es/blog/que-es-unity-3d-tutorial/

Morales Corral, C. (2020). *Enseñanza robotica mediante guiado manual con estimuklación de la fuerza de interacción.* Juarez: Universidad autonoma de ciudad de Juarez.

Moran, O., Künning, F., & Cuello, J. (2011). Implementción en el robot antropomorfo CXN-I del control cinemático de trayectorias rectilíneas. *22(1)*(93-100(2011)).

Navarro, N., Robles, S., & Paulsen, K. (2007). *Diseño y fabricación de un brazo robotico de conco grados de libertad articulado verticalmente modelado cinematico y dinamico.* Lima, Perú: CIBIM8.

Negrón, G. (2017). Realidad Virtual y Realidad Mixta: conceptos que intervienen en la era digital. *ResearchGate*, 6.

Ocampo, A., Olague, G., & Clemente, E. (2014). *Programación genetica para proponer un modelado que genera trayectorias para un robot articulado con base en su entorno.* Mexico: Congreso de instrumentación SOMI.

Ollero Baturone, A. (2001). *Robótica manipuladores y robots móviles.* Barcelona: Marcombo,.

Peña Tapia, E., Roldán, J. J., Garzón, M., Martín Barrio, A., & Barrientos, A. (2017). iNTERFAZ DE CONTROL PARA UN ROBOT MANIPULADOR MEDIANTE REALIDAD VIRTUAL. *uNIVERSIDADE CORUÑA*, 7.

Platzi. (Diciembre de 2020). *Platzi*. Obtenido de https://platzi.com/blog/que-es-unity-motor-videojuegos/

Prendes Espinosa, C. (2015). REALIDADA AUMENTADA Y EDUCACIÓN: ANÁLISIS DE EXPERIENCIAS PRÁCTICAS. *PIXEL-BIT Revista de Medios y Educación*, 18.

Quality devs. (2 de Junio de 2020). *Unity*. Obtenido de https://www.qualitydevs.com/2020/06/02/unity-que-es/

Ramires Arias, J., & Rubiano Fonseca, A. (2012). Modelamiento Matemático de la cinemática directa e inversa de un robot manipulador de tres grados de libertaad. *8*(15).

Ramíres Benavides, K. D. (s.f.). *Cinematica directa del robot.* Costa Rica: UCR – ECCI.

Ramírez Benavides, K. (s.f.). *Cinemática inversa del robot.* Escuela de ciencias de la computación e informatica.

Ramos Fuentes, G. A. (2000). Una herramienta de programación y comunicación para micro-robots industriales RV-M1 en ambientes gráficos. *4*(7).

Reyes Cortés, F. (2011). *Robótica control de robots manipuladores.* Mexico: Alfaomega.

Rico Riveros, L. f., Sanabria Sanabria, J. E., Bernal Tristancho , V. H., Quintero Urrea, J. D., Pinilla Santana, R. M., & Manosalva Fonseca, J. L. (2020). *Estrategia didactica basada en industria 4.0 para la integración de tecnologías de controladores lógicos programables y robots industriales.* Bogotá: Encuentro internacional de educación en ingeniería ACDFI.

Román Romero, D. M. (2014). *Simulador sinematico de un robot manioulador industrial.* Málaga: Escuela técnica superior de ingeniería informática.

Rossiter, J. (2016). *La robótica, los materiales inteligentes y su impacto futuro para la humanidad.* Obtenido de OpenMind BBVA: https://www.bbvaopenmind.com/articulos/la-robotica-los-materiales-inteligentes-y-su-impacto-futuro-para-la-humanidad/

Saltarén Pazmiño, R. J., Azorín Poveda, J. M., Almonacid Kroeger, M., & Sabater Navarro, J. M. (s.f.). *Prácticas de Robótica utilizando Matlab.* España: Universitas Miguel Hernandez.

Sierra Bueno, D. A., & Martinez Angel, R. (2002). Planeación de trayectoria para un robot en una celda de manufactua fisicomecánicas. *1*(22).

Slawiñski, E., Postigo, J., Mut, V., Carestía, D., & Castro, F. (2007). Estructura abierta de software para un robot industrial. *4*(3).

Solis Bautista, H. A., Mendoza Pérez, M. A., & Cruz Flores, R. G. (2020). DISEÑO DE UNA APLICAIÓN EN REALIDAD AUMENTADA PARA LA ENSEÑANAZA DE UN SEGUIDOR DE LÍNEA. *RILCO*, 14.

Somolinos Sánchez, J. A., & Salido Tercero, J. (2002). *Avances en robótica y visión por computadora.* Ciudad real: Ediciones de la universidad de Castilla-LaMancha.

Soto Cajiga, J. A., Vargas Soto, J. E., & Pedraza Ortega, J. C. (2005). *Generación de trayectorias por visión para un robot manipulador de 5 grados de libertad.* Coahuila: 4° Congreso nacional de mecatrónica.

Soto Cajiga, J. A., Vargas Soto, J. E., & Pedraza Ortega, J. C. (2006). Generación de trayectorias para un robot manipulador utilizando procesamiento de imágenes y splines.

Soto Duran, J., Peña, C., & Gualdron, O. (2013). *Diseño de un sistema robotizado de clasificación de brevas con fines academicos.* Pamplona: Ingenieria.

Suaréz , R. (1999). *Programación, planificación y control en robótica.* Barranquilla: I encuentro Nacional de robótica y manufactura flexiblke.

Suñer, J. L., Valero, F., Ródenas, J. J., & Besa, A. (2007). Comparación entre procedimientos de solución de la interpolación por funciones SPLINES para la planificación de trayectorias de robots industriales.

Tornil Sin, S., & Gámiz Caro, J. (2014). La robotica industrial en el ámbito de la automatización global: estado actual y tendencias.

Universidad Sntiago de Chile. (28 de Dicemebre de 2020). *UdeSantiagoVirtual*. Obtenido de http://www.udesantiagovirtual.cl/moodle2/mod/book/view.php?id=24819

Zaplana, I., Arnau Claret, J., & Basanez, L. (2018). Análisis cinemático de robots manipuladores redundantes: Aplicaión a los robots Kuka LWR 4+ y ABB Yumi. *15*(2).

ANEXOS

ANEXO 1. Estructura del código para cargar las piezas de la parte del robot

```
clear, close all, clc

%Longitud de los eslabones
L1a = 85;
L1b = 50.5;
L2 = 120.23;
L3 = 94.6;
L4 = 65;

%Ajuste de los eslabones
Raplim.eslabon0 = fun_stl2matlab('stl_eslabon0.STL', [0.64,0.68,0.72],0);
A0CAD = MTHtrasz(L1a);
Raplim.eslabon0 =
transforma_objeto_matlab_from_stl(Raplim.eslabon0,A0CAD);

Raplim.eslabon1 = fun_stl2matlab('stl_eslabon1.STL', [0.64,0.68,0.72],0);
A1CAD = MTHRoty(pi/2)*MTHRotx(-pi/2);
Raplim.eslabon1 =
transforma_objeto_matlab_from_stl(Raplim.eslabon1,A1CAD);

Raplim.eslabon2 = fun_stl2matlab('stl_eslabon2.STL',
[0.64,0.68,0.72],0);
A2CAD = MTHtrasx(-L2);
Raplim.eslabon2 =
transforma_objeto_matlab_from_stl(Raplim.eslabon2,A2CAD);

Raplim.eslabon3 = fun_stl2matlab('stl_eslabon3.STL', [0.64,0.68,0.72],0);
A3CAD = MTHtrasx(-L3);
Raplim.eslabon3 =
transforma_objeto_matlab_from_stl(Raplim.eslabon3,A3CAD);

Raplim.eslabon4 = fun_stl2matlab('stl_eslabon4.STL', [0.64,0.68,0.72],0);
A4CAD = MTHtrasx(-L4);
Raplim.eslabon4 =
transforma_objeto_matlab_from_stl(Raplim.eslabon4,A4CAD);
```

ANEXO 2. Registro de las muestras y resultados de los algoritmos de optimización.

En este anexo se dan a conocer los datos de las *50* muestras capturadas para cada una de las pruebas realizadas (línea recta, diagonal, triangulo, cuadrado y arco), a continuación, se asignan las siglas que pertenecen a cada uno de los atributos que aparecen en las tablas.

Donde:

Puntos capturados (P.C.): Se refiere a la cantidad de puntos capturados del movimiento del operador al representar la figura de prueba.

Error de captura (E.C.): Representa la similitud de los puntos que componen cada muestra y los obtenidos de cada algoritmo con respecto a los puntos del movimiento capturado.

Error de modelo (E.M.): Representa la similitud de los puntos que componen cada muestra y los obtenidos de cada algoritmo con respecto a la figura de prueba.

Puntos algoritmo (P.A.): Se refiere a la cantidad de puntos obtenidos de cada algoritmo de optimización.

Puntos control cinemático (P.C.C.): Se refiere a la cantidad de puntos generados automáticamente cada 5 milímetros en el espacio de la tarea para cada uno de los segmentos formados a partir de los puntos obtenidos de cada algoritmo de optimización. A estos puntos se les aplica el control cinemático para generar las trayectorias del robot.

Las tomas subrayadas en verde de cada una de las tablas del anexo 2 son aquellas que presentan menor error con respecto al modelo en el algoritmo filtro.

Anexo tabla 1.
Resultados de la figura de prueba línea.

Toma #	REGISTRO			FILTRO		VECTORIAL				ANALÍTICO			
	P. C.	E.C. (mm)	E.M. (mm)	E.C. (mm)	E.M. (mm)	P.A.	P.C.C.	E.C. (mm)	E.M. (mm)	P.A.	P.C.C.	E.C. (mm)	E.M. (mm)
1	36	0	59.27	5.45	58.42	3	25	15.90	47.49	3	26	11.95	48.21
2	45	0	16.80	2.38	16.31	3	23	6.92	17.28	3	23	8.90	15.18
3	37	0	23.86	3.42	23.01	3	23	7.80	16.53	3	25	6.70	19.87

4	34	0	22.88	2.83	22.85	2	23	10.04	14.47	3	23	4.62	21.52
5	45	0	30.64	4.31	30.29	3	24	11.00	19.60	3	24	11.06	19.64
6	36	0	12.97	2.64	12.63	4	21	5.10	10.43	3	22	4.09	11.26
7	38	0	11.43	3.28	9.43	6	22	4.99	7.23	3	21	6.43	5.70
8	35	0	35.11	5.30	33.46	3	21	18.02	18.14	3	25	11.91	25.03
9	30	0	28.01	3.37	27.70	2	21	13.51	15.83	3	24	6.08	24.46
10	30	0	19.24	3.22	18.54	5	21	8.24	15.04	3	24	7.59	13.35
11	28	0	24.44	3.17	24.08	2	22	13.78	11.86	3	23	7.09	19.00
12	29	0	38.62	7.10	36.73	3	23	18.11	21.10	3	24	14.66	26.88
13	33	0	25.47	3.71	25.45	4	24	6.49	23.21	3	26	7.44	22.50
14	32	0	31.41	5.13	30.37	2	22	18.78	13.18	3	25	8.64	25.40
15	43	0	29.26	3.22	29.27	3	24	11.00	20.84	3	28	6.55	24.89
16	39	0	24.80	2.96	24.86	4	23	12.33	13.46	3	25	8.57	16.83
17	35	0	23.78	4.28	23.53	3	22	10.41	16.95	3	25	8.08	18.05
18	30	0	16.60	2.52	15.84	3	21	9.79	8.52	3	23	7.50	11.17
19	37	0	30.17	2.89	29.89	2	24	11.14	17.99	3	25	7.35	24.15
20	40	0	36.45	6.18	35.14	3	25	12.99	27.71	3	29	10.75	30.75
21	39	0	28.39	4.70	27.36	4	21	14.11	17.09	3	24	9.39	20.92
22	38	0	19.64	3.10	19.56	3	21	9.65	10.68	3	24	4.60	17.92
23	36	0	31.28	3.31	31.05	4	21	10.64	23.98	3	25	6.53	27.92
24	33	0	28.44	4.88	27.90	2	22	15.78	13.92	3	23	8.12	21.95
25	38	0	25.00	2.71	24.78	3	23	10.79	14.95	3	23	5.98	21.45
26	36	0	36.00	2.93	34.72	3	24	9.37	28.76	3	24	6.67	29.37
27	31	0	23.13	2.20	22.48	2	23	9.91	14.29	3	23	5.06	18.88
28	35	0	25.08	2.74	24.76	4	21	5.96	19.96	3	24	5.40	21.22
29	42	0	21.14	2.55	21.34	3	22	8.58	14.58	3	23	6.70	16.90
30	36	0	29.39	4.56	28.34	3	21	17.08	13.70	3	24	8.17	23.83
31	30	0	16.10	2.34	15.55	3	22	8.44	9.07	3	23	4.35	14.07
32	42	0	36.10	4.05	35.18	2	24	15.73	23.96	3	28	6.53	33.58
33	43	0	38.12	4.98	37.72	2	23	18.12	21.19	3	26	11.24	30.78
34	40	0	52.38	6.34	51.37	2	22	27.18	26.65	3	27	15.19	41.28
35	31	0	35.86	5.79	34.11	2	22	17.30	16.63	3	25	10.50	28.73
36	37	0	17.21	3.83	16.81	5	21	10.82	8.82	3	23	7.11	11.91
37	30	0	58.11	5.77	57.12	3	24	14.18	47.40	3	25	11.31	48.03
38	39	0	15.37	2.14	15.25	4	23	5.15	13.02	3	23	4.55	13.04
39	42	0	51.98	5.87	50.91	3	26	15.58	37.89	3	26	15.61	37.90
40	38	0	55.43	4.77	54.50	3	27	8.42	51.32	3	27	8.36	51.31
41	43	0	29.80	6.52	28.45	4	23	11.70	22.71	3	26	10.30	24.91
42	36	0	29.06	4.71	27.24	3	23	9.09	25.09	3	25	7.74	26.34
43	30	0	26.11	3.60	25.26	2	21	14.65	12.24	3	22	6.69	21.05

44	28	0	23.21	3.15	22.83	3	22	6.83	18.58	3	23	5.56	20.34
45	27	0	24.72	5.30	24.59	3	21	10.81	17.48	3	23	7.99	20.88
46	32	0	62.42	12.03	59.87	3	26	25.48	41.56	3	31	20.89	46.43
47	38	0	25.23	2.60	25.00	2	22	9.76	16.01	3	24	3.98	23.44
48	33	0	31.21	3.51	31.03	2	22	12.11	22.28	3	24	4.89	28.62
49	33	0	49.55	6.75	48.07	3	22	19.66	30.06	3	26	12.48	40.55
50	37	0	34.75	3.36	34.30	2	21	12.97	23.24	3	24	5.40	32.46

Nota: El valor de "0" en la columna de Error Captura (E.C.) en el registro se da porque se está comparando con respecto a la misma muestra. En el algoritmo filtro no se agregó la columna P.A. y P.C.C. ya que los puntos que se obtuvieron y usaron para el cálculo del control cinemático son los mismos capturados de cada muestra en el registro.

La toma *7* del Anexo tabla 1 subrayada en verde es la muestra que menor error presenta de las *50* tomas, con ella se graficaron los resultados de la Tabla 6.

Anexo tabla 2.
Resultados de la figura de prueba diagonal.

Toma #	**REGISTRO**			**FILTRO**		**VECTORIAL**				**ANALÍTICO**			
	P. C.	E.C. (mm)	E.M. (mm)	E.C. (mm)	E.M. (mm)	P.A.	P.C.C.	E.C. (mm)	E.M. (mm)	P.A.	P.C.C.	E.C. (mm)	E.M. (mm)
1	50	0	40.15	2.81	39.25	3	33	5.32	35.79	3	33	5.32	35.79
2	51	0	58.17	6.78	56.24	3	40	11.85	51.33	3	40	11.79	51.33
3	47	0	52.93	6.86	50.79	3	40	13.00	48.09	3	40	13.00	48.09
4	57	0	50.25	1.68	49.45	2	33	7.71	42.76	3	33	4.32	46.79
5	42	0	41.94	3.92	40.99	3	33	9.77	31.26	3	35	6.21	37.94
6	43	0	62.31	3.57	61.64	2	33	14.46	50.38	3	35	9.44	58.13
7	44	0	43.25	2.43	43.23	3	33	6.01	38.92	3	33	5.07	39.06
8	54	0	60.02	3.06	59.27	2	33	9.43	51.66	3	34	5.07	57.07
9	30	0	34.66	5.36	32.52	3	30	14.16	22.51	3	33	10.66	28.76
10	36	0	52.89	4.70	51.76	2	33	16.13	40.98	3	35	7.59	48.40
11	32	0	39.87	6.06	37.67	3	31	12.22	27.18	3	34	8.74	36.09
12	40	0	52.27	2.87	50.91	2	33	10.93	40.21	3	33	7.54	43.51
13	41	0	38.05	1.94	37.57	2	33	4.84	33.72	3	32	3.60	35.42
14	46	0	42.21	2.94	41.27	3	35	7.53	38.94	3	35	7.53	38.94
15	39	0	48.62	6.31	46.53	3	39	12.25	42.03	3	39	12.25	42.03
16	32	0	39.48	3.61	37.95	2	32	13.37	26.74	3	32	7.92	32.08
17	41	0	41.19	5.65	39.02	3	34	15.00	27.40	3	38	10.14	36.52
18	49	0	34.42	3.73	33.13	4	33	7.04	31.94	3	33	8.25	31.61
19	44	0	41.16	2.13	40.57	2	32	6.89	35.40	3	32	3.75	38.58
20	50	0	48.13	3.32	47.41	3	34	10.18	38.87	3	35	8.84	42.08
21	40	0	53.03	2.89	51.88	2	33	7.70	44.13	3	34	5.05	48.25

22	42	0	45.16	2.40	44.50	2	33	12.68	32.63	3	33	8.04	38.73
23	40	0	30.72	1.75	30.21	2	32	7.08	26.40	3	33	3.40	28.61
24	42	0	70.41	8.24	68.21	3	41	15.96	60.39	3	41	15.96	60.39
25	39	0	48.93	7.07	46.69	4	35	12.98	38.64	3	39	12.62	41.93
26	44	0	53.11	3.33	52.16	2	34	11.10	45.52	3	35	5.04	50.24
27	47	0	47.60	5.76	45.63	4	36	11.43	41.26	3	37	10.99	43.25
28	38	0	44.51	2.43	43.76	2	31	12.71	33.54	3	31	8.90	36.64
29	39	0	22.66	3.43	21.04	3	32	8.68	18.63	3	32	9.47	15.65
30	43	0	60.43	4.75	59.46	3	34	10.80	55.65	3	35	9.64	56.39
31	37	0	47.08	2.55	46.25	2	33	10.36	38.15	3	32	5.68	43.47
32	50	0	35.83	1.79	35.39	2	34	6.59	32.14	3	34	5.05	33.11
33	47	0	35.30	3.45	34.05	3	35	6.93	29.92	3	35	5.61	33.14
34	47	0	59.22	3.39	58.42	2	33	16.20	42.84	3	34	8.98	49.11
35	47	0	56.99	2.47	56.51	2	34	14.85	43.06	3	33	7.18	51.73
36	58	0	53.52	3.17	52.59	3	35	9.22	45.20	3	35	9.22	45.20
37	56	0	45.58	2.61	45.04	2	32	12.54	30.69	3	34	5.40	40.40
38	49	0	50.59	2.92	49.97	2	33	13.50	38.47	3	34	6.27	44.46
39	45	0	48.75	5.98	46.85	3	34	19.50	28.30	3	38	13.40	37.47
40	55	0	55.55	3.03	54.14	3	35	7.58	52.40	3	34	5.97	50.51
41	42	0	37.40	3.63	35.82	4	35	6.21	37.31	3	35	6.06	36.79
42	47	0	38.67	1.71	38.05	2	33	7.39	35.25	3	34	7.44	36.32
43	46	0	28.90	2.33	28.15	3	33	4.78	26.71	3	33	4.78	26.71
44	47	0	34.07	3.65	32.61	3	34	9.61	26.83	3	35	7.77	30.35
45	42	0	51.04	3.39	50.16	2	34	14.66	39.40	3	35	7.27	44.83
46	46	0	27.74	1.69	27.38	2	33	7.33	25.50	3	32	3.83	27.53
47	41	0	38.13	2.25	37.47	2	33	4.81	33.79	3	33	3.84	34.88
48	52	0	35.22	4.22	34.55	3	30	13.56	23.30	3	34	8.47	29.72
49	57	0	37.22	4.73	35.42	3	33	16.21	20.28	3	36	10.98	31.59
50	48	0	29.04	2.20	28.64	2	31	8.24	22.97	3	32	5.19	28.65

Nota: El valor de "0" en la columna de Error Captura (E.C.) en el registro se da porque se está comparando con respecto a la misma muestra. En el algoritmo filtro no se agregó la columna P.A. y P.C.C. ya que los puntos que se obtuvieron y usaron para el cálculo del control cinemático son los mismos capturados de cada muestra en el registro.

La toma *29* del Anexo tabla 2 subrayada en verde es la muestra que menor error presenta de las *50* tomas, con ella se graficaron los resultados de la Tabla 7.

Anexo tabla 3.
Resultados de la figura de prueba triángulo.

	REGISTRO	**FILTRO**	**VECTORIAL**	**ANALÍTICO**

Toma #	P. C.	E.C. (mm)	E.M. (mm)	E.C. (mm)	E.M. (mm)	P.A.	P.C.C.	E.C. (mm)	E.M. (mm)	P.A.	P.C.C.	E.C. (mm)	E.M. (mm)
1	142	0	32.23	2.04	31.88	8	72	6.35	28.81	4	73	6.26	28.44
2	167	0	20.19	1.82	20.15	14	72	4.74	19.40	4	71	6.60	18.30
3	112	0	31.73	2.40	31.12	6	72	9.74	26.86	4	70	8.34	28.56
4	108	0	16.06	2.17	15.78	8	77	5.66	14.75	4	69	6.00	15.19
5	113	0	40.76	2.78	40.41	8	68	9.93	36.62	4	69	11.92	34.40
6	99	0	54.25	4.10	53.91	6	73	11.68	50.40	4	73	12.43	49.42
7	100	0	50.50	2.51	49.70	4	66	9.93	45.63	4	66	9.60	43.61
8	130	0	25.68	1.77	25.61	12	68	5.90	24.77	4	66	8.59	26.20
9	104	0	43.84	2.09	43.82	4	75	8.29	40.14	4	73	6.81	40.00
10	97	0	39.06	1.98	38.76	5	73	6.76	36.25	4	72	7.00	36.41
11	87	0	43.67	2.48	43.65	4	71	8.97	38.30	4	71	7.95	40.27
12	128	0	39.41	1.80	39.22	4	74	10.50	35.88	4	70	9.14	36.85
13	107	0	57.89	2.60	57.59	5	74	12.07	50.89	4	75	12.43	51.66
14	135	0	36.34	2.23	36.29	5	78	9.36	31.47	4	77	9.95	33.29
15	101	0	53.53	2.34	52.99	4	68	11.03	48.36	4	67	11.11	46.76
16	89	0	63.92	2.40	63.96	4	70	6.32	61.03	4	70	5.68	60.57
17	85	0	55.15	3.23	55.10	4	65	12.61	51.49	4	71	9.26	49.78
18	119	0	60.63	2.62	60.40	5	70	10.49	54.00	4	70	10.76	54.00
19	94	0	47.68	3.48	47.45	5	71	10.57	44.24	4	72	10.38	44.34
20	96	0	50.22	2.63	50.26	4	72	10.94	43.98	4	72	10.94	43.98
21	105	0	33.02	2.46	32.87	5	74	10.36	30.75	4	74	10.79	28.50
22	87	0	53.09	2.70	53.06	4	71	9.64	46.57	4	72	9.64	46.45
23	101	0	27.01	2.43	26.87	9	72	7.39	24.90	4	72	9.03	24.92
24	87	0	43.05	2.36	42.93	4	63	10.20	42.50	4	67	6.79	39.95
25	96	0	48.17	3.59	47.94	5	71	8.15	46.16	4	73	11.10	43.80
26	100	0	64.13	2.77	64.04	4	69	13.71	55.83	4	66	12.89	55.52
27	128	0	32.27	1.82	32.06	11	71	6.69	30.20	4	71	7.47	32.05
28	122	0	40.22	2.09	40.06	4	71	8.55	36.45	4	71	8.38	34.84
29	118	0	45.04	2.17	44.77	6	72	9.55	40.22	4	74	9.90	40.26
30	136	0	37.12	2.38	36.92	6	74	9.48	33.68	4	72	10.84	33.67
31	101	0	39.11	2.55	38.61	6	70	6.16	38.93	4	70	8.01	35.06
32	101	0	51.29	2.31	51.25	5	73	10.26	46.14	4	69	9.82	45.96
33	109	0	37.58	2.02	37.55	4	75	7.68	35.17	4	75	7.56	35.14
34	100	0	34.53	2.19	34.33	5	68	8.23	31.91	4	69	7.79	32.62
35	82	0	54.24	3.87	53.89	6	69	11.33	48.67	4	69	11.11	49.70
36	133	0	38.70	2.07	38.75	7	75	6.73	37.82	4	75	8.36	38.60
37	92	0	34.17	2.41	34.01	4	70	6.28	32.96	4	70	5.07	32.70
38	115	0	32.32	2.31	32.06	8	73	6.45	29.83	4	71	9.54	28.51
39	107	0	46.93	2.11	46.82	8	66	7.52	43.60	4	67	7.56	43.09

40	100	0	35.37	1.96	34.92	4	66	8.39	33.86	4	67	8.30	32.54
41	95	0	43.61	1.94	43.82	4	71	5.48	40.72	4	71	5.26	40.70
42	136	0	27.24	1.74	27.10	11	81	7.73	26.52	4	72	8.61	28.27
43	100	0	43.56	2.44	42.91	4	67	8.30	41.51	4	67	8.04	41.16
44	123	0	41.33	3.29	41.30	10	76	11.08	33.89	4	73	12.05	35.84
45	88	0	50.37	2.40	50.44	4	68	10.19	47.79	4	69	8.66	46.91
46	93	0	54.53	2.97	54.26	5	62	13.23	50.53	4	67	9.99	48.75
47	93	0	52.11	2.11	51.75	4	68	6.20	46.74	4	66	5.91	46.63
48	104	0	47.11	2.45	46.75	4	65	10.57	43.40	4	64	10.24	44.04
49	120	0	36.31	2.04	36.12	6	73	6.09	34.70	4	72	5.56	35.23
50	104	0	31.28	2.10	30.78	7	79	8.93	29.02	4	77	7.75	30.38

Nota: El valor de "0" en la columna de Error Captura (E.C.) en el registro se da porque se está comparando con respecto a la misma muestra. En el algoritmo filtro no se agregó la columna P.A. y P.C.C. ya que los puntos que se obtuvieron y usaron para el cálculo del control cinemático son los mismos capturados de cada muestra en el registro.

La toma *4* de la Anexo tabla 3 subrayada en verde es la muestra que menor error presenta de las *50* tomas, con ella se graficaron los resultados de la Tabla 8.

Anexo tabla 4.
Resultados de la figura de prueba cuadrado.

Toma #	**REGISTRO**			**FILTRO**		**VECTORIAL**				**ANALÍTICO**			
	P. C.	E.C. (mm)	E.M. (mm)	E.C. (mm)	E.M. (mm)	P.A.	P.C.C.	E.C. (mm)	E.M. (mm)	P.A.	P.C.C.	E.C. (mm)	E.M. (mm)
1	138	0	61.11	2.71	61.27	8	82	8.95	58.00	5	83	10.20	59.59
2	123	0	43.57	3.00	43.51	8	86	8.69	40.23	5	86	11.29	39.17
3	139	0	42.42	2.24	42.38	8	80	8.30	40.31	5	85	9.14	39.11
4	126	0	35.87	2.73	35.40	6	81	7.43	31.99	5	81	6.13	33.06
5	145	0	47.52	2.56	47.30	7	78	11.75	40.33	5	80	9.81	42.86
6	149	0	42.63	2.42	42.30	7	75	9.68	39.42	5	77	10.13	39.66
7	161	0	37.97	2.37	37.91	9	79	6.33	33.97	5	80	8.96	35.82
8	130	0	46.30	2.69	46.11	11	76	8.71	45.09	5	77	8.15	45.53
9	132	0	39.80	2.01	39.82	8	78	10.10	36.56	5	79	8.68	38.78
10	107	0	43.85	2.28	43.47	6	73	8.45	41.67	5	79	7.07	42.22
11	113	0	40.52	3.03	40.69	7	75	9.32	40.11	5	81	7.60	39.25
12	163	0	47.92	1.79	47.82	10	81	8.86	42.08	5	82	8.10	45.32
13	128	0	46.70	2.82	46.37	6	80	9.86	41.08	5	79	9.77	42.31
14	126	0	50.34	2.73	50.46	8	76	8.54	48.34	5	77	8.14	50.16
15	117	0	47.42	2.40	47.19	6	78	12.21	38.43	5	81	9.41	42.90
16	129	0	46.20	3.02	46.05	7	71	11.33	40.29	5	75	9.92	40.90
17	123	0	46.88	2.45	46.79	8	76	13.21	38.29	5	80	10.15	45.86

18	129	0	43.63	2.25	43.10	8	76	8.62	36.84	5	79	9.06	39.49
19	119	0	27.91	2.06	27.62	7	75	7.77	29.33	5	79	7.11	27.47
20	126	0	37.10	2.25	37.01	6	76	10.22	33.81	5	77	7.61	34.52
21	115	0	39.09	2.65	38.77	5	81	13.99	29.04	5	83	9.77	33.62
22	137	0	36.67	1.95	36.42	7	77	6.60	36.22	5	81	6.30	36.87
23	125	0	44.21	2.70	44.00	5	83	9.84	39.17	5	84	8.51	39.59
24	132	0	38.84	2.08	38.77	5	79	7.70	35.59	5	83	6.35	35.56
25	125	0	36.80	2.32	36.54	6	74	8.84	31.72	5	78	7.72	35.95
26	130	0	29.91	2.06	29.78	10	80	6.73	29.91	5	80	6.62	29.11
27	125	0	50.71	2.52	50.78	9	74	8.68	48.53	5	77	8.94	46.49
28	122	0	38.77	2.04	38.59	7	73	9.23	36.67	5	76	6.07	36.37
29	135	0	36.29	2.28	36.44	7	76	11.27	34.28	5	80	8.32	34.00
30	109	0	36.84	2.54	36.43	7	74	9.15	33.31	5	79	7.60	34.02
31	118	0	37.81	2.00	37.83	7	75	8.29	33.43	5	78	6.76	35.78
32	133	0	33.38	2.00	33.00	7	69	15.71	31.08	5	76	8.10	31.34
33	138	0	33.09	2.20	32.90	6	79	8.22	30.74	5	79	7.36	33.05
34	139	0	33.84	2.38	33.49	8	70	9.03	31.21	5	79	7.38	34.46
35	103	0	37.18	1.97	37.20	5	80	8.06	32.70	5	80	6.42	34.95
36	132	0	28.01	1.80	27.96	8	88	6.54	26.01	5	84	6.46	28.78
37	103	0	32.12	2.27	31.88	5	75	9.54	29.26	5	79	6.56	30.84
38	115	0	36.09	2.24	35.62	8	84	7.51	32.56	5	82	7.86	33.55
39	124	0	35.59	2.12	35.70	7	76	8.51	33.88	5	80	7.37	36.02
40	132	0	47.91	3.01	47.65	8	81	10.74	42.60	5	78	10.76	43.09
41	131	0	33.31	2.24	33.09	6	81	9.06	32.12	5	82	7.08	31.59
42	117	0	40.83	2.11	40.53	5	79	7.89	35.59	5	77	7.44	36.97
43	98	0	35.76	2.63	35.47	5	73	7.99	34.83	5	78	6.45	34.72
44	106	0	45.07	3.13	44.76	6	72	11.66	42.75	5	75	8.06	41.63
45	124	0	43.36	2.39	42.91	6	80	8.56	40.21	5	79	8.81	40.00
46	98	0	50.99	2.89	50.93	5	78	12.75	47.73	5	78	12.27	47.16
47	98	0	43.90	2.37	43.85	5	77	11.18	37.92	5	82	8.51	38.99
48	97	0	43.91	2.44	43.32	6	82	9.81	36.40	5	81	10.15	38.13
49	88	0	38.58	3.67	38.22	7	75	10.47	36.92	5	77	9.17	37.54
50	104	0	48.69	2.34	48.29	5	82	8.47	43.21	5	79	7.24	44.15

Nota: El valor de “0” en la columna de Error Captura (E.C.) en el registro se da porque se está comparando con respecto a la misma muestra. En el algoritmo filtro no se agregó la columna P.A. y P.C.C. ya que los puntos que se obtuvieron y usaron para el cálculo del control cinemático son los mismos capturados de cada muestra en el registro.

La toma *19* de la Anexo tabla 4 subrayada en verde es la muestra que menor error presenta de las *50* tomas, con ella se graficaron los resultados de la Tabla 9.

Anexo tabla 5.
Resultados de la figura de prueba arco.

Toma #	REGISTRO			FILTRO		VECTORIAL				ANALÍTICO			
	P. C.	E.C. (mm)	E.M. (mm)	E.C. (mm)	E.M. (mm)	P.A.	P.C.C.	E.C. (mm)	E.M. (mm)	P.A.	P.C.C.	E.C. (mm)	E.M. (mm)
1	105	0	14.17	1.87	14.15	12	31	15.19	24.37	4	38	6.86	18.68
2	70	0	17.43	2.62	17.19	8	36	8.97	21.72	4	40	6.70	18.84
3	68	0	15.05	1.98	14.94	9	40	6.83	19.67	4	40	5.48	17.95
4	98	0	12.11	2.58	11.86	12	39	6.15	15.50	4	44	5.89	14.22
5	90	0	8.13	2.30	8.19	13	45	10.54	12.23	4	46	6.91	7.23
6	70	0	10.52	2.33	10.58	9	38	9.09	16.08	4	43	6.13	12.42
7	99	0	8.48	3.19	8.31	15	38	9.26	16.00	4	41	7.21	12.55
8	70	0	9.98	3.05	10.13	10	39	7.64	14.86	4	42	5.91	12.33
9	69	0	13.36	2.85	13.12	10	35	5.07	13.91	4	39	5.99	14.77
10	76	0	9.74	2.22	9.62	9	40	7.37	13.59	4	46	5.61	9.59
11	68	0	19.03	2.93	19.13	9	36	10.79	23.75	4	37	5.41	20.42
12	74	0	12.44	2.37	12.00	10	40	7.77	18.03	4	43	5.82	15.14
13	51	0	12.79	3.81	12.33	8	39	12.32	18.32	4	43	8.28	17.04
14	75	0	13.94	3.46	13.35	11	37	11.87	21.42	4	41	6.35	15.78
15	75	0	9.93	2.27	9.82	8	41	7.73	14.79	4	47	6.87	12.82
16	65	0	14.72	2.47	14.40	7	46	8.68	18.29	4	49	5.56	13.73
17	83	0	8.50	2.58	8.17	12	46	10.71	13.24	4	50	8.28	8.22
18	81	0	12.61	2.01	13.10	10	42	5.95	17.87	4	41	8.30	16.46
19	83	0	13.50	2.48	13.60	10	37	8.59	19.97	4	40	7.27	18.07
20	78	0	11.36	2.66	11.23	10	41	9.90	19.55	4	45	6.25	16.57
21	80	0	12.11	4.06	12.07	9	37	10.40	18.59	4	44	8.20	15.97
22	60	0	11.68	3.04	11.77	7	35	14.45	22.56	4	44	5.99	16.16
23	67	0	13.62	2.06	13.39	9	39	9.74	20.62	4	44	6.16	16.59
24	70	0	9.04	2.23	8.98	9	48	7.44	8.40	4	53	7.91	9.50
25	89	0	9.49	1.85	9.50	11	41	8.05	14.57	4	40	6.15	11.38
26	95	0	10.42	1.91	10.46	14	37	8.30	13.37	4	47	5.64	10.76
27	86	0	11.07	2.33	11.11	14	41	7.15	16.69	4	45	7.69	15.09
28	86	0	10.63	1.95	10.64	13	39	13.08	19.49	4	45	6.30	13.60
29	57	0	17.86	2.56	17.85	7	35	17.34	23.80	4	42	6.07	20.40
30	78	0	15.25	3.49	15.09	10	41	8.03	17.74	4	39	9.30	20.33
31	76	0	11.50	2.38	11.45	8	45	5.48	16.04	4	49	5.06	13.64
32	90	0	11.09	2.30	10.94	9	40	12.56	18.38	4	46	7.25	13.24
33	72	0	14.93	2.41	15.04	10	36	10.37	21.50	4	43	5.29	18.89
34	92	0	12.51	2.24	12.49	14	47	4.72	13.69	4	48	6.80	13.94
35	86	0	13.11	2.17	12.95	12	42	13.44	21.64	4	48	5.88	15.24
36	98	0	11.39	2.29	11.28	12	42	5.51	13.15	4	44	6.36	11.97

37	100	0	12.32	2.57	12.48	19	49	4.69	14.20	4	50	6.91	12.66
38	97	0	8.10	3.01	8.10	13	36	9.74	11.79	4	45	6.64	8.82
39	90	0	11.46	2.23	11.09	15	42	7.13	15.29	4	49	5.55	12.63
40	89	0	11.45	2.22	11.25	8	37	9.68	19.26	4	44	4.96	14.79
41	93	0	9.32	3.09	8.89	11	39	9.27	15.46	4	46	6.10	10.67
42	109	0	13.12	1.83	12.89	14	44	4.23	14.07	4	46	5.82	16.51
43	87	0	15.46	2.34	15.14	11	44	8.85	20.12	4	46	5.19	16.78
44	109	0	12.69	2.18	12.59	12	38	5.94	13.59	4	39	5.93	12.61
45	80	0	12.22	2.83	12.33	12	37	9.57	18.24	4	38	5.92	15.17
46	67	0	12.77	2.53	12.64	10	38	5.90	15.29	4	45	5.37	14.57
47	82	0	11.18	2.14	11.21	12	36	10.29	19.47	4	45	5.73	15.01
48	83	0	12.57	1.75	12.57	12	39	15.76	21.83	4	43	5.16	17.24
49	76	0	14.37	2.25	14.23	9	45	4.40	16.81	4	47	5.18	15.95
50	98	0	14.80	1.99	14.50	12	40	3.76	16.68	4	45	4.46	16.37

Nota: El valor de "0" en la columna de Error Captura (E.C.) en el registro se da porque se está comparando con respecto a la misma muestra. En el algoritmo filtro no se agregó la columna P.A. y P.C.C. ya que los puntos que se obtuvieron y usaron para el cálculo del control cinemático son los mismos capturados de cada muestra en el registro.

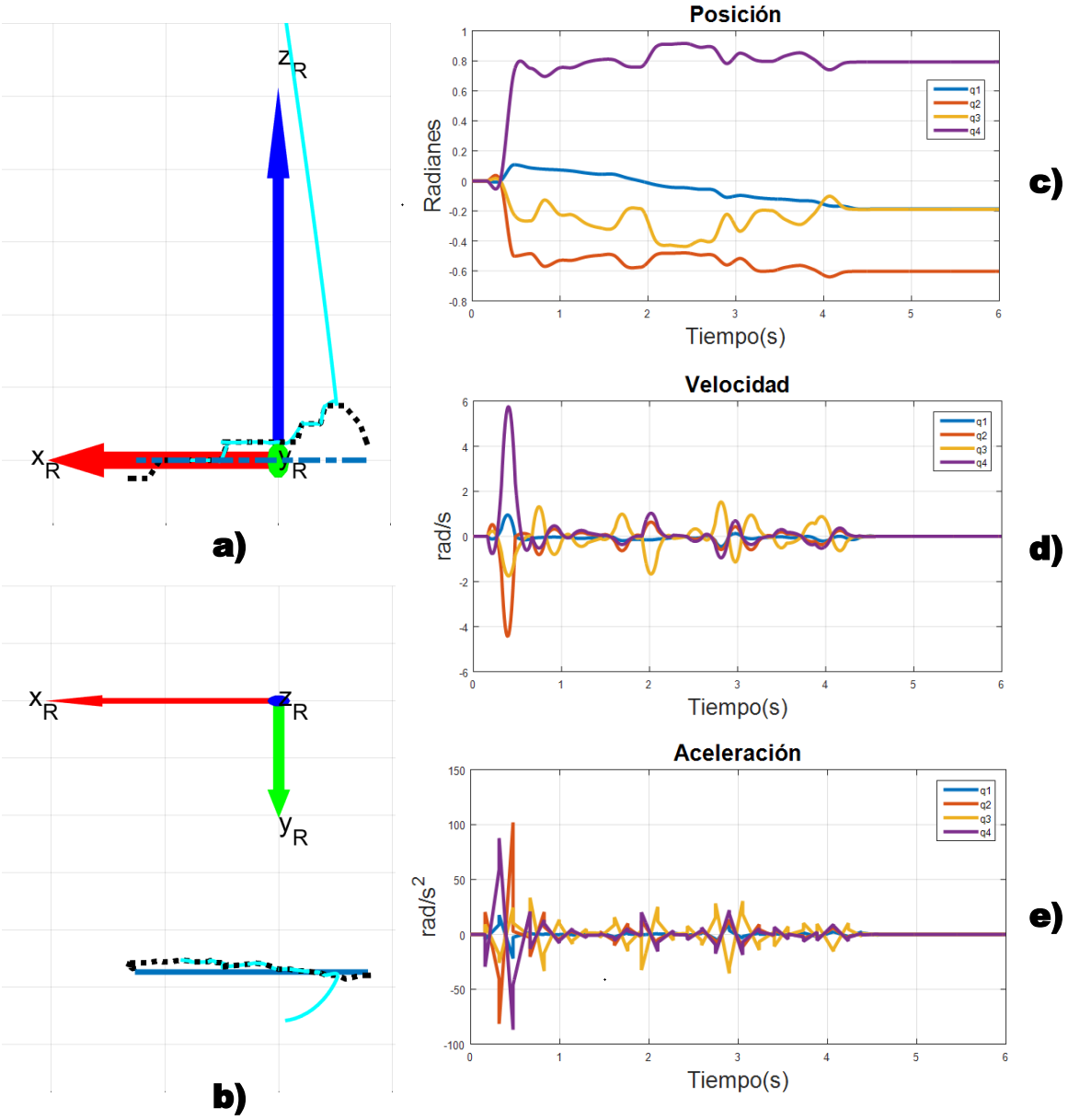
La toma *38* de la Anexo tabla 5 subrayada en verde es la muestra que menor error presenta de las *50* tomas, con ella se graficaron los resultados de la Tabla 10.

ANEXO 3. Resultados de las trayectorias obtenidas en cada prueba.

En este anexo se presentan los resultados correspondientes a las trayectorias (interpolador spline cúbico) en forma de graficas de cada una de las formas de prueba para el algoritmo de optimización analítico, vectorial y filtro. Los resultados corresponden a la muestra seleccionada de cada prueba vista en el capítulo de resultados.

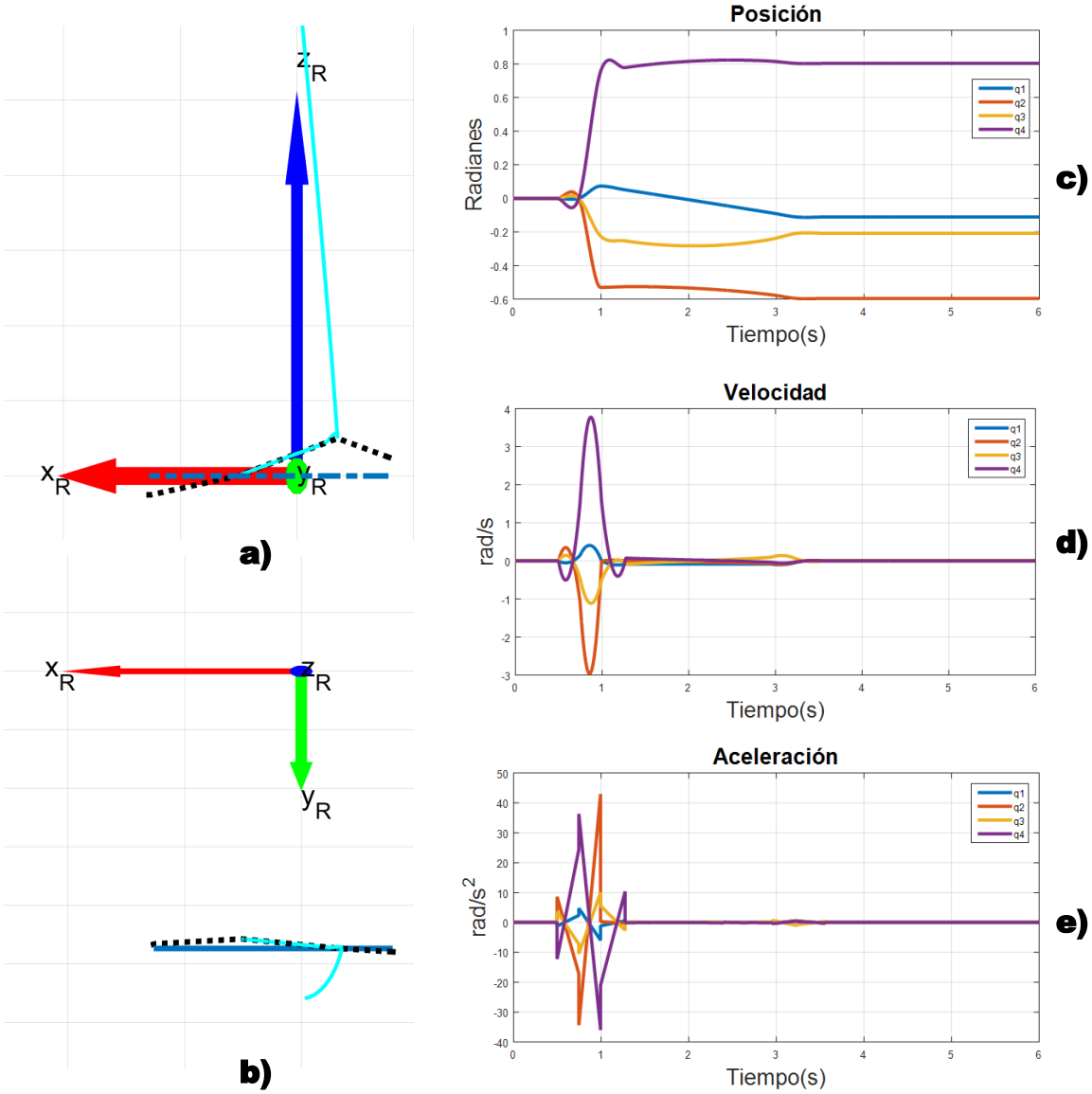
Las figuras presentadas en el nexo 3 se dividen en: a) Vista lateral de la muestra, donde la línea de color azul claro representa el movimiento del robot, la línea negra punteada representa el resultado del algoritmo aplicado y la línea azul oscuro representa la figura de prueba (línea recta, línea diagonal, triángulo, cuadrado y arco). b) Vista de techo de la muestra, donde la línea de color azul claro representa el movimiento del robot, la línea negra punteada representa el resultado del algoritmo aplicado y la línea azul oscuro representa la figura de prueba (línea recta, línea diagonal, triángulo, cuadrado y arco). c) Grafica de la posición articular Vs tiempo de cada una de las cuatro articulaciones del robot al ejecutar la trayectoria (línea azul claro) obtenida del interpolador. d) Grafica de la velocidad articular Vs tiempo de cada una de las cuatro articulaciones del robot al ejecutar la trayectoria (línea azul claro) obtenida del interpolador. e) Grafica de la aceleración articular Vs tiempo de cada una de las cuatro articulaciones del robot al ejecutar la trayectoria (línea azul claro) obtenida del interpolador. Las indicaciones anteriores se aplican para todas las figuras del anexo 3.

Forma: Línea
Algoritmo: Filtro



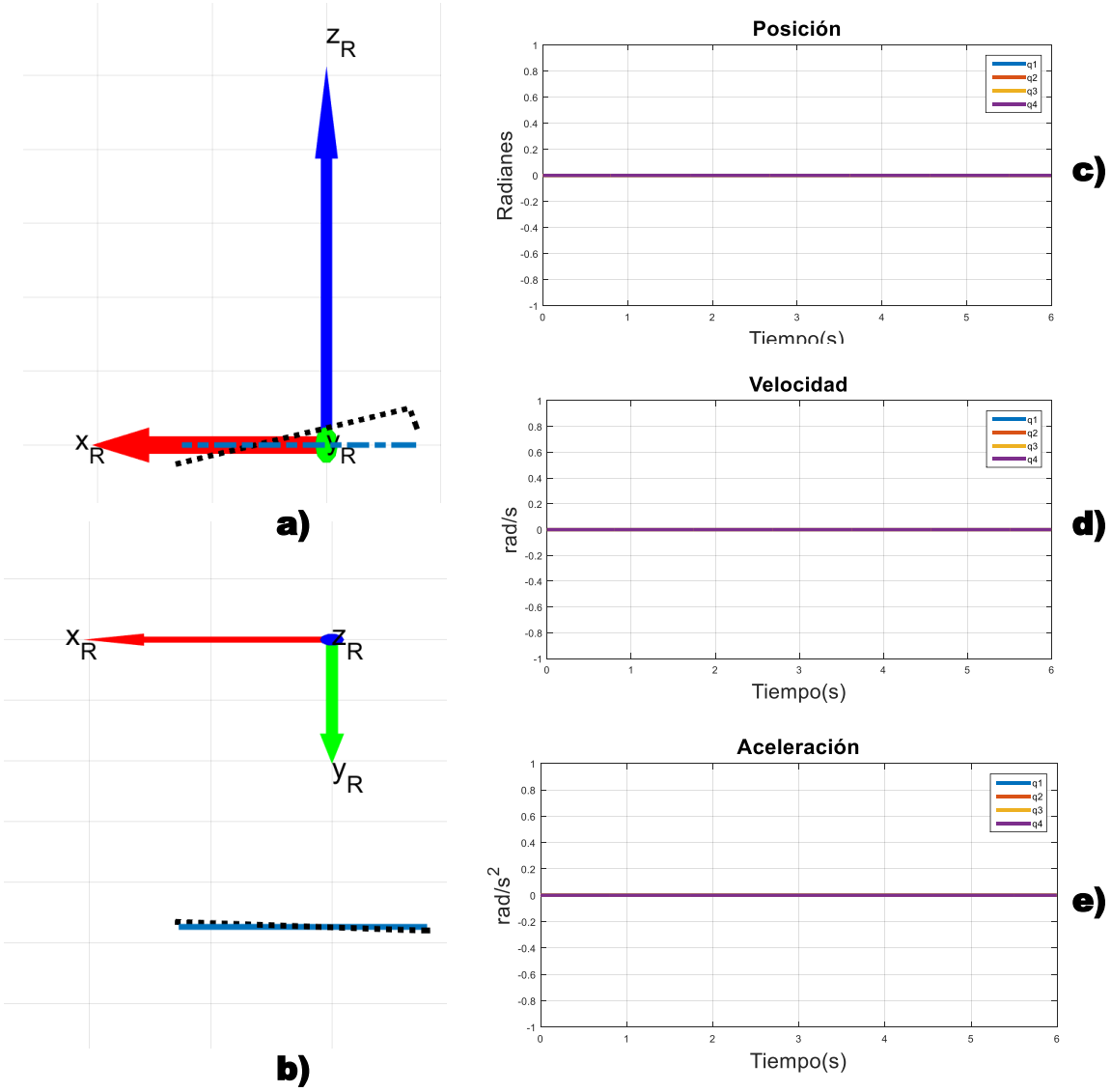
Anexo figura 1. Resultado de la trayectoria al aplicar el algoritmo filtro para la línea recta.

Forma: Línea
Algoritmo: Vectorial



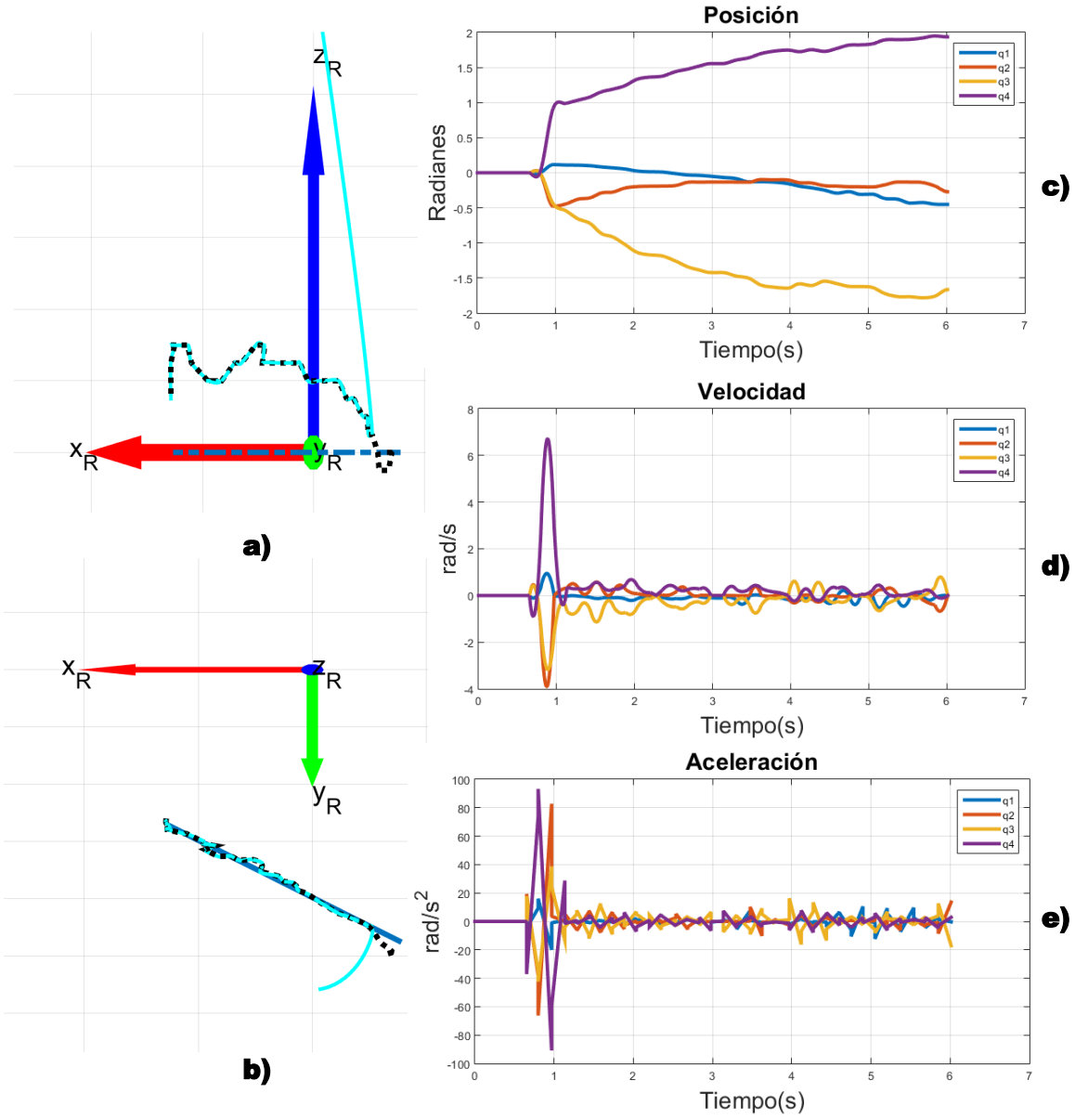
Anexo figura 2. Resultado de la trayectoria al aplicar el algoritmo de optimización vectorial para la línea recta.

Forma: Línea
Algoritmo: Analítico



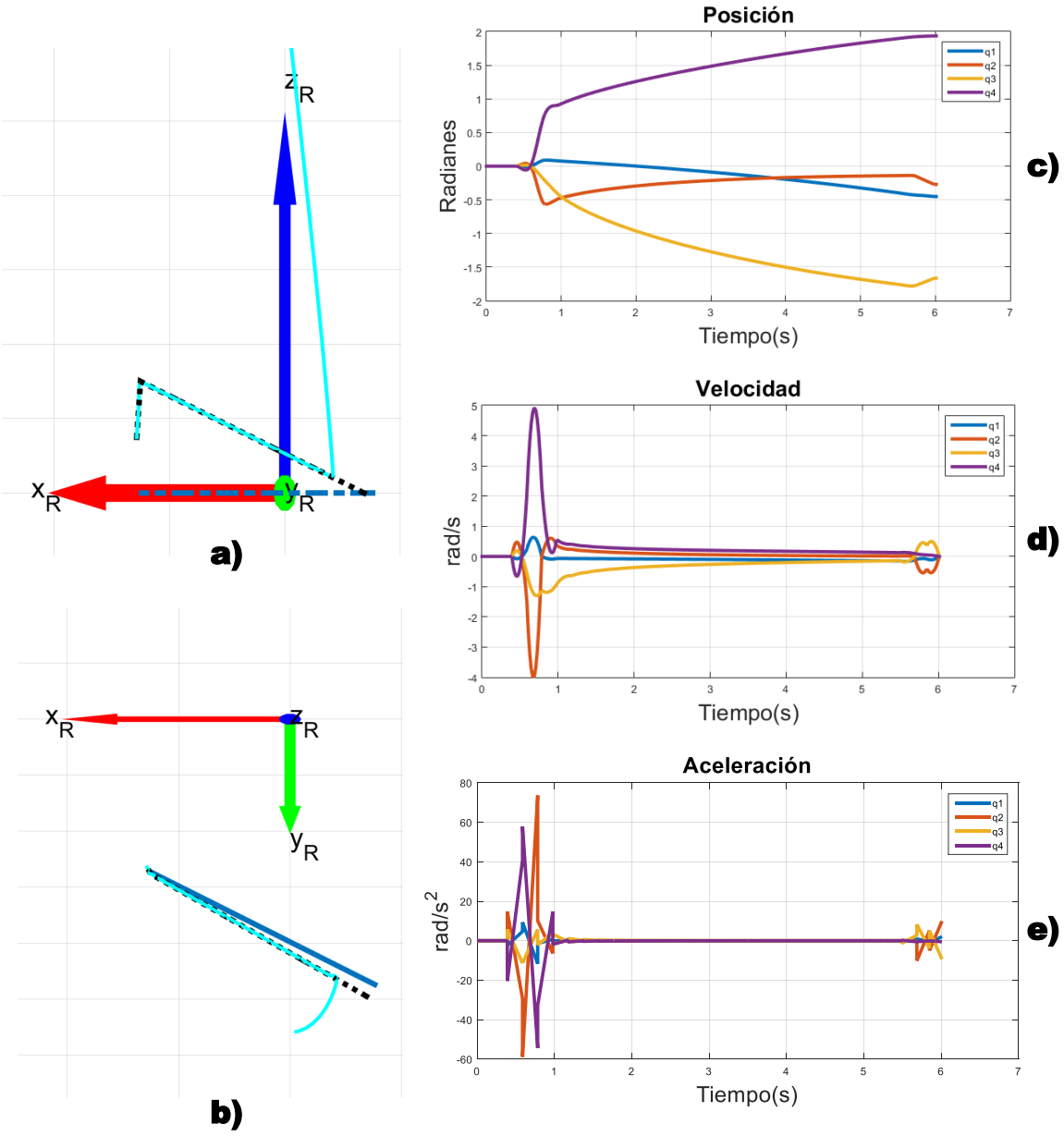
Anexo figura 3. Resultado de la trayectoria al aplicar el algoritmo de optimización analítico para la línea recta. Nota: Al aplicar el filtro analítico no se encuentran puntos alcanzables por el robot.

Forma: Diagonal
Algoritmo: Filtro



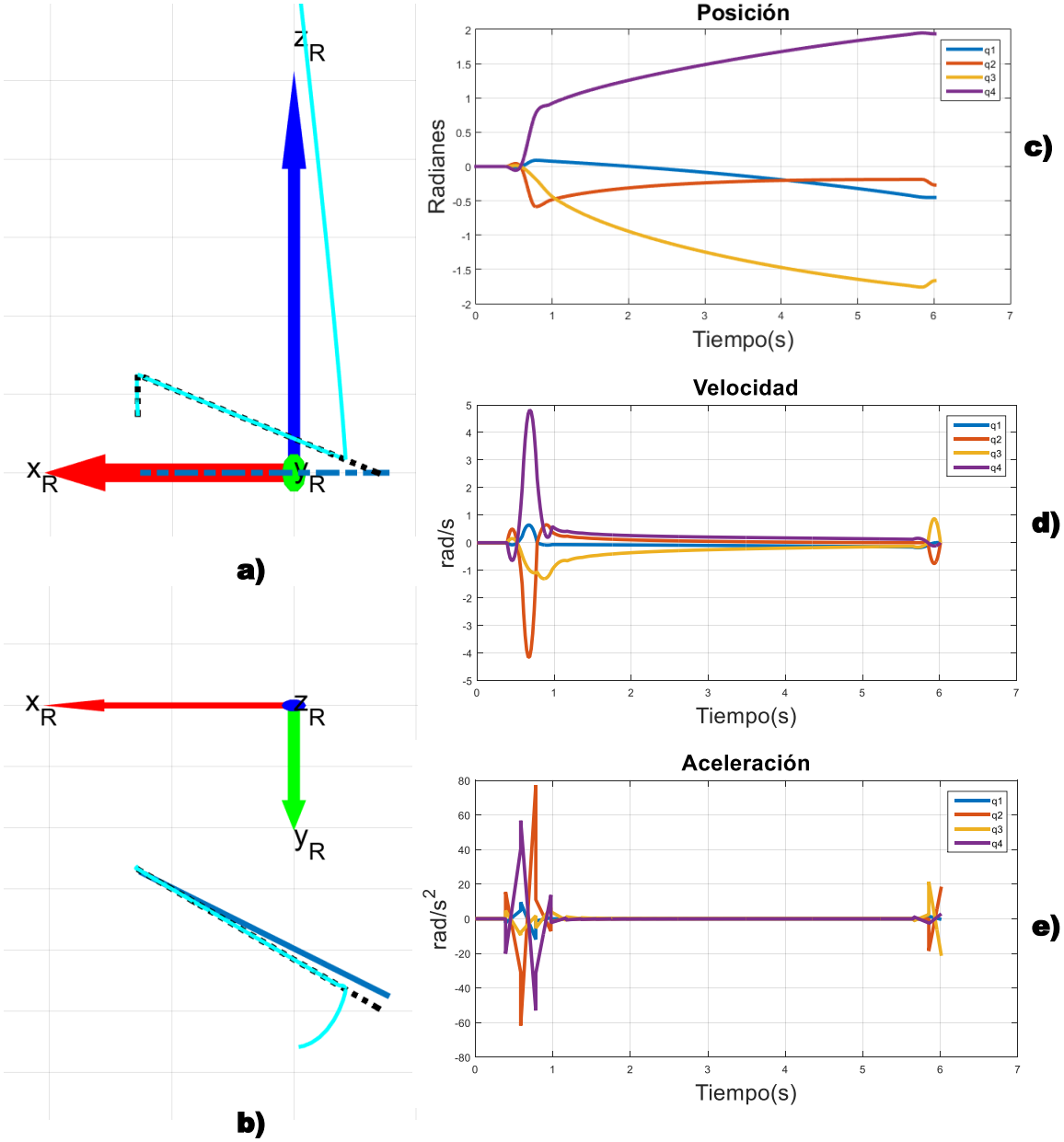
Anexo figura 4. Resultado de la trayectoria al aplicar el algoritmo filtro para la diagonal.

Forma: Diagonal
Algoritmo: Vectorial



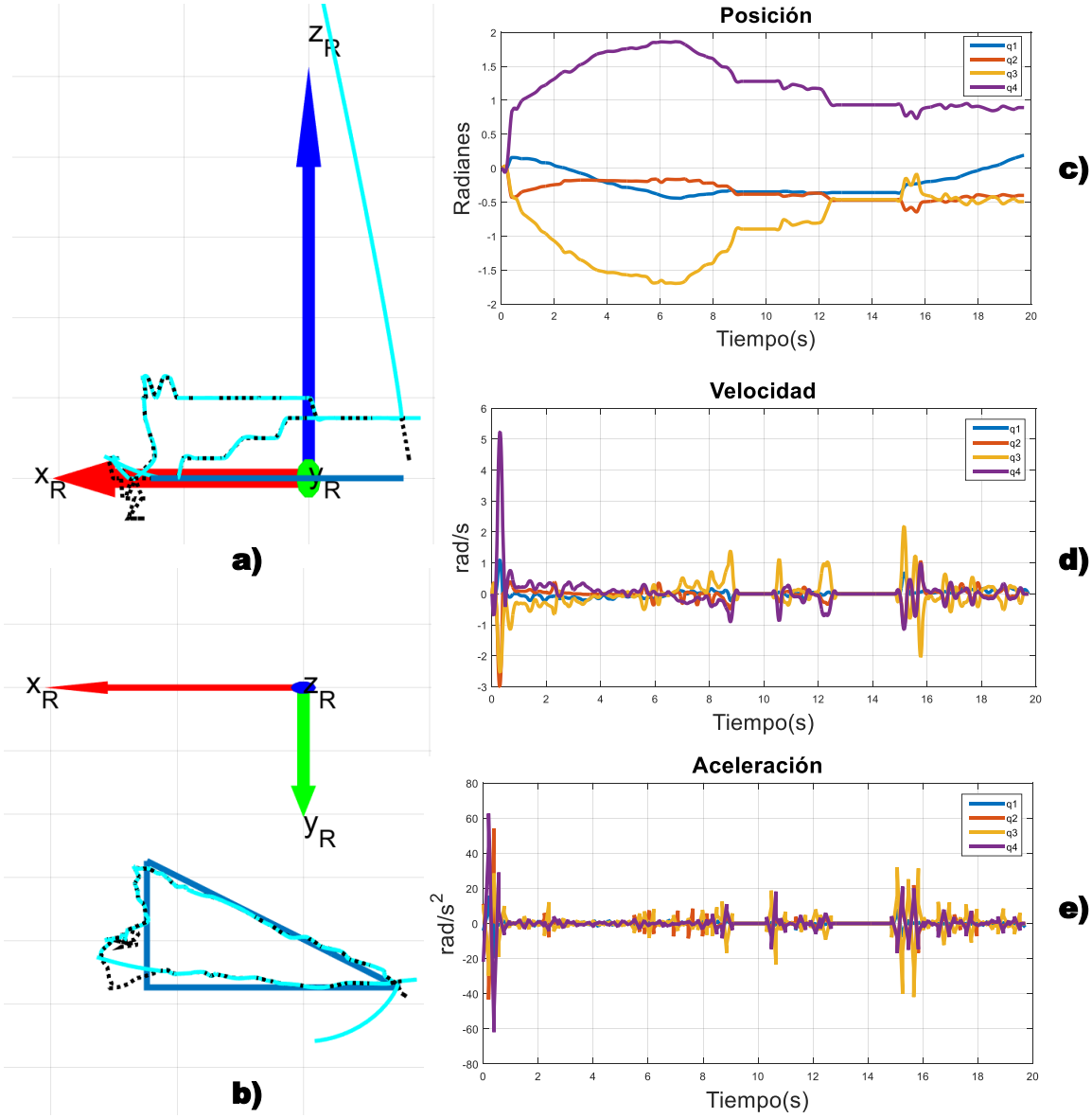
Anexo figura 5. Resultado de la trayectoria al aplicar el algoritmo de optimización vectorial para la diagonal.

Forma: Diagonal
Algoritmo: Analítico



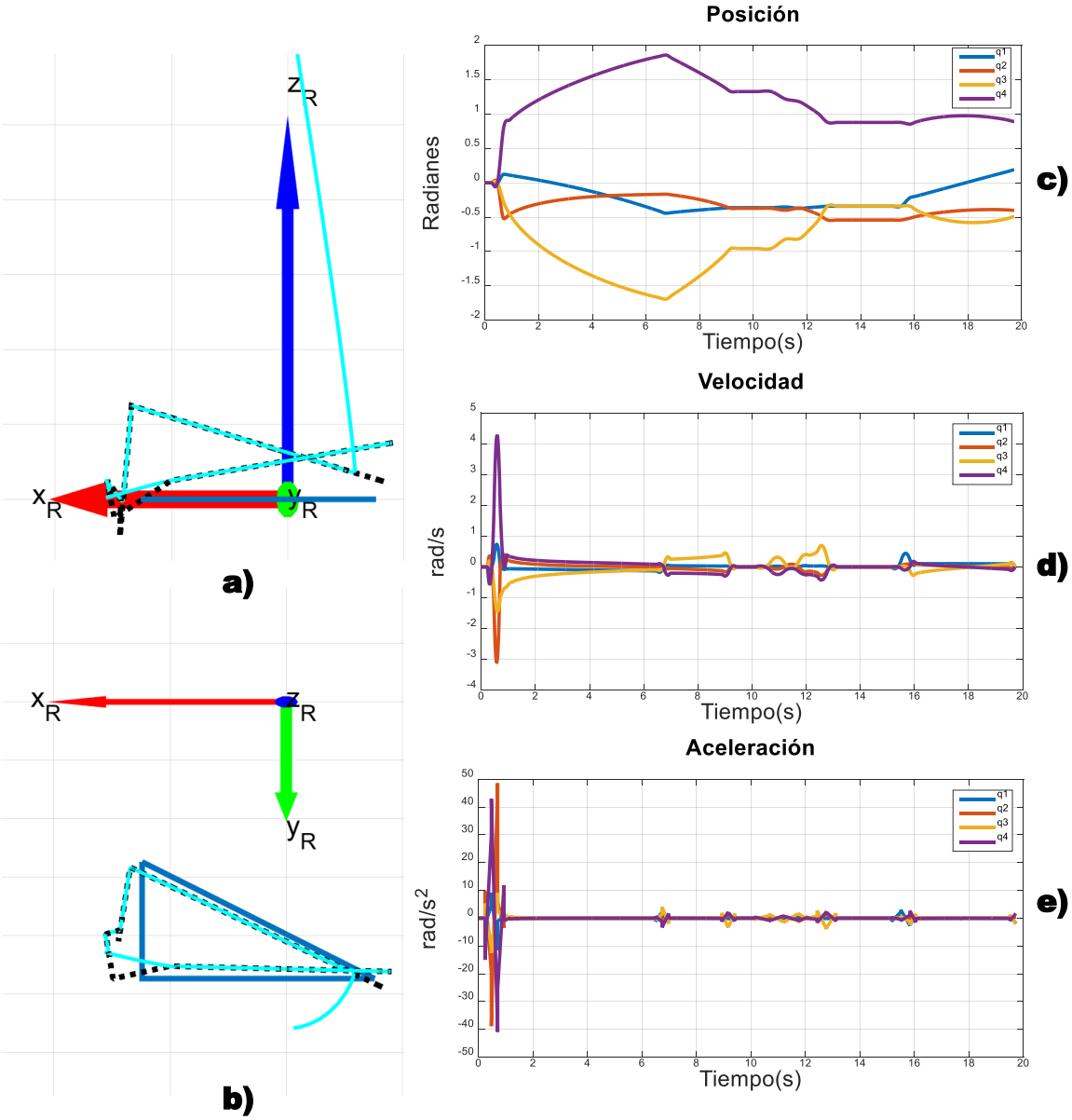
Anexo figura 6. Resultado de la trayectoria al aplicar el algoritmo analítico para la diagonal.

Forma: Triangulo
Algoritmo: Filtro



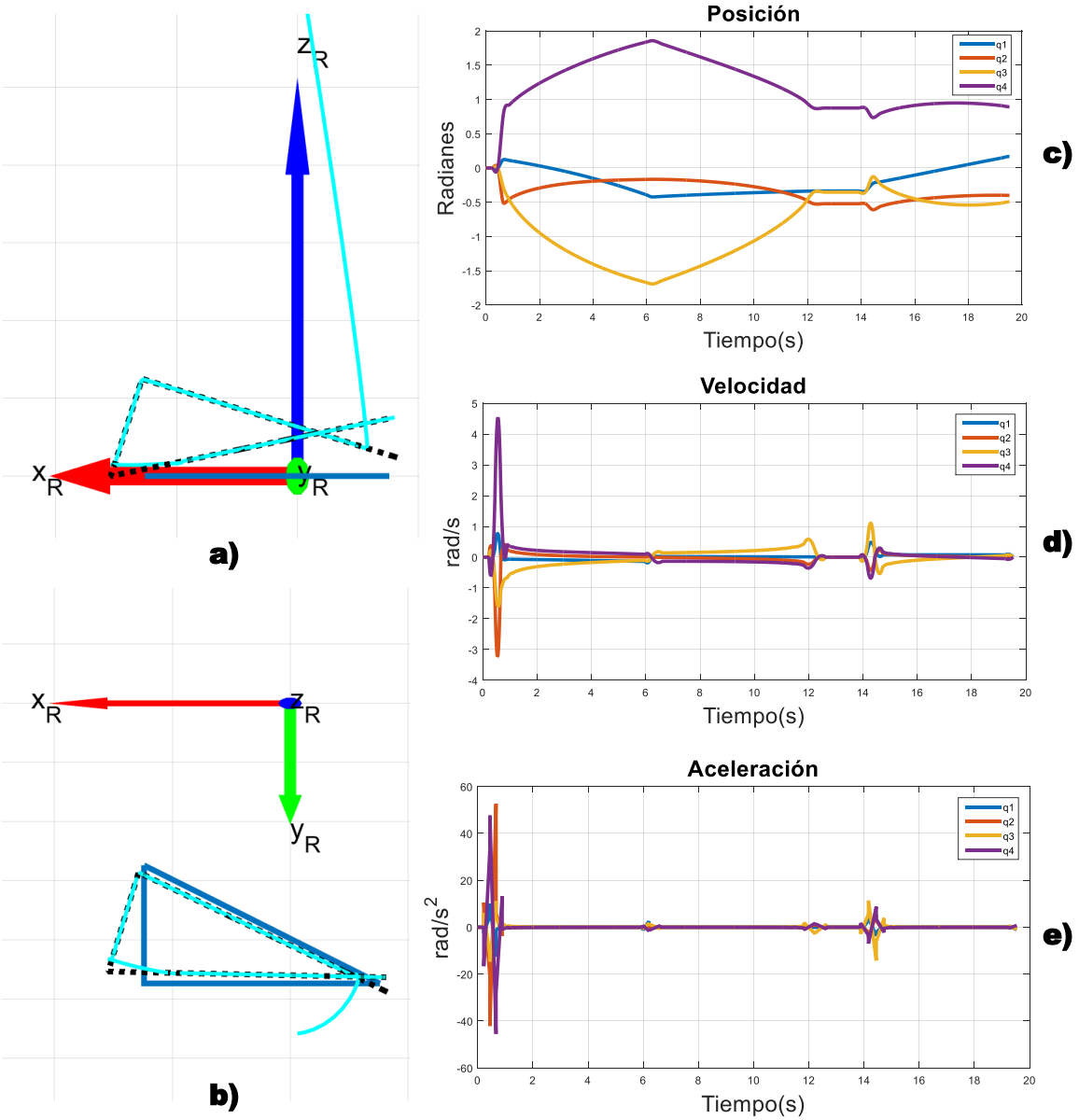
Anexo figura 7. Resultado de la trayectoria al aplicar el algoritmo filtro para el triángulo.

Forma: Triangulo
Algoritmo: Vectorial



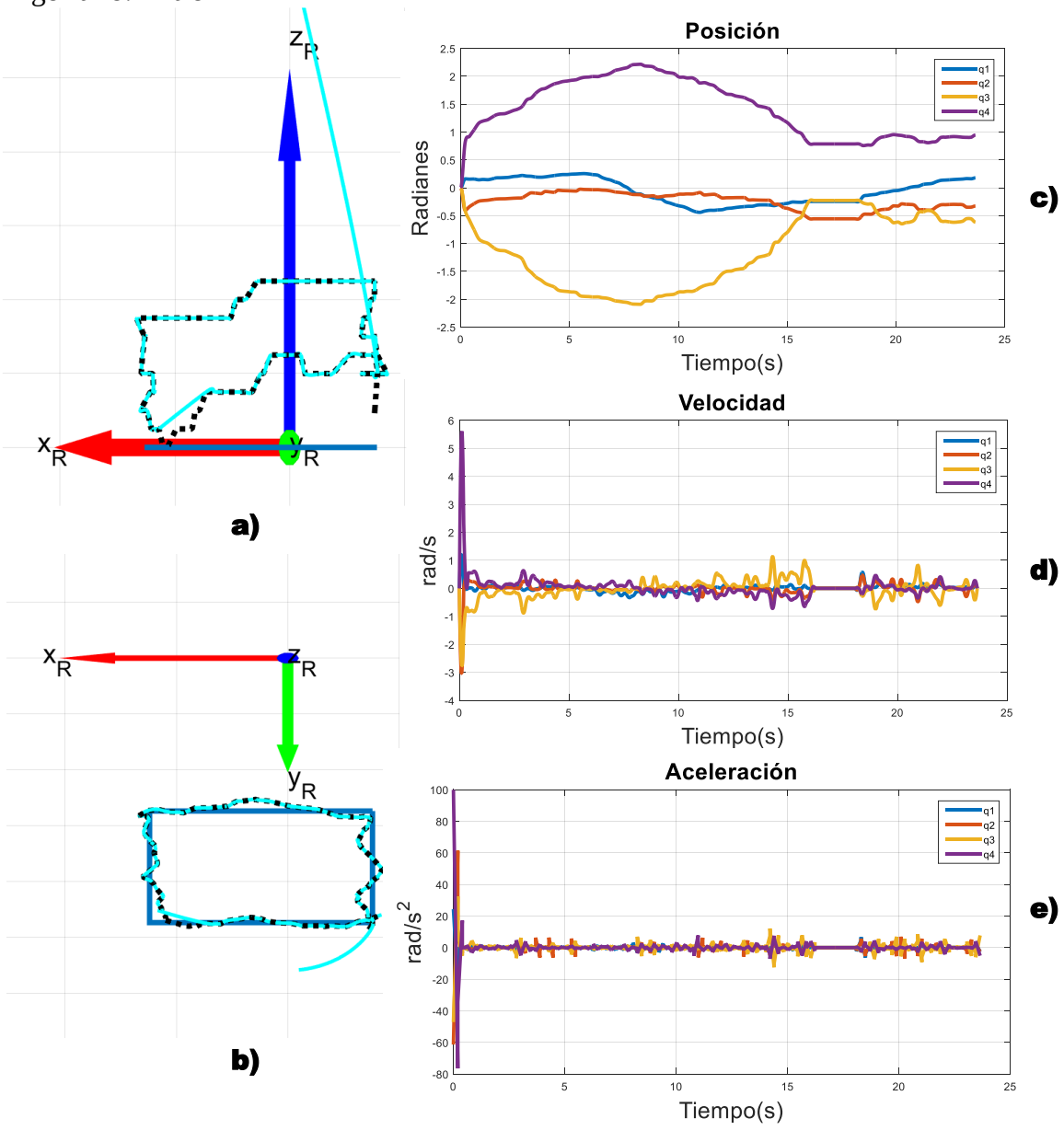
Anexo figura 8. Resultado de la trayectoria al aplicar el algoritmo vectorial para el triángulo.

Forma: Triangulo
Algoritmo: Analítico



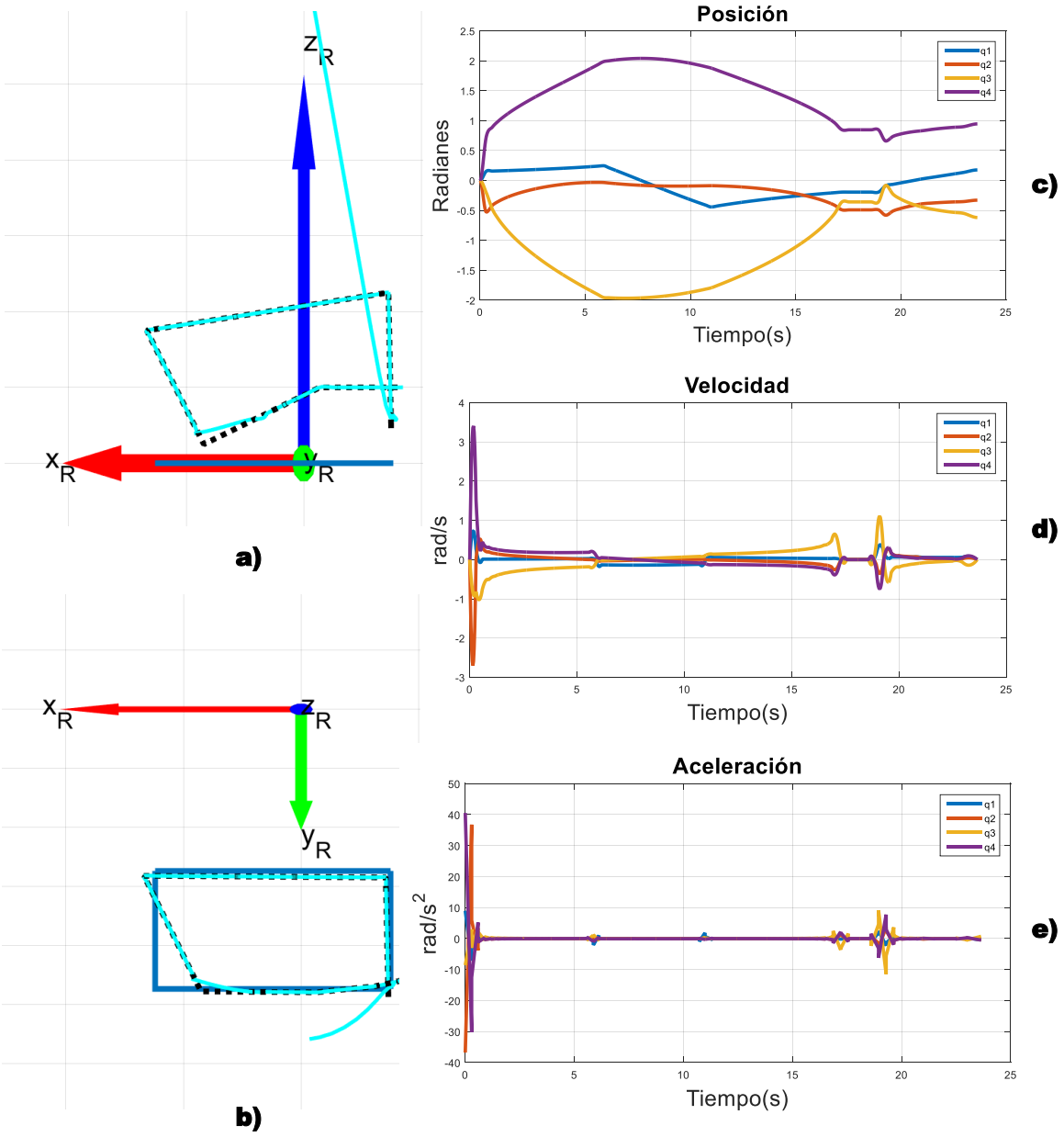
Anexo figura 9. Resultado de la trayectoria al aplicar el algoritmo analítico para el triángulo.

Forma: Cuadrado
Algoritmo: Filtro



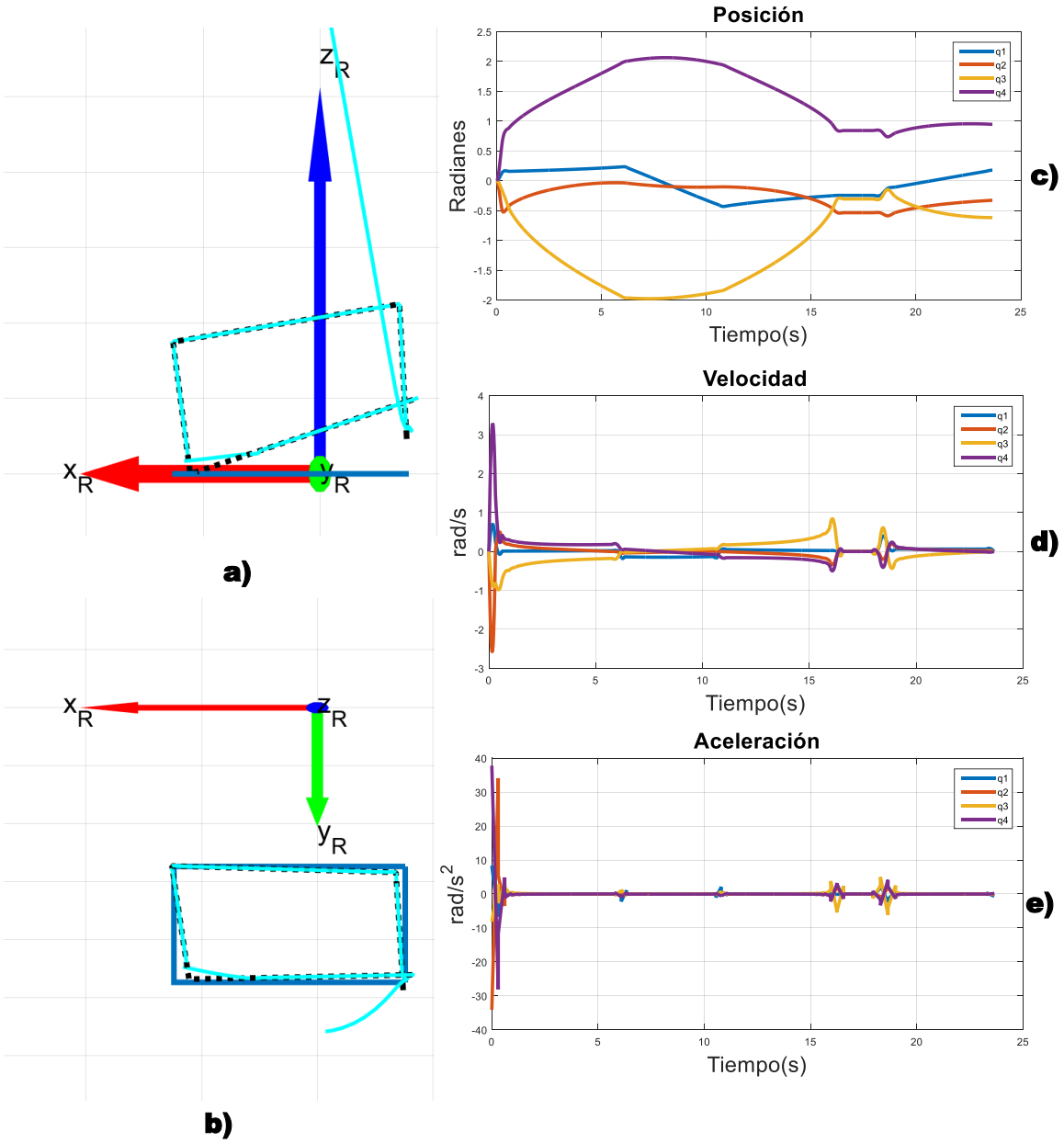
Anexo figura 10. Resultado de la trayectoria al aplicar el algoritmo filtro para el cuadrado.

Forma: Cuadrado
Algoritmo: Vectorial



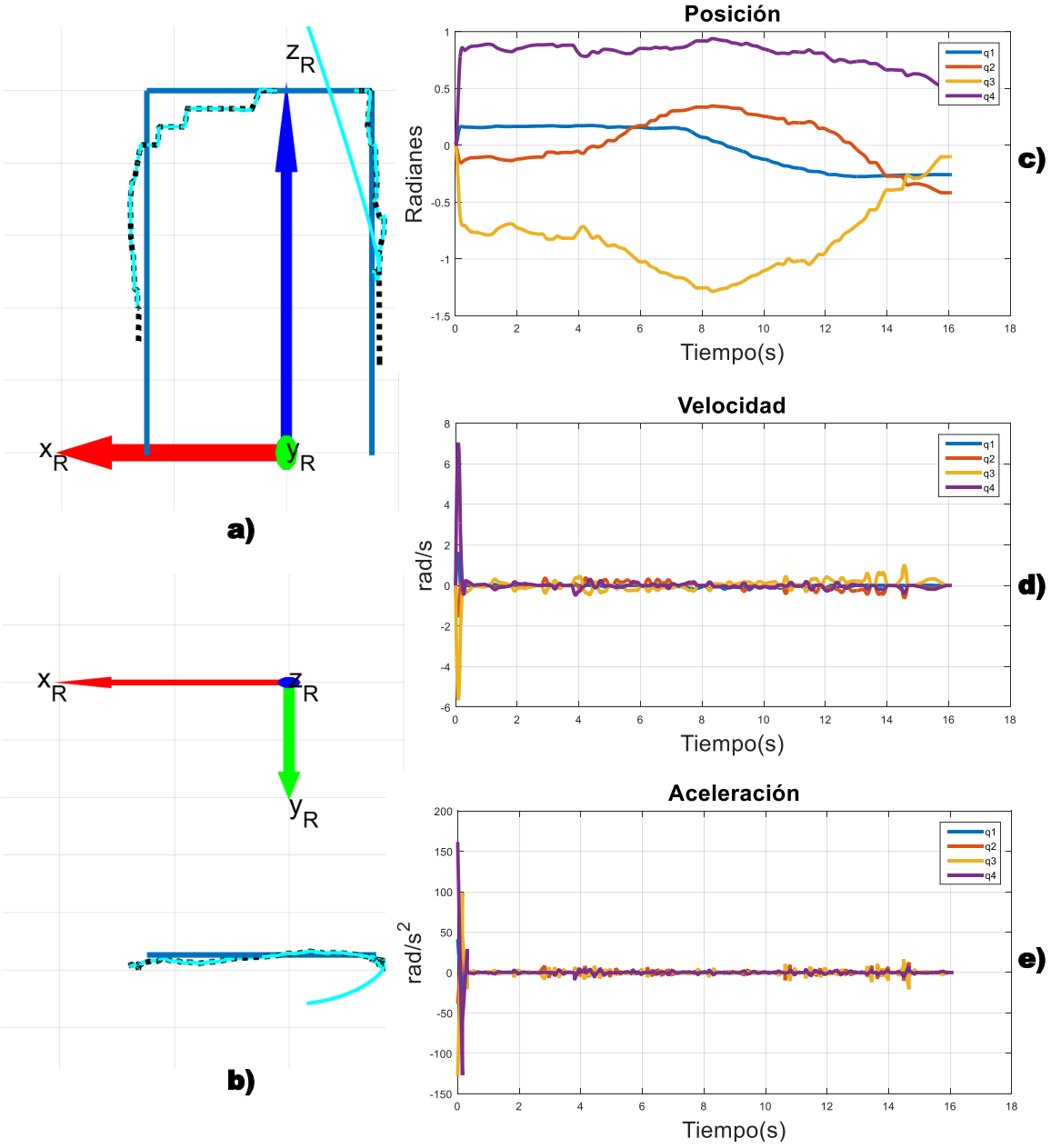
Anexo figura 11. Resultado de la trayectoria al aplicar el algoritmo vectorial para el cuadrado.

Forma: Cuadrado
Algoritmo: Analítico



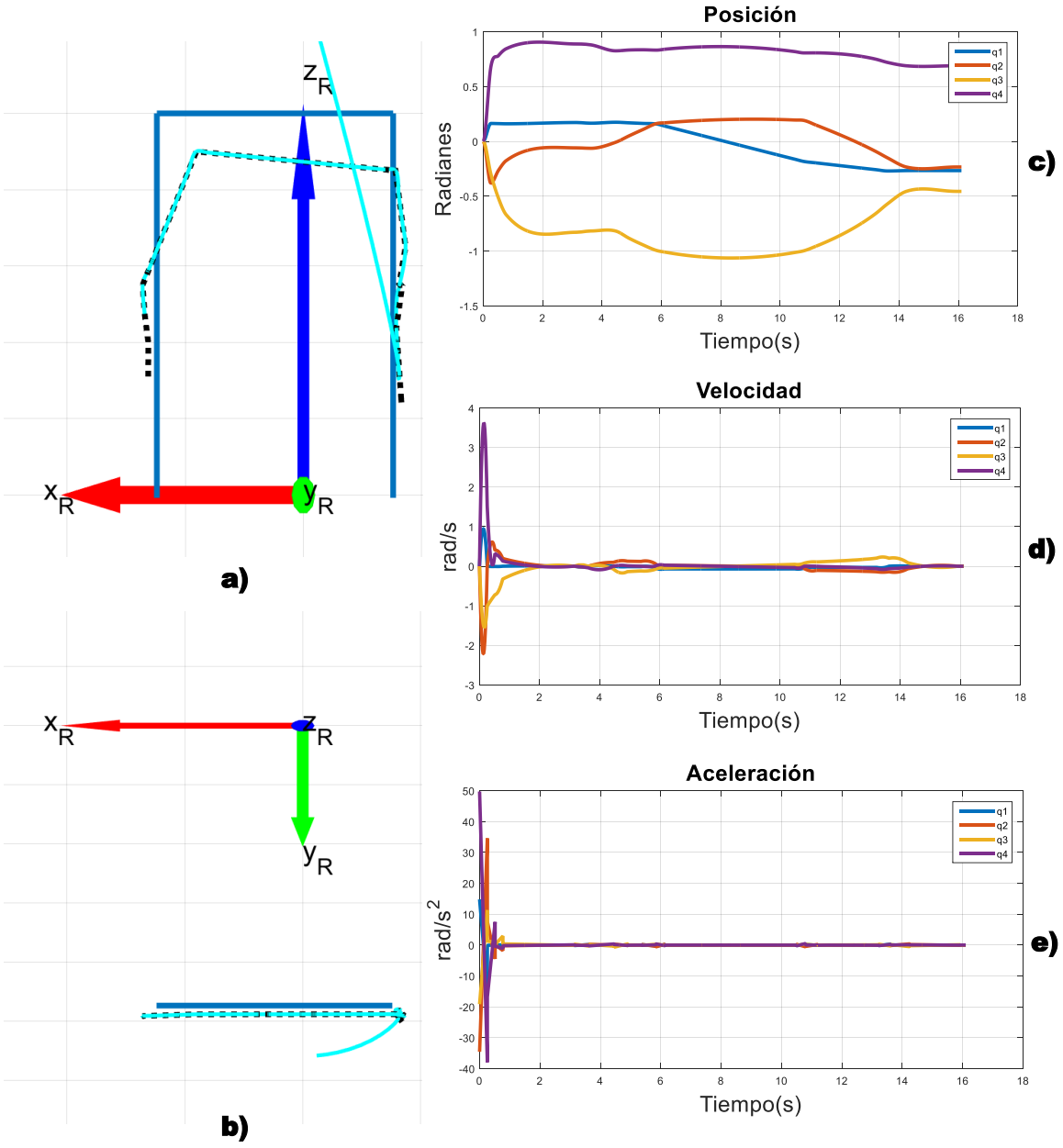
Anexo figura 12. Resultado de la trayectoria al aplicar el algoritmo analítico para el cuadrado.

Forma: Arco
Algoritmo: Filtro



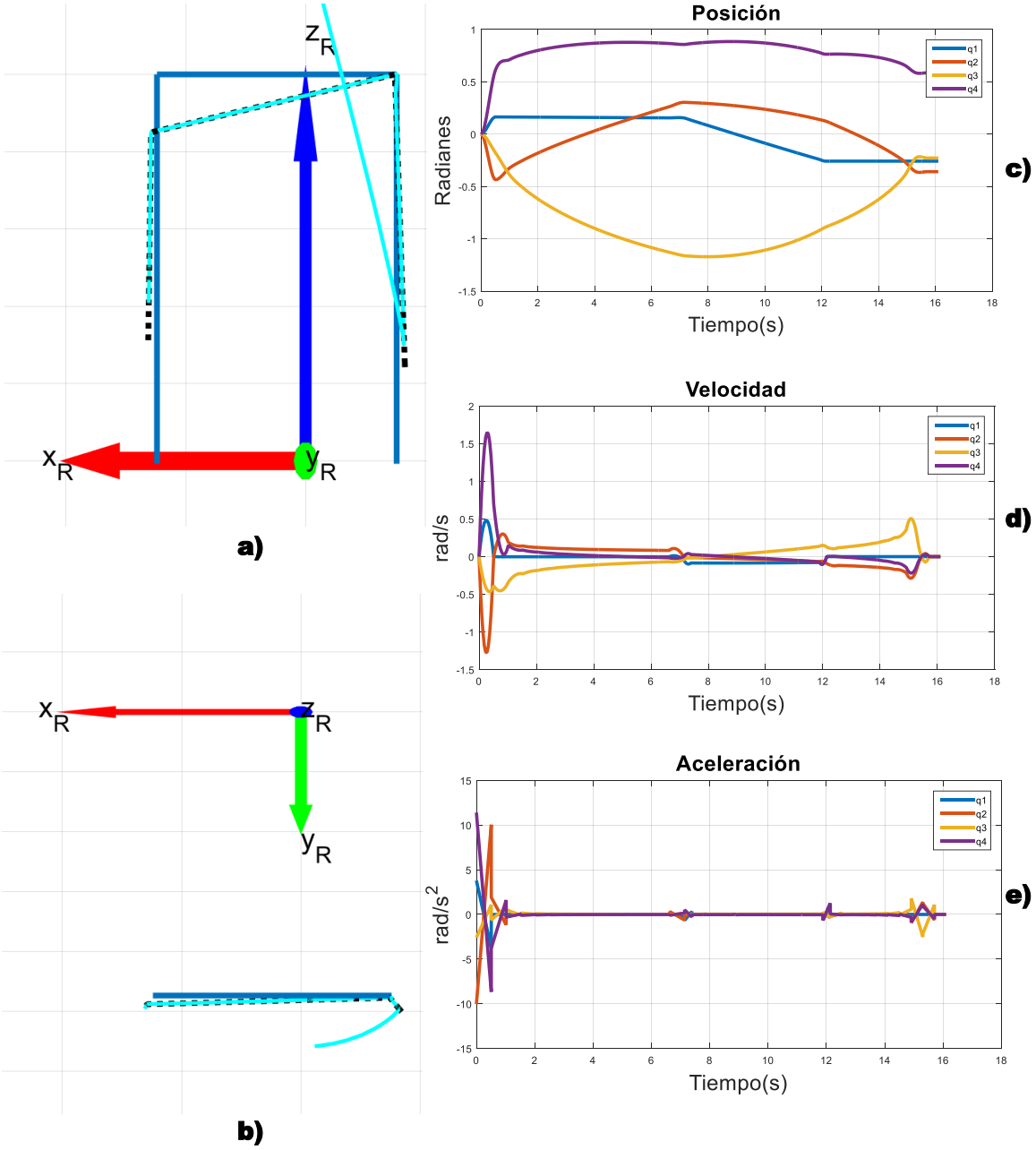
Anexo figura 13. Resultado de la trayectoria al aplicar el algoritmo filtro para el arco.

Forma: Cuadrado
Algoritmo: Vectorial



Anexo figura 14. Resultado de la trayectoria al aplicar el algoritmo vectorial para el arco.

Forma: Arco
Algoritmo: Analítico



Anexo figura 15. Resultado de la trayectoria al aplicar el algoritmo analítico para el arco.