UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Desarrollo de herramientas en software para el manejo, monitoreo y programación de la nueva versión del humanoide Robonova

Protocolo de trabajo de graduación presentado por Stefano Papadopolo,
estudiante de Ingeniería Mecatrónica

Guatemala,

2023

Resumen

Este protocolo de trabajo graduación se basa en la revitalización de los robots Robonova-1 de la Universidad del Valle de Guatemala. El objetivo de este proyecto es diseñar una herramienta de software para monitorear y programar los Robonova de forma gráfica. El proyecto consiste en el diseño del software y coreografías básicas, la comunicación con el controlador del robot y el monitoreo en tiempo real del mismo.

Para lograr lo anterior, se tomarán como ejemplo herramientas de software como RoboBASIC de David Buckley y Coreographe de Aldebaran Robotics, ambos siendo programas para facilitar la programación de robots humanoides por medio de una interfaz gráfica. Para lograr el movimiento correcto de este robot humaniode se tomará en cuenta la biomecánica detrás de los movimientos básicos de una persona como caminar y saludar. También se analizará el cuerpo como un robot de base flotante permitiendo modelar su cinemática y dinámica logrando así el control deseado. Para analizar matemáticamente y visualizar estos modelos se utilizarán librerías de Python específicas para el análisis físico y control robótico siendo estas pyBullet y Robotics Toolbox, respectivamente.

El proyecto de graduación descrito en este protocolo consistirá de 3 distintas etapas: diseño de la interfaz gráfica, diseño de rutinas y conexión y monitoreo en tiempo real. El diseño de la interfaz gráfica consistirá de diversas ventanas, cada una con un objetivo específico sea visuzalizar una simulación del robot, monitoreo en tiempo real del mismo, programación de las rutinas, control directo de los servomotores, etc. El diseño de rutinas, dentro de su propia ventana, perimitirá al programador seleccionar rutinas básicas de una librería y organizarlas de tal manera que el robot haga una coreografía completa concatenando estas rutinas. Por último, el programa también mantendrá una conexión en tiempo real al robot para monitorear el estado de los servomotores y mostrarlo en pantalla si así se desea.

Antecedentes

Este proyecto de graduación se trabajará con los robos humanoides Robonova sin embargo, debido a que no se han trabajado proyectos con estos robots dentro de la Universidad del Valle de Guatemala, los antecedentes a continuación serán obtenidos de fuentes externas a la institución. Estos antecedentes incluyen: la programación original en RobotBASIC de los Robonova como fue diseñado por David Buckley, un proyecto de robots bailarines autónomos que utilizó a los Robonova como prototipo y las características de otros robots humanoides como NAO de Aldebaran Robotics y Atlas de Boston Dynamics.

Programación en RoboBASIC

Originalmente, los robots Robonova eran programados a través del IDE RoboBASIC. Como su nombre lo indica, la programación de estos robots se llevaba a cabo en el lenguaje de programación BASIC. Para ello contaba con comandos generales de programación en BASIC además de comandos específicos para facilitar la programación de los Robonova, como lo es el control de los servos y la asignación de grupos de motores estipulada en el manual de RoboBASIC. Además de estos comandos para el control principal del robot, el

programa también contaba con comandos para funciones adicionales como lo son comandos de sonido para identificar y generar diversas frecuencias y comandos de control para LCD.

Además de la programación de alto nivel en la que se trabaja, el software de RoboBASIC también cuenta con varias interfaces para facilitar aún más el control de los robots. La primera, llamada "Catch-and-Play", permitía controlar cada servomotor de los Robonova en tiempo real a través de la interfaz gráfica que se muestra en la figura 1.

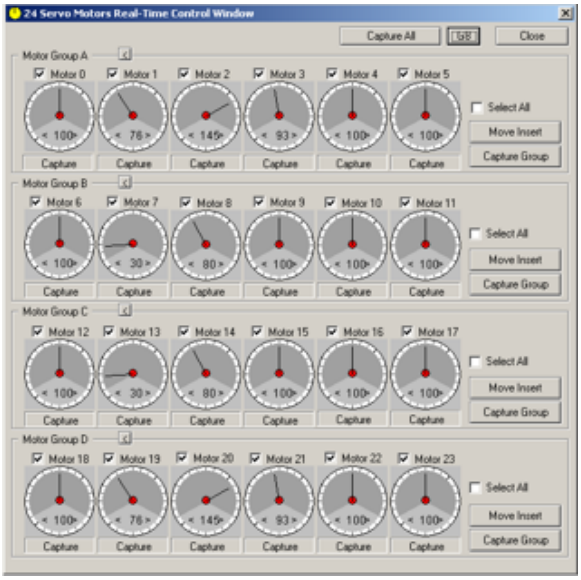

Figura 1: Captura de pantalla de la interfaz Catch and Play de RoboBASIC

Otra interfaz útil dentro de RoboBASIC era RoboScript. Esta podia considerarse como una combinación entre la interfaz de Catch-and-Play y la programación directa en BASIC. Contaba con comandos básicos necesarios para crear una rutina del robot además de poder modificar la posición de los servos a través de una interfaz gráfica que se muestra en la Figura 2.

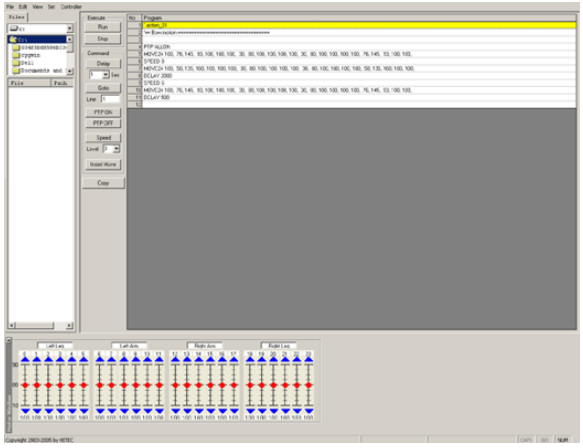

Figura 2: Captura de pantalla de la interfaz RoboScript de RoboBASIC

Por úlitmo, los Robonovas cuentan con un control remoto. Este puede ser utilizado para controlar directamente el movimiento del robot o para activar rutinas del mismo. Esta

configuración es establecida a través de RoboRemocon, la última interfaz auxiliar de RoboBASIC. Cabe notar que no se lleva a cabo ningún tipo de programación en esta como se hacía en las anteriores, RoboRemocon solamente asigna las funciones y rutinas ya programadas con alguna de las herramientas ateriores y las asigna a un botón específico del control. La interfaz se muestra en la Figura 3.

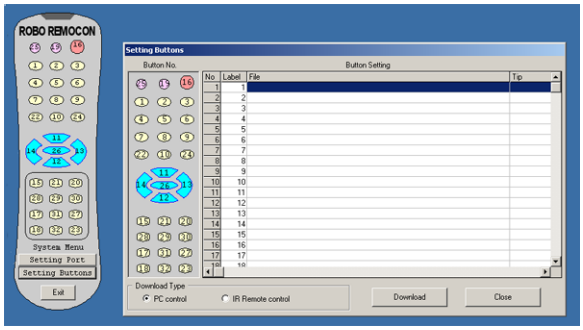

Figura 3: Captura de pantalla de la interfaz RoboReomocon de RoboBASIC

Programación de robots bailarines

En [1] se presenta un proyecto cuyo objetivo era crear un programa para que los robots Robonova-1 reaccionaran a señales de audio y bailen al ritmo de la música.

Los algoritmos de control de este proyecto se dividieron en 3 etapas: identificación del ritmo, la plataforma a utilizar (Robonova-1) y la generación de gestos y control. Para la primera etapa de identificación de ritmo, dado que el Robonova no contaba con el hardware necesario para el procesamiento de las señales, se realizó el programa en una computadora externa al robot la cual se comunicaba con el mismo por medio de Bluetooth. Este programa externo se encargaba tanto de el procesamiento de la música como la transmisión de los gestos al robot. El baile del Robonova consistía en una concatenación de diferentes gestos ya programados cuya velocidad de acción y de cambio entre gestos era regida por el ritmo analizado anteriormente.

Además de implementar el programa para hacer que el robot bailase, el equipo también creó un nuevo ecosistema para controlarlo en lugar de utilizar RoboBASIC. Lograron crear un mejor ecosistema de control que contaba con una latencia mucho menor a la obtenida al utilizar RoboBASIC (0.2 segundos) además de contar con una mayor exactitud en cuanto a las posiciónes deseadas de los servos. Como se evidencia en la Figura 4.

NAO de Aldebaran Robotics

Atlas de Boston Dynamics

Una de las compañías más innovadoras en el área de la robótica es Boston Dynamics. En su repertorio de robots cuentan con varios robots cuadrúpedos (Spot, Big Dog,LS3, etc.), robots con ruedas (Handle) y robots humanoides como es el caso del robot Atlas.

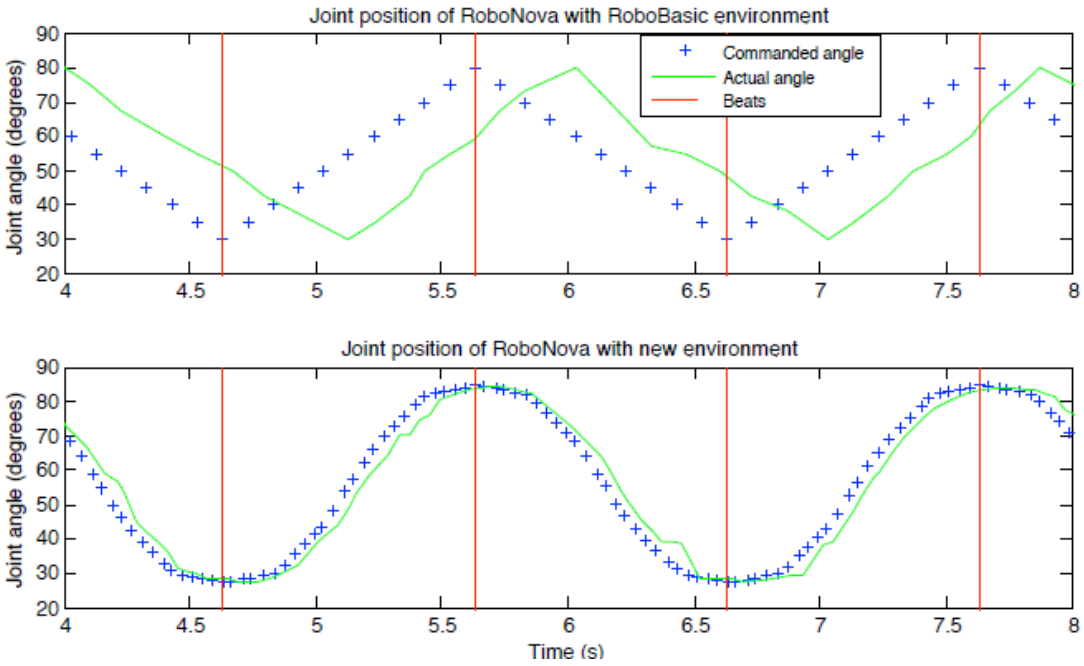

Figura 4: Comparación entre RoboBASIC y el ecosistema creado por investigadores de Drexel University

Con 28 grados de libertad y un sistema de control de estado del arte, Atlas es uno de los robots humanoides con mejor movilidad en la academia. Con este robot, Boston Dynamics demuestra los posibles usos de robots humanoides y como estos podrían revolucionar la industria. Este se puede observar en la Figura 5.

Figura 5: Robot Humanoide Atlas de Boston Dynamics

Justificación

En los últimos años se han dado bastantes avances en robótica gracias a la investigación y desarrollo que varias instituciones llevan a cabo utilizando robots humanoides.Desde la

manufactura de piezas mecánicas más eficientes hasta la programación de sistemas de control más avanzados y robustos, el uso de estos robots en investigación ha aportado bastante en el área de ingeniería. Por esta razón, este proyecto se llevará a cabo con el fin de revitalizar los robots humanoides Robonova-1 disponibles en la Universidad del Valle de Guatemala. Lo que se desea es actualizar su interfaz de programación además de su conexión y control de forma inalámbrica. El contar con estas herramientas aplicadas a un prototipo como el Robonova mejorará la investigación y aprendizaje dentro de la universidad en las áreas anteriormente mencionadas y abriría el campo para el futuro control de robots humanoides más sofisticados.

Como se pudo observar en los antecedentes, la plataforma RoboBASIC que se utiliza por defecto para la programación de los Robonova, a pesar de contar con diversas interfaces para facilitar la creación de rutinas, no es tan amigable y ya no cuenta con soporte, con su última actualización siendo en mayo de 2008. Además de ello, no cuenta con monitoreo en software de el estado actual del Robonova lo cual es una característica importante para el control de robots. Lo que se busca entonces es crear una interfaz de programación similar a Coreographe (la interfaz utilizada por los robots NAO). Esta debe permitir al usuario observar el estado de los robots y crear las rutinas en una interfaz comprensible y amigable.

Objetivos

Objetivo general

Diseñar una herramienta de software para el monitoreo y programación de los robots humanoides Robonova-1 de forma gráfica.

Objetivos específicos

- Conectar la herramienta de software con el robot de forma inalámbrica para su programación y monitoreo en tiempo real.
- Crear una librería de subrutinas para facilitar la creación de coreografías complejas.
- Crear una interfaz gráfica amigable y sencilla para el fácil desarrollo de coreografías de los robots.

Marco teórico

Biomecánica del movimiento humano

Para lograr que un robot bípedo emule correctamente los movimientos de una persona, primero se debe empezar por analizar la mecánica de una persona en movimiento. El área de estudio que se enfoca en esto se denomina biomecánica. Esta se enfoca en modelar a los seres vivos por medio de representaciones físicas más sencillas como masas, eslabones,

uniones, resortes, etcétera, para poder implementar un análisis cinemático y dinámico de su movimiento.

Como se puede observar en la Figura 6, un modelo simple para una persona (o robot humanoide) es un conjunto de eslabones unidos por diferentes tipos de uniones. Durante su movimiento, estos eslabones se moverán de acuerdo a los grados de libertad que cada unión permita. Para modelos biomecánicos sencillos, estas uniones generalmente son revolutas con un grado de libertad. Cabe notar que, a demás de las articulaciones, también se encuentra marcado el centro de masa del modelo al rededor del cual se llevan a cabo los cálculos.

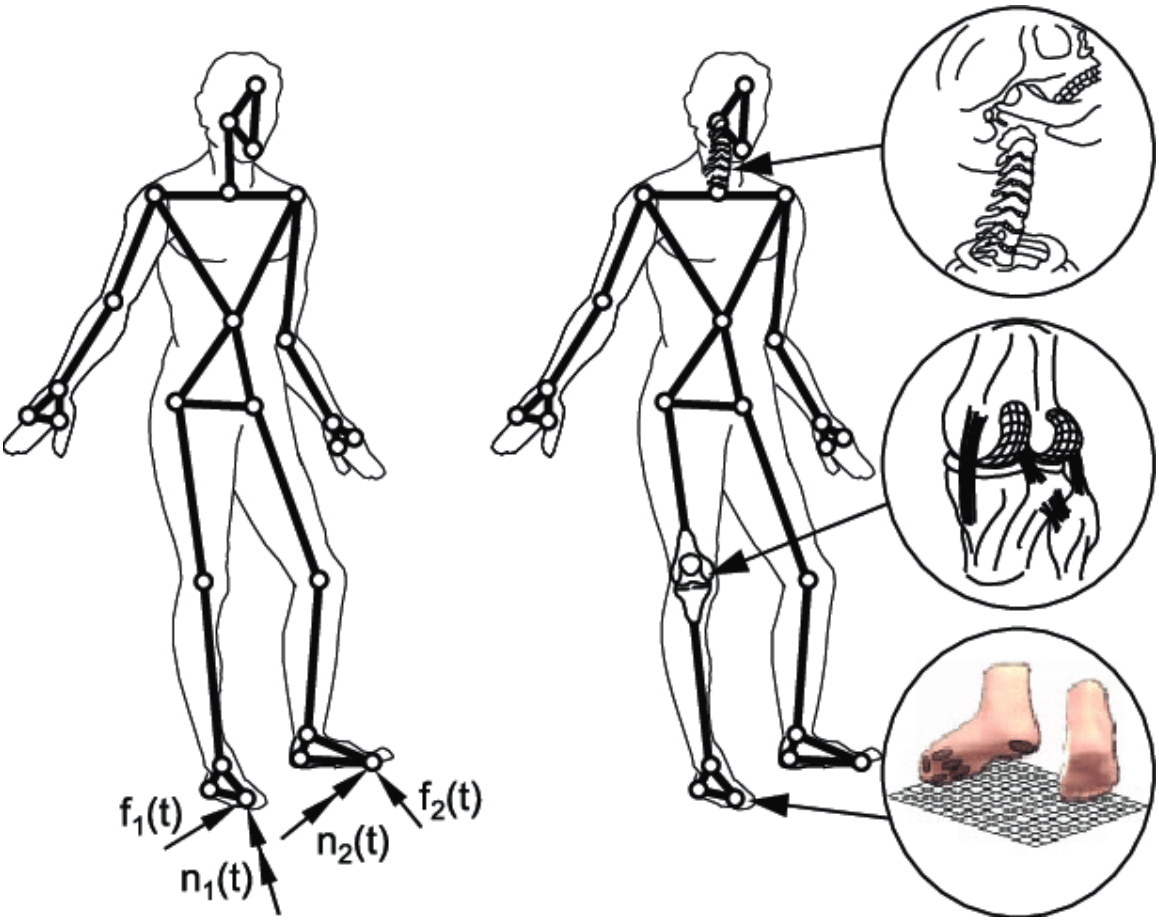

Figura 6: Modelo de juntas y eslabones de un sistema biomecánico

Ciclo de Gait

Al llevar a cabo movimientos que se repiten una y otra vez como es el caso de caminar y correr, estos pueden ser analizados por medio de un ciclo de Gait. Este ciclo separa los movimientos es distintas fases para su análisis. El ciclo de Gait de caminar puede ser de dos fases: la postura y el balanceo. Durante la fase de postura, la cual es el 60 % del ciclo de Gait [2], la pierna se encuentra en contacto con el piso. Este contacto se subdivide en 5 fases: *Heel strike*, ocurre cuando el talón entra en contacto con el piso; *loading response*, cuando este se prepara para cargar con todo el peso del cuerpo; *midstance*, donde todo el peso se encuentra

sobre esa pierna; *heel off*, cuando el pie comienza a levantarse del piso; y por último, *toe off*, cuando ya se levantó por completo. Durante la fase de balanceo, como lo indica su nombre, la pierna se balancea con momentum hacia adelante para llegar a su nueva posición. Esta fase también puede ser dividida en 3 fases propias: aceleración, balanceo y desaceleración. Para un bípedo, el proceso de caminar consiste en un ciclo de gait continuo en cada pierna, desfazados entre sí. [2]

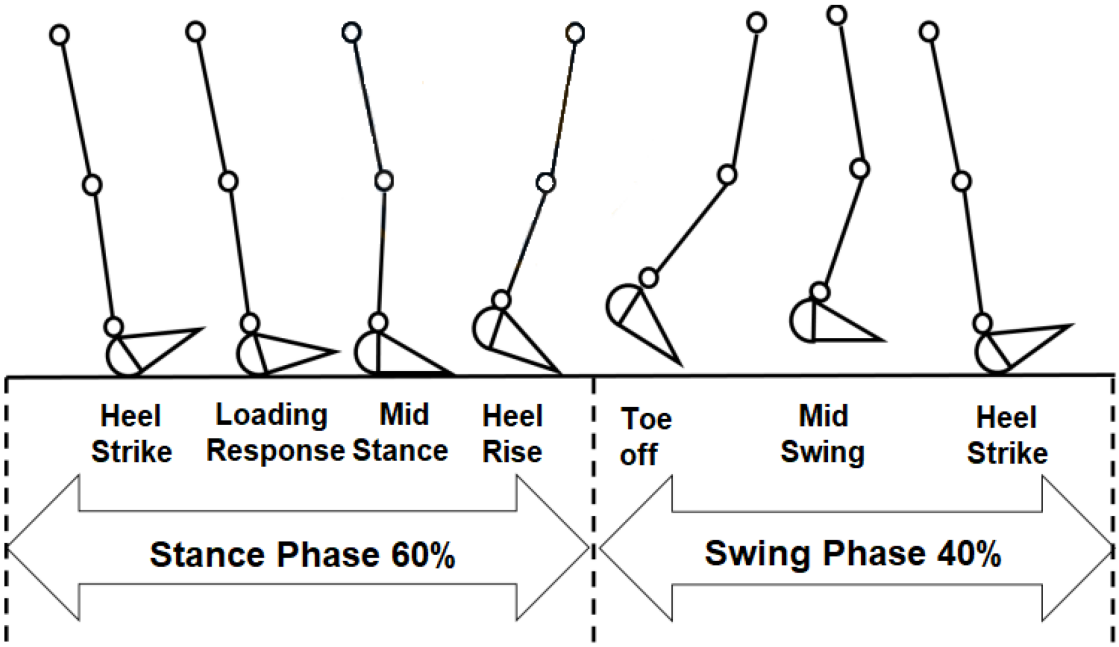

Figura 7: Ciclo de Gait

Cinemática de robots humanoides

Un robot humanoide es un tipo específico de robot de base flotante cuya forma tiene inspiración en el cuerpo humano. Por ello, la mayoría de estos robots cuentan con 4 extremidades: 2 brazos y 2 piernas. En los robots de base flotante, generalmente, su base se encuentra en el centro de masa del mismo. Ya que no es posible actuar sobre la base directamente este tipo de robot moviliza su base de forma indirecta controlando sus extremidades.[3]

Para el control de las extremidades del robot se puede considerar cada una como un manipulador serial independiente saliendo de la base. Como se observa en [4], cada una de las extremidades tendrá su propia matriz de Denevit-Hartenberg en relación a la base flotante del robot. Al plantear el robot de esta manera, la cinemática directa e inversa de los brazos del mismo se vuelve trivial.

Este no es el caso para las piernas del robot. Debido a que estas deben sostener el peso entero del robot, además del control cinemático que se lleva a cabo, también deben tomarse en cuenta consideraciones dinámicas al controlar las piernas. En [5], se muestran dos modelos para tomar en cuenta estas consideraciones: el modelo LIMP (Linear Inverted Pendulum Model) y el modelo ZMP (Zero Moment Point). En el modelo LIMP, las piernas actuan como

una distancia desde el pivote (el suelo) hasta el centro de masa del robot, simplificando así el análisis fisico. Una vez modelado de esta forma el ZMP es el posicionamiento del centro de masa de tal manera que este no genere un torque en el plano xy evitando así que el robot pierda su estabilidad al momento de dar un paso.

PyBullet

PyBullet es una librería para Python del Bullet Physics engine. Esta librería se enfoca en el estudio de detección de colisiones y dinámica de cuerpos rígidos. La librería cuenta con simulaciones de dinámica directa, dinámica inversa, detección de colisiones y intersección de rayos. Esta librería se utiliza principalmente la simulación de sistemas de robótica antes de su implementación física además de tener aplicaciones en videojuegos y realidad virtual.

Con el área de interés siendo simulaciones de robótica, PyBullet cuenta con herramientas para crear y analizar diversos tipos de robots. Además de contar con una librería de robots de ejemplo, PyBullet cuenta con comandos para cargar la información de un robot especifico en diversos formatos como URDF, SDF, MJCF, entre otros. Una vez cargados los datos del robot se puede analizar y controlar dentro del espacio de simulación. PyBullet cuenta con funciones para obtener las propiedades físicas del robot además de funciones que le permiten al programador controlar el mismo.[6]

Además de esto, la librería también cuenta con funciones para el cálculo mátemático de la cinemática y dinámica inversa de los robots, detección de colisiones, visión de computadora y realidad virtual.[6]

Robotics Toolbox de Peter Corke

Esta librería es una versión planteada para Python de la Robotics Toolbox para MATLAB de Peter Corke, y cuenta con herramientas para el análisis físico de robots. Utilizando todas estas herramientas, el programa puede modelar robots, analizar y crear trayectorias, llevar a cabo manipuulación simbólica de variables, analizar la cinemática diferencial del robot e incluso la dinámica del mismo.[7]

La Robotics Toolbox ya cuenta con más de 30 modelos cinemáticos de diversos tipos de robots. Además de esto, es posible utilizarla para modelar robots propios de diversas maneras. Se puede generar la matriz de Denavit-Hartenberg con las dimensiones adecuadas del robot o también se pueden cargar los datos del mismo por medio de un string ETS o un archivo URDF. Además de los parámetros cinemáticos, también se pueden agregar otros parametros como la masa, CoG, inercia de los motores, limites de las articulaciónes, etcetera, utilizando palabras clave dentro de la programación.

En cuanto a las trayectorias, el Robotics Toolbox trabaja con polinomios de quinto orden para proporcionar un *jerk* continuo. Estas trayectorias proporcionan tanto la posición, velocidad y aceleración de los actuadores para llegar al punto deseado.

Como se estipuló anteriormente, la Robotics Toolbox también computa cinemática diferencial de los robots. Puede computar los Jacobianos, Hessianos y la manipulabilidad de

todos los robots.

Por último, también es posible analizar la dinámica de los robots utilizando integración numérica. Esta se puede utilizar para calcular los términos de inercia, Coriolis y gravedad por medio de dinámica inversa.[7]

Metodología

Interfaz gráfica

- Diseño visual de la interfaz gráfica: Se utilizará Tkinter de Python para crear la interfaz gráfica del programa. Esta interfaz contará con diferentes ventanas dependiendo de lo que se desee hacer. Estas serán:
  - Definición del robot: se extraerá la información de un URDF o una matriz de Denavit-Hartenberg para la definición del robot en el sistema. Una vez extraída, debe mostrar un modelo físico del robot.
  - Control de servomotores: mostrar todos los servomotores con un slider para variar su posición. Se podrán observar estas variaciones en el ecosistema de pyBullet o en tiempo real. Esto se logrará conectando por programación los sliders a la variable de control, q, que controlará los servomotores del robot.
  - Generación de rutinas: Se utilizará una función base donde se guarará un registro de los movimientos deseados de los servomotores
  - Coreografía: Ventana que permitirá seleccionar las rutinas creadas anteriormente y colocarlas en un orden específico para generar coreografías más complejas.
- Integración de pyBullet para mostrar modelos 3D: Dentro de la interfaz se mostrarán modelos físicos del robot con el fin de simular su movimiento antes de programar el robot físico. Para visualizar esta simulación, se integrará la librería de pyBullet al programa la cual se utilizará, en conjunto con la librería de Robotics Toolbox, para programar y visualizar la cinemática del robot.

Diseño de rutinas

- Generación de rutinas con cinemática directa: Dentro de la pantalla de control de servomotores se podrán guardar los cambios realizados a los servos para posicionar las extremidades en la pose deseada dentro del espacio de configuración.
- Generación de rutinas con cinemática inversa: Dentro del ecososistema de pyBullet, con la ayuda del Robotics Toolbox, se diseñarán rutinas desde el espacio de tarea logrando movimientos más precisos del Robonova-1. Esto se basará en el análisis de cinemática inversa logrando obtener el vector de configuración desde la pose deseada en lugar del inverso que se logró en la etapa anterior.
- Generación de coreografías complejas: Se utilizarán timers y las funciones creadas anteriormente para las rutinas.

Conexión para programación y monitoreo en tiempo real

- Programación: Utilizando un ESP32 como microcontrolador para el robot, se programará el control de los servomotores del nuevo Robonova-1. Este programa no solo controlará los servomotores del robot, sino que enviará los datos de los mismos de regreso a la computadora para poder ser analizados y utilizados en la cinemática inversa de los mismos.
- Monitoreo: Se ntegrará un sistema de registro que actualice constantemente las posiciones actuales de los servomotores. Cuando el robot esté funcionando, esta información será introducida a pyBullet para observar el mismo movimiento dentro de la simulación.

Cronograma de actividades

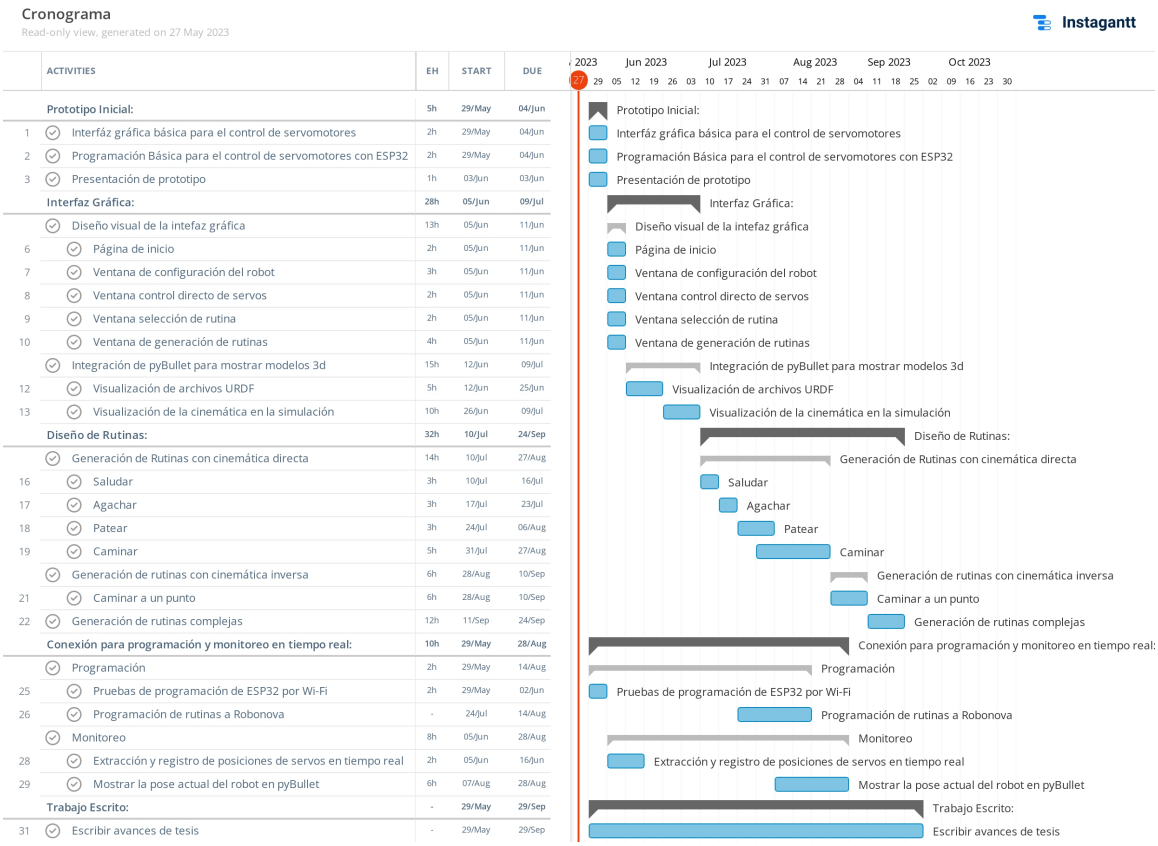

Figura 8: Cronograma de Proyecto de Graduación

Índice preliminar

Referencias

[1] D. Grunberg, R. Ellenberg, I. Kim, J. Oh, P. Oh e Y. Kim, "Development of an Autonomous Dancing Robot," 2010.

[2] S. S. Ekka. "Gait definition, its phases & abnormal gait." Running Time: 59 Section: Health & Fitness. (16 de oct. de 2016), dirección: `https://physiosunit.com/gait-definition-phases-of-cycle-explained/` (visitado 20-05-2023).

[3] Miguel Zea, *Cinemática de robots de base flotante, fuerzas de contacto y el modelo de fricción de Coulomb*, 2023.

[4] N. Kofinas, "Forward and inverse kinematics for the NAO humanoid robot," 2012.

[5] K. Erbatur y O. Kurt, "Natural ZMP trajectories for biped robot reference generation," *IEEE Transactions on Industrial Electronics*, vol. 56, n.º 3, págs. 835-845, mar. de 2009, ISSN: 0278-0046. DOI: 10.1109/TIE.2008.2005150. dirección: http://ieeexplore.ieee.org/document/4633623/ (visitado 22-05-2023).

[6] E. Coumans e Y. Bai, "PyBullet quickstart guide," 2022.

[7] P. Corke y J. Haviland, "Not your grandmother's toolbox – the robotics toolbox reinvented for python," en *2021 IEEE International Conference on Robotics and Automation (ICRA)*, Xi'an, China: IEEE, 30 de mayo de 2021, págs. 11 357-11 363, ISBN: 978-1-72819-077-8. DOI: 10.1109/ICRA48506.2021.9561366. dirección: https://ieeexplore.ieee.org/document/9561366/ (visitado 21-05-2023).