UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE QUITO

CARRERA DE MECATRÓNICA

DESARROLLO DE UN PLANIFICADOR DE TRAYECTORIA PARA UN BRAZO ROBOT EN CELEC EP HIDROAGOYÁN PARA EL PROCESO DE SOLDADURA DE ÁLABES DE TURBINAS DESGASTADAS TIPO PELTON A PARTIR DE MODELOS 3D GENERADOS MEDIANTE UN ESCÁNER

Trabajo de titulación previo a la obtención del
Título de Ingeniero en Mecatrónica

AUTOR: **LUIS FERNANDO SILVA ARIAS**
TUTORA: **CARMEN JOHANNA CELI SÁNCHEZ**

Quito – Ecuador
2026

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Yo, Luis Fernando Silva Arias con documento de identificación N° 1724368400 manifiesto que:

Soy autor y responsable del presente trabajo; y, autorizo a que sin fines de lucro la Universidad Politécnica Salesiana puede usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Quito, 15 de enero del 2026

Atentamente,

Luis Fernando Silva Arias
1724368400

CERTIFICADO DE CESIÓN DE DERECHO DE AUTOR DEL TRABAJO DEL TITULACIÓN A LA UNIERSIDAD POLITÉCNICA SALESIANA

Yo, Luis Fernando Silva Arias con documento de identificación N° 1724368400, expreso mi voluntad y por medio del presente documento cedo a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que soy el autor del Dispositivo Tecnológico: "Desarrollo de un planificador de trayectoria para un brazo robot en CELEC EP Hidroagoyán para el proceso de soldadura de álabes de turbinas desgastadas tipo Pelton a partir de modelos 3d generados mediante un escáner", el cual ha sido desarrollado para optar por el título de: Ingeniero en Mecatrónica, en la Universidad Politécnica Salesiana, quedando a Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribo este documento en el momento que hago la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Quito, 15 de enero del 2026

Atentamente,

Luis Fernando Silva Arias
1724368400

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Carmen Johanna Celi Sánchez con documento de identificación N° 1717437808, docente de la Universidad Politécnica Salesiana, declaro que bajo mi autoría fue desarrollado el trabajo de titulación: DESARROLLO DE UN PLANIFICADOR DE TRAYECTORIA PARA UN BRAZO ROBOT EN CELEC EP HIDROAGOYÁN PARA EL PROCESO DE SOLDADURA DE ÁLABES DE TURBINAS DESGASTADAS TIPO PELTON A PARTIR DE MODELOS 3D GENERADOS MEDIANTE UN ESCÁNER, realizado por Luis Fernando Silva Arias con documento de identificación N° 1724368400, obtenido como resultado final el trabajo de titulación bajo la opción Dispositivo Tecnológico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Quito, 15 de enero del 2026

Atentamente,

Ing. Carmen Johanna Celi Sánchez, MSc.
1717437808

Baños de Agua Santa, 13 de enero de 2026

CARTA DE AUTORIZACIÓN DE PUBLICACIÓN

Yo, Carlos Andrés Guerra Vásquez, en calidad de Jefe de Ingeniería de Recuperación, Diseño y Optimización – CIRT, y como representante de la Corporación Eléctrica del Ecuador – CELEC EP, Unidad de Negocio Hidroagoyán, por medio del presente manifiesto que:

Autorizo al señor Luis Fernando Silva Arias, con número de cédula 1724368400, estudiante de la carrera de Ingeniería Mecatrónica de la Universidad Politécnica Salesiana, la publicación y difusión académica de su trabajo de titulación titulado:

"DESARROLLO DE UN PLANIFICADOR DE TRAYECTORIA PARA UN BRAZO ROBOT EN CELEC EP HIDROAGOYÁN PARA EL PROCESO DE SOLDADURA DE ÁLABES DE TURBINAS DESGASTADAS TIPO PELTON A PARTIR DE MODELOS 3D GENERADOS MEDIANTE ESCÁNER".

El presente trabajo ha representado un aporte significativo para CELEC EP, ya que contribuye al desarrollo de soluciones técnicas que apoyan los procesos de mantenimiento, recuperación y mejora, así como al fortalecimiento de proyectos futuros.

Para constancia de lo expuesto, se suscribe la presente.

Atentamente,

Firmado electrónicamente por:
CARLOS ANDRES
GUERRA VASQUEZ
Validar únicamente con FirmaEC

Ing. Carlos Andrés Guerra Vásquez

C.C: 1714216635

Jefe de Ingeniería de Recuperación, Diseño y Optimización – CIRT

INDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

El presente trabajo de titulación desarrolla una metodología integral para la programación de un brazo robotizado KUKA, con el objetivo de realizar procesos de soldadura en turbinas tipo Pelton que presentan desgaste por abrasión. Para ello, se implementa un levantamiento geométrico de la turbina mediante escaneo 3D, lo que permite obtener un modelo preciso y detallado de la superficie a reparar. A partir de esta geometría digital, se generan trayectorias óptimas de soldadura utilizando el entorno de programación paramétrica Grasshopper en conjunto con el complemento KUKA|prc, asegurando un control preciso sobre los movimientos del robot y la calidad del cordón de soldadura. Además, se integra el paquete KUKA.ArcTech para la gestión de los parámetros del arco eléctrico y la sincronización de la activación y desactivación del proceso de soldadura MIG. La metodología propuesta busca optimizar los tiempos de reparación y garantizar una mayor precisión en aplicaciones de mantenimiento industrial, contribuyendo así a la extensión de la vida útil de componentes críticos en plantas hidroeléctricas.

Palabras clave: programación de robots, soldadura robotizada, KUKA, turbinas Pelton, escaneo 3D, KUKA|prc, KRL, ArcTech.

ABSTRACT

This degree work develops an integral methodology for the programming of a KUKA robotic arm, in order to perform welding processes in Pelton turbines that present abrasion wear. For this purpose, a geometric survey of the turbine is implemented by means of 3D scanning, which allows obtaining an accurate and detailed model of the surface to be repaired. From this digital geometry, optimal welding trajectories are generated using the Grasshopper parametric programming environment in conjunction with the KUKA|prc add-on, ensuring precise control over the robot's movements and the quality of the weld bead. In addition, the KUKA.ArcTech package is integrated for the management of the electric arc parameters and the synchronization of the activation and deactivation of the MIG welding process. The proposed methodology seeks to optimize repair times and ensure greater accuracy in industrial maintenance applications, thus contributing to the extension of the lifetime of critical components in hydroelectric power plants.

Keywords: robot programming, robotic welding, KUKA, Pelton turbines, 3D scanning, KUKA|prc, KRL, ArcTech.

PLANTEAMIENTO DEL PROBLEMA

Las turbinas tipo Pelton utilizadas en la generación hidroeléctrica están expuestas a condiciones operativas extremas que generan un desgaste significativo por abrasión en sus álabes. Para restaurar su funcionalidad y prolongar su vida útil, es necesario aplicar procesos de soldadura que restituyan el material perdido. Sin embargo, la soldadura manual presenta imprecisiones y variabilidad en los resultados, lo que afecta la eficiencia y calidad del proceso de recuperación [1].

El uso de brazos robóticos en soldadura industrial ha demostrado ser una alternativa eficaz para mejorar la precisión y repetibilidad del proceso. No obstante, la programación de estos robots para trabajar sobre geometrías complejas, como las turbinas tipo Pelton, representa un desafío técnico. La generación de trayectorias de soldadura precisas basadas en la geometría real de la turbina desgastada es fundamental para garantizar una soldadura automatizada eficiente.[2]

Actualmente, uno de los principales problemas es la ausencia de una metodología estandarizada que permita transformar los datos obtenidos mediante escaneo 3D en trayectorias óptimas para la programación de un brazo robótico KUKA. La falta de integración entre la captura de la geometría, el procesamiento de los datos en un software adecuado y la generación de trayectorias de soldadura MIG genera ineficiencias en la programación y ejecución del proceso [3].

Además, la soldadura manual no solo implica desafíos técnicos, sino que también representa riesgos significativos para la salud de los operarios. La exposición prolongada a humos metálicos, radiación ultravioleta, calor extremo y chispas puede ocasionar afecciones respiratorias, daños oculares y quemaduras en la piel. Estos riesgos incrementan la necesidad de adoptar procesos automatizados que minimicen la intervención humana y mejoren las condiciones de seguridad en el ambiente laboral.[4]

Asimismo, la selección del software adecuado es un factor crítico, ya que este debe permitir la conversión eficiente del modelo 3D en trayectorias funcionales para el robot. La ausencia de un procedimiento claro puede ocasionar errores en la precisión de la soldadura, tiempos de programación elevados y dificultades en la ejecución del proceso automatizado. Por lo tanto, es necesario desarrollar una metodología que combine el levantamiento de la geometría mediante escaneo 3D, la planificación de trayectorias en un software adecuado y la programación del brazo robótico para la soldadura de turbinas tipo Pelton. Esta

metodología permitirá optimizar el tiempo de programación y ejecución, asegurando una mayor precisión y eficiencia en el proceso de recuperación de estas turbinas, al mismo tiempo que reduce los riesgos ocupacionales asociados a la soldadura manual [5]

JUSTIFICACION

La automatización del proceso de soldadura en turbinas tipo Pelton es un avance significativo en la industria hidroeléctrica, debido a la creciente necesidad de mejorar la eficiencia, precisión y seguridad en los procesos de reparación. La integración de tecnologías avanzadas, como el uso de brazos robóticos KUKA y la planificación de trayectorias mediante modelos 3D, ofrece una solución innovadora a los desafíos asociados a la soldadura manual. Este enfoque no solo optimiza el tiempo y la calidad del trabajo, sino que también reduce los riesgos laborales inherentes al proceso tradicional de soldadura [5].

Una de las principales ventajas de la soldadura robotizada, es la mejora en la seguridad industrial, ya que minimiza la exposición del operario a peligros como: humos tóxicos, radiación ultravioleta y temperaturas extremas, reduciendo el riesgo de enfermedades respiratorias, daños en la piel y afecciones oculares. La implementación de un sistema automatizado para la reparación de turbinas no solo protege la salud de los trabajadores, sino que también contribuye a un entorno de trabajo más seguro y eficiente [4].

Adicionalmente, la automatización mediante robots permite una mayor precisión y repetibilidad en la aplicación de la soldadura, lo que asegura una distribución homogénea del material de aporte y optimiza el uso de recursos. Esto es crucial en la reparación de componentes críticos como las turbinas, donde la calidad y la fiabilidad de la soldadura tienen un impacto directo en el rendimiento de la unidad y en la vida útil del equipo [6].

El desarrollo de una metodología eficiente para la programación de los brazos robóticos KUKA, a partir de modelos 3D, generados por escaneo también representa un avance en el ámbito de la ingeniería, facilitando la implementación de soluciones tecnológicas de vanguardia en el sector hidroeléctrico. Esto no solo optimiza el proceso de reparación, sino que también reduce la dependencia de técnicas manuales, promoviendo una mayor eficiencia operativa y reduciendo los costos asociados a los tiempos de ejecución[7].

Por lo tanto, este estudio no solo contribuye al avance tecnológico en la automatización de procesos industriales, sino que también tiene un impacto positivo en la seguridad, salud ocupacional y sostenibilidad del sector hidroeléctrico. Al optimizar los procesos de reparación de turbinas, se garantiza una mayor precisión, calidad y eficiencia, fortaleciendo la competitividad y sostenibilidad de las plantas hidroeléctricas.

OBJETIVOS

Objetivo General

Desarrollar un planificador de trayectoria para un brazo robot en el proceso de soldadura de álabes de turbinas desgastadas tipo Pelton a partir de modelos 3D generados mediante un escáner.

Objetivos Específicos

- Investigar las estrategias de comunicación entre el escáner 3D, el brazo robot KUKA y el entorno de programación para una correcta integración de componentes a través de la revisión de los manuales técnicos de hardware y software.
- Implementar un planificador de trayectoria para el brazo robot KUKA, enfocado en la recuperación del álabe de la turbina desgastada por abrasión mediante el análisis mecánico de la soldadura a ser empleada a través de software de ingeniería.
- Realizar las pruebas de funcionamiento del sistema integrado para la validación de los resultados mediante pruebas de campo.

CAPÍTULO I

MARCO TEÓRICO

1.1. Centrales Hidroeléctricas

Las centrales hidroeléctricas son una fuente de generación de energía eléctrica renovable la cual ayuda a la reducción de emisiones de Gases de Efecto Invernadero (GEI) debido que su producción se basa en aprovechar la energía del agua ya sea en movimiento (energía cinética) o en almacenamiento en represas (energía potencial) para poder mover turbinas las cuales gracias a su movimiento se produce la energía eléctrica[8]. Básicamente el proceso se basa en la transformación de energía mecánica en energía eléctrica. Todo esto se logra gracias a la caída de agua a gran presión que chocan con los álabes de las turbinas y así generan un movimiento el cual es aprovechado por los rotores de transformadores que ayudan a transformar el movimiento en energía eléctrica[9].

1.2. Turbinas Hidráulicas

Las turbinas hidráulicas son dispositivos utilizados en las centrales de generación hidroeléctrica las cuales transfieren la energía del agua ya sea energía cinética, potencial o ambas para convertirla en energía mecánica haciendo girar un eje el cual forma parte crucial al producir energía eléctrica ya que este está conectado a los rotores del generador en el cual se transforma a energía eléctrica [10]. Las turbinas giran o dan vueltas como respuesta a la inyección de agua en sus palas. Es por eso por lo que las turbinas son esenciales en el ámbito de generación de energía a partir del agua[11].

Las turbinas se denotan desde las civilizaciones griegas y romanas ya que estos las usaban para moler granos. Pero estas no tenían la forma que se las conoce hoy en día eran más parecida como una rueda, pero ya eran usadas y útiles para aprovechar la energía del agua. Pero no fue hasta el siglo XIX que se volvió a tomar en cuenta esta manera de usar la energía del agua. Para ser más específicos hasta el 1827 que el ingeniero francés Benoî Fourneyron que invento un nuevo diseño de turbinas con el que se buscaba aprovechar la fuerza del agua para generar energía eléctrica.

Por lo general la construcción de las turbinas son las mismas. Una hilera de álabes o cangilones los cuales se acoplan a un eje. El agua impacta en los alabes haciendo girar el eje interior y así creando el ciclo de transferencia de energía. Existen varios tipos de ellas dependiendo del entorno donde va a ser aplicada[12].

1.3. Tipos de Turbinas hidráulicas

Existen diferentes tipos de turbinas, pero todas manejan el mismo principio. Estas se clasifican dependiendo de la aplicación[13]. Es decir, se clasifican de acuerdo con la presión que manejan y de la cantidad de flujo de agua que van a ser expuestas.

1.3.1. Clasificación por caudal

Las turbinas hidráulicas se pueden clasificar a partir de como fluye el agua a través de estas[14]. Ya que cuando pasa por la turbina el agua puede tomar distintos caminos y esto da lugar a tres categorías:

1.3.1.1. Flujo radial

El agua fluye por la turbina perpendicularmente al eje de rotación. Un ejemplo de las turbinas de flujo radial son las turbinas Francis.

1.3.1.1.1. Turbinas Francis

Fue desarrollada por el ingeniero británico James Bicheno Francis en 1849. En esta turbina tiene un rodete con álabes fijos. El agua entra justo por encima del rodete y a su alrededor para después caer a través de él y así poder girar los álabes.

El flujo de agua entra en dirección radial a la turbina, es decir que el agua fluye en dirección perpendicular al eje de rotación y sale de ahí en dirección axial paralela al eje[15]. Con este tipo de turbinas se puede realizar una conversión eficaz de la energía, esto gracias a que la turbina aprovecha eficientemente la energía cinética del flujo de agua.

Las turbinas Francis suelen ser aplicadas para alturas entre 2 a 800 metros. Las turbinas más grandes suelen tener una potencia de 0.25 a 800 MW y estas pueden trabajar bien tanto en orientación horizontal como vertical[16].

1.3.1.2. Flujo axial

El agua fluye de manera paralela al eje de rotación de la turbina. Esta configuración es usada por la turbina Kaplan.

1.3.1.2.1. Turbina Kaplan

Fue desarrollada por el profesor austriaco Viktor Kaplan 1913 el cual tomo como inspiración la turbina Francis. Es una turbina de hélice la cual tiene álabes orientables la cual es parecida a la turbina de un barco. En esta turbina el agua entra en dirección paralela al eje de rotación y fluye por los álabes de la turbina en la misma dirección axial. Ya que consta con la configuración de flujo axial permite una conversión de energía eficiente, esto se debe a que la turbina capta eficazmente la energía cinética del flujo de agua. Las turbinas Kaplan como las

turbinas de flujo axial son especialmente adecuadas para aplicaciones con poca altura y gran caudal.

Este tipo de turbinas se utilizan en saltos de altura de agua de hasta 50m y con caudales medianos y grandes los cuales suelen superar los 15 metros cúbicos por segundo.

1.3.1.3. Flujo Tangencial

El agua impacta de manera tangencial al eje de rotación. Una turbina con flujo tangencial es la turbina tipo Pelton.

1.3.1.3.1. Turbina Pelton

Creada por el inventor americano Lester Allan Pelton en 1879. El agua golpea a los cangilones o alabes de manera tangencial a su trayectoria de rotación. El chorro es direccionado cuidadosamente para maximizar su componente de velocidad tangencial, lo que permite una conversión de la energía eficiente a medida que giran los álabes de turbina.

Las turbinas Pelton son ideales para saltos de entre 80 a 1000 metros (saltos grandes) pero con pequeños caudales. La ventaja de este tipo de turbinas es que cada uno de los inyectores puede ser controlado individualmente. Es por eso que las fluctuaciones de caudal no son un problema ya que estas pueden ser controladas de manera eficiente y fácil.

1.3.2. Clasificación por presión

Las turbinas hidráulicas pueden clasificarse en turbinas de impulso y turbinas de reacción en función de la conversión de la energía hidráulica y de acuerdo con la presión que experimenta el fluido[17].

1.3.2.1. Turbina de impulso

En una turbina de impulso aprovecha únicamente la velocidad del fujo del agua para hacer girar el rodete[18]. Una corriente de agua golpea cada cubo del rodete. Al no haber succión en la parte inferior de la turbina, el agua sale por la parte inferior de la carcasa de la turbina después de golpear el rodete. Una turbina de impulsión suele ser adecuada para aplicaciones de gran altura y bajo caudal[19].

1.3.2.2. Turbina de reacción

Los álabes (rodetes) están totalmente sumergidos en agua y encerrados en una carcasa a presión. Los álabes están inclinados de modo que las diferencias de presión que los atraviesan crean fuerzas de sustentación, como las de las alas de los aviones, y las fuerzas de sustentación hacen girar el rodete. Las turbinas de reacción son las más comunes en aplicaciones de baja altura.

1.4. Tipos de desgastes en rodetes

Las turbinas tipo Pelton sufren a medida de desgastes los cuales deben ser corregidos cada cierto tiempo para que las centrales hidroeléctricas no sufran fallas más graves. Pero para poder arreglar una falla primero se debe conocer de que es lo que se trata[20]. Las turbinas están sometidos a diferentes tipos de desgastes lo cuales son mencionados a continuación:

1.4.1. Desgaste por Erosión

Este tipo de desgaste se origina debido que el agua suele llevar consigo partículas sólidas ya sean piedras, arena o polvo. Estos al momento de chocar fuertemente contra los cangilones produce que erosione las superficies lisas produciendo un desgaste las cuales generan irregularidades y estas distorsionan el flujo del agua generando turbulencias y causando también desgastes por cavitación de una manera acelerada.

1.4.2. Desgaste por Abrasión

Al igual que el desgaste por erosión se debe a la problemática de sedimentos sólidos transportados por el agua[21]. A diferencia de la erosión este degaste se genera principalmente por la fricción producida entre el agua con sedimentos contra la superficie del material. Es por esto que al momento de suceder esta fricción produce el desprendimiento de material en la superficie de los álabes lo cual generan irregularidades que son causantes de distorsión y turbulencias en el chorro de agua.

1.4.3. Desgaste por Cavitación

Este tipo de desgaste se debe a la presencia de flujos turbulentos que producen burbujas de aire o bolsas de vapor, los cuales al chocar entre sí de manera rápida y continua producen presiones del orden de 30000 a 50000 psi[22]. Esto se origina debido a una distribución no adecuada del chorro, a velocidades inadecuadas de disparo, pero también se puede deber a un diseño incorrecto de perfil de la cuchara por fabricación o por desgaste que hace que esta geometría cambie. Todo esto genera que el material sufra un desgaste por arranque debido al choque de burbujas de aire presentes en el chorro que da como producto irregularidades en la superficie del cangilón[23].

1.4.4. Desgaste por Corrosión

El desgaste corrosivo genera el desprendimiento de pequeñas partículas del material debido que agentes químicos o electroquímicos reaccionan con la superficie de las turbinas[24]. Estos pueden ser: oxigeno, el agua, sustancias químicas presentes en el agua o sales sueltas presentes en la misma. Este fenómeno suele presentarse más a menudo al detener la producción por dicha turbina y al no estar en funcionamiento suelen presentarse los agentes corrosivos causantes de

este tipo de desgaste lo cual forma irregularidades en la cuchara generado así turbulencias las cuales ayudan a la aparición de otros tipos de desgastes como puede ser la cavitación.

1.5. Soldadura

Soldadura es el proceso de unir dos o más materiales (comúnmente metales) mediante la aplicación de calor, presión o ambas y en ciertos casos un material aportante. Esta técnica se diferencia a las demás técnicas para unir materiales ya que la soldadura genera una unión permanente de las piezas[25]. Dicha técnica se la descubrió cuando se intentaba dar forma al hierro para poder utilizarlo como alguna herramienta. Pero al calentarlo y golpearlo con otro pedazo de hierro se logró unir y así sería el inicio de una de las técnicas más usadas en todas las industrias. Y con el tiempo se desarrollarían diferentes técnicas de soldaduras las cuales cuentan con diferentes propiedades que ayudan para distintas aplicaciones.

1.6. Tipos de soldadura

No fue hasta los finales del siglo XIX cuando se aparecieron las técnicas de soldadura moderna ya que se necesitaba mejorar los procesos industriales y producción a gran escala debido al comienzo de la Primera Guerra Mundial. Llevar a cabo esta producción solo fue posible gracias a la aparición de distintas técnicas de soldadura los cuales han sido mejorados y siguen siendo usados hasta la actualidad los cuales son presentados a continuación.

1.6.1. SMAW

Al hablar de SMAW se referiere a la soldadura por arco con electrodo revestido o por sus siglas en inglés Shielded Metal Arc Welding (SMAW) esta técnica se introdujo por primera vez en 1888 por el inventor ruso Nikolay Gavrilovich. Esta técnica se basa en la fusión de las partes gracias al calor producido por el arco eléctrico que se crea entre el electrodo combustible y el metal base que se desea unir[26]. Esta técnica es la más utilizadas en la actualidad gracias a su gran versatilidad, puede ser usada utilizada en cualquier industria y tiene grandes propiedades.

La soldadura SMAW un proceso manual que utiliza un electrodo consumible recubierto de fundente para crear una unión metálica. Al establecer un arco eléctrico (entre 3,000-3,500°C) entre el electrodo y la pieza de trabajo, se funden tanto el núcleo metálico del electrodo como el material base, mientras que el revestimiento genera gases protectores y una capa de escoria que protege el baño de fusión de la oxidación. Este método requiere una fuente de poder, una pinza portaelectrodos y una pinza de tierra para completar el circuito eléctrico [27]. Su principal ventaja radica en su versatilidad: puede usarse en exteriores, con diversos metales (aceros al carbono, inoxidables e incluso hierro fundido) y sin necesidad de equipos complejos. Aunque requiere destreza del operador y presenta limitaciones como salpicaduras y necesidad de

remover escoria, sigue siendo uno de los procesos más económicos y accesibles para trabajos de unión metálica en campo y talleres.

1.6.2. GMAW

La soldadura Gas Metal Arc Welding (GMAW) o en español se traduce como soldadura por arco metálico con gas. El proceso utiliza un electrodo continuo consumible (alambre de soldadura) y un gas de protección para la producir el arco deseado. La soldadura GMAW es una de las más utilizadas en las industrias debido que su proceso es muy rápido y continuo a diferencia de los otros[28]. Este proceso se subdivide en dos ramas dependiendo a los metales en los cuales se van a soldad y a su vez a las propiedades que se desea tener en el cordón de soldadura.

Metal Inert Gas (MIG).- El proceso es el mismo con la diferencia que utiliza un gas diferente lo cual cambia sus propiedades. En el caso de MIG se trata de un gas inerte lo cual permite más penetración de la soldadura.

Metal Active Gas (MAG).- A diferencia del anterior proceso este utiliza gas activo.

1.6.3. FCAW

Se refiere a la Soldadura por Arco con Núcleo Fundente o Fluxed Cored Arc Welding (FCAW)[29]. Este tipo es un proceso que utiliza un electrodo tubular continuo relleno de fundente. A diferencia de la soldadura GMAW, puede operar con o sin gas protector adicional, lo que permite clasificarla en dos variantes principales.

- Gas-Shielded **(FCAW-G)** emplea un gas protector externo (CO_2 o mezclas de argón/CO_2) que complementa la acción del fundente del electrodo, proporcionando un arco más estable, menor porosidad y mejor acabado superficial.
- Self-Shielded (**FCAW-S**) no requiere gas externo, ya que el fundente del electrodo genera su propia protección mediante gases y escoria durante la combustión. Esta característica lo hace ideal para trabajos en exteriores con condiciones adversas (viento, humedad), aunque presenta mayores salpicaduras y requiere remoción de escoria posterior.

1.7. Brazo Robótico Soldador

Los brazos robóticos soldadores son sistemas automatizados diseñados para realizar operaciones de soldadura con alta precisión, repetibilidad y eficiencia en entornos industriales. Equipados con herramientas de soldadura (como MIG/MAG, TIG o por puntos) y controlados

por programas avanzados, estos robots optimizan la calidad de las uniones, reducen desperdicios y minimizan riesgos para los operarios.

1.7.1. Brazo Robótico

Los brazos robóticos, también denominados brazos industriales, son dispositivos mecánicos programables diseñados para optimizar los procesos de producción industrial. Su implementación no solo incrementa significativamente la productividad al ejecutar tareas con mayor rapidez y precisión, sino que también mejora las condiciones de seguridad al reducir la exposición de los trabajadores a entornos peligrosos.

Estos sistemas automatizados están compuestos por una estructura articulada cuyos movimientos son controlados mediante motores estratégicamente ubicados en sus juntas. Su capacidad operativa varía según diferentes parámetros técnicos, incluyendo el número de grados de libertad, la configuración de sus eslabones y el tipo de efector final implementado[30]. Estas características determinan su funcionalidad específica, permitiendo su adaptación a diversas aplicaciones industriales, desde ensamblaje y soldadura hasta manipulación de materiales y operaciones de pintura.

1.7.2. Grados de libertad

Los grados de libertar (GDL) son el número de movimientos independientes que puede realizar un brazo robótico ya sean rotacional o lineal. Existe un grado de libertad al observar cada eje geométrico alrededor del cual puede girar o extenderse una articulación[31]. Es decir, según el número de grados de libertad tenga un brazo robótico más flexible y hábil será.

1.7.3. Controladores

Los controladores de brazos robóticos son sistemas computarizados que actúan como el "cerebro" del robot, encargándose de procesar las instrucciones de programación y convertirlas en movimientos precisos. Estos sistemas integran hardware especializado (como PLCs o unidades de control dedicadas) con software avanzado para gestionar múltiples ejes de movimiento simultáneamente. Utilizan algoritmos de cinemática inversa para calcular las posiciones exactas de cada articulación y garantizar que el efector final alcance su objetivo con precisión milimétrica. Los controladores modernos incorporan sensores de retroalimentación (encoders, células de carga) que permiten ajustes en tiempo real, compensando variables como vibraciones o desgaste mecánico. Para aplicaciones complejas como soldadura o montaje, pueden incluir funciones avanzadas como seguimiento de trayectorias adaptativas, control de fuerza o integración con sistemas de visión artificial. La comunicación se realiza mediante protocolos industriales como EtherCAT o PROFINET, asegurando sincronización perfecta con

otros equipos en líneas de producción automatizadas. Utilizan algoritmos de cinemática inversa para calcular ángulos y trayectorias, mientras procesan datos de encoders y sensores de fuerza/torque en tiempo real. Los más avanzados integran PLC industriales o sistemas basados en Robot Operating System (ROS), permitiendo adaptarse a tareas complejas como soldadura o ensamblaje con tolerancias de ±0.01 mm [32].

1.7.4. Soldadura robotizada

La soldadura robotizada es la aplicación de brazos robóticos industriales para realizar procesos de unión metálica con alta precisión, repetibilidad y eficiencia[33]. Estos sistemas combinan robots articulados (generalmente de 6 ejes) con equipos de soldadura especializados (MIG/MAG, TIG, láser o por puntos), controlados por software avanzado que permite programar trayectorias, parámetros de corriente y patrones de movimiento. A diferencia de la soldadura manual, ofrece ventajas clave como mayor velocidad (hasta 3 veces más rápida), calidad consistente (con tolerancias de ±0.1 mm) y reducción de defectos (porosidad, falta de penetración)[34]. Se emplea en sectores como la automotriz (chasis, carrocerías), aeroespacial (componentes estructurales) y fabricación pesada (tuberías, tanques), donde se integra con sistemas de visión artificial y sensores láser para correcciones en tiempo real.

1.7.5. KRL

Kuka Robot Lenguaje es el lenguaje principal que usan los robots KUKA para ejecutar movimientos, controlar el flujo del programa, gestionar señales, etc. Se debe tener en cuenta al elegir un software el cual conste con un post-procesador el cual genere un código KRL[35]

1.7.6. ArcTech

Es el paquete de software complementario de KRL el cual se especializa en la soldadura. Es decir, es un paquete complementario netamente de robots KUKA soldadores. En el cual ayuda a controlar el ON/OFF del arco de soldadura y a su vez generar parámetros de soldadura como el avance, velocidad del alambre, amperaje y ángulo de ataque de la antorcha[36].

1.8. Ingeniería inversa

La ingeniería inversa es una metodología técnica que consiste en analizar un producto o componente existente para comprender su diseño, funcionamiento y características, con el fin de replicarlo, mejorarlo o adaptarlo a nuevas necesidades. A diferencia del proceso de ingeniería convencional, que parte de un diseño conceptual hacia la fabricación, la ingeniería inversa sigue el camino opuesto: desarma y examina un objeto ya fabricado para extraer su

conocimiento técnico y generar documentación detallada, como planos o especificaciones de materiales[37].

Esta técnica es especialmente útil en contextos industriales donde se requiere reproducir piezas que carecen de planos originales, ya sea por obsolescencia o porque la documentación se ha perdido. Un ejemplo claro es su aplicación en el Centro de Investigación y Recuperación de Turbinas (CIRT), donde se emplea para reconstruir componentes hidroeléctricos deteriorados, garantizando su funcionalidad sin depender de fabricantes externos. Además, la ingeniería inversa permite optimizar diseños existentes, identificando debilidades en materiales o geometrías para aumentar su eficiencia o durabilidad. En el sector energético, esto se traduce en equipos más confiables y con mayor vida útil, reduciendo costos de mantenimiento y tiempos de inactividad.

Otro aspecto relevante es su papel en la independencia tecnológica. Al aplicar ingeniería inversa, empresas e instituciones pueden dejar de depender de proveedores extranjeros para la reparación o fabricación de piezas críticas, fomentando la industria local[38]. Esto no solo genera ahorros económicos, sino que también fortalece las capacidades técnicas nacionales. En resumen, la ingeniería inversa es una herramienta clave para la innovación, la sostenibilidad operativa y el desarrollo industrial, permitiendo la recuperación, mejora y adaptación de tecnologías existentes a nuevas demandas del mercado.

1.9. CIRT

El Centro de Investigación y Recuperación de Turbinas (CIRT), gestionado por CELEC EP HIDROAGOYÁN, cumple un rol estratégico al ofrecer servicios especializados tanto al sector público como al privado, consolidándose como un referente en la recuperación y mantenimiento de componentes electromecánicos para centrales hidroeléctricas[39]. Su operación ha demostrado ser eficiente gracias a los resultados obtenidos en su taller, respaldados por la capacidad técnica de profesionales y mano de obra ecuatoriana. Esto ha permitido establecer alianzas con empresas privadas, generando ingresos significativos que superan los USD 300.000, lo que refleja su impacto económico.

Entre los servicios destacados se incluyen procesos de rectificado, pulido, balanceamiento estático y tratamiento térmico de piezas como obturadores y rodetes Pelton,). Además, el CIRT ha extendido su alcance a centrales hidroeléctricas privadas en Tungurahua, específicamente en Baños, donde brinda mantenimiento *in situ*, reparación de componentes, asistencia en montaje y verificación de estándares para garantizar una puesta en marcha eficiente.

CIRT impulsa proyectos de ingeniería inversa y convencional para desarrollar planos detallados, estudios de materiales (como espectrometrías) y diseño de componentes, con el objetivo de convertirse en proveedor directo de repuestos para el sector hidroeléctrico[40]. Esta iniciativa no solo reduce la dependencia tecnológica del exterior, sino que también optimiza costos y tiempos de reparación, fortaleciendo la industria nacional.

Históricamente, el centro ha contribuido a disminuir la externalización de reparaciones de equipos hidroeléctricos, que antes se realizaban en el extranjero, generando ahorros y reduciendo tiempos de inactividad. Actualmente, su apoyo se extiende al sector privado mediante evaluaciones técnicas previas que determinan la viabilidad de las intervenciones, asegurando procesos eficaces.

CAPÍTULO II

ANÁLISIS DE ALTERNATIVAS

En este capítulo se realiza un análisis comparativo de distintas alternativas tecnológicas y metodológicas para la programación de un brazo robótico KUKA para la aplicación de soldadura en turbinas tipo Pelton. El objetivo es seleccionar la solución más adecuada considerando criterios técnicos, económicos y de implementación.

2.1. ALTERNATIVAS PARA LA GENERACIÓN DE TRAYECTORIAS

Una vez comprendido el lenguaje que se debe manejar se pudo analizar las distintas alternativas que se tiene para la generación de trayectorias para el brazo KUKA KR30L16. Se seleccionó la alternativa más adecuada considerando criterios técnicos y operativos, tales como costo, precisión, adaptabilidad.

2.1.1. RobotDK

Es un software de simulación y programación offline para robots industriales. Permite crear, simular y validar trayectorias de robots sin necesidad de tener el robot físico, y luego exportar el código en el lenguaje específico del robot (como KRL para KUKA, RAPID para ABB, o JBI para Yaskawa).

RoboDK fue diseñado para robots industriales, en el cual permite planificar, simular y generar trayectorias sin necesidad de tener acceso físico al robot durante las etapas iniciales del proyecto[41]. Gracias a su entorno tridimensional, el usuario puede construir celdas completas, importar modelos CAD y ensamblar robots, herramientas y piezas con precisión. Una de sus principales fortalezas radica en su capacidad para generar código específico para distintos fabricantes, como KUKA, ABB o FANUC, mediante post-procesadores integrados que traducen las trayectorias simuladas en instrucciones directamente ejecutables por el robot.

Para aplicaciones como la programación de un brazo robotizado en procesos de soldadura sobre turbinas tipo Pelton, RoboDK representa una solución eficiente, ya que permite importar la geometría obtenida a través de escaneado 3D, establecer trayectorias ajustadas al perfil real del componente y analizar con antelación la viabilidad del movimiento del robot en cuanto a colisiones, accesibilidad y orientación de la herramienta. Además, su integración con lenguajes como Python facilita la automatización de tareas repetitivas o la personalización de trayectorias complejas[42]. Aunque no está específicamente diseñado para soldadura por arco, su

versatilidad permite configurar herramientas, velocidades, encendidos y apagados de procesos de forma adaptable. En resumen, RoboDK es una alternativa robusta y flexible para validar trayectorias robóticas de forma segura y eficiente antes de ejecutar cualquier operación en el entorno real.

Como se menciona sirve para importar geometrías, curvas y líneas las cuales ayudan o en ciertos casos son la trayectoria como tal. Esto puede ser importado gracias a que RoboDK puede agregarse como plug-in a la mayoría de software de diseño como Solidworks, Rhinoceros, etc.

Se debe mencionar que las pruebas se las analizó con su versión gratuita.

2.1.2. Rhinoceros (Grasshopper)

Rhinoceros, comúnmente conocido como Rhino, es un software de modelado 3D basado en geometría NURBS, ampliamente utilizado en diseño industrial, arquitectura y manufactura avanzada[43]. Su principal fortaleza radica en la precisión con la que puede representar formas complejas, curvas libres y superficies orgánicas. Grasshopper, por su parte, es un complemento visual (plug-in) integrado a Rhino que permite la programación paramétrica mediante nodos, sin necesidad de escribir código tradicional. A través de esta interfaz visual, los usuarios pueden controlar geometrías, crear algoritmos de generación de formas y automatizar procesos de diseño de manera flexible y escalable[44].

En el contexto de aplicaciones robóticas, Rhinoceros con Grasshopper ofrece una plataforma potente para diseñar trayectorias basadas en geometría real, especialmente cuando se trata de piezas con superficies irregulares o formas personalizadas, como los cangilones de una turbina Pelton. La ventaja de trabajar con geometría paramétrica es que se puede ajustar dinámicamente la trayectoria del robot en función de cambios en la forma o en la estrategia de trabajo. Además, Grasshopper permite integrar plugins especializados para robótica, los cuales facilitan la creación de trayectorias en función de condiciones geométricas o restricciones físicas. Si bien este entorno no genera por sí solo código de control para robots, su capacidad de personalización y precisión geométrica lo convierten en una herramienta valiosa para la etapa de diseño y planificación de trayectorias en procesos industriales complejos. En resumen, Rhinoceros con Grasshopper constituye una solución versátil, visualmente intuitiva y especialmente poderosa cuando se requiere adaptar trayectorias a geometrías no estándar.

2.1.2.1. KUKA|prc

KUKA|prc (Parametric Robot Control) es un complemento desarrollado por la asociación alemana Robots in Architecture, diseñado para integrarse dentro del entorno de Grasshopper en Rhinoceros[45]. Este plugin permite programar y simular robots KUKA de forma visual y paramétrica, aprovechando la potencia de generación geométrica de Grasshopper. A diferencia de otros entornos tradicionales de programación robótica, KUKA|prc no requiere escribir código manualmente en KRL, sino que permite construir trayectorias, definir velocidades, establecer orientaciones de herramienta, configurar señales de entrada/salida y simular el comportamiento del robot directamente desde un entorno gráfico[46]. La herramienta genera automáticamente el código KRL que puede ser exportado y ejecutado en el controlador del robot físico.

En el marco de este proyecto de titulación, KUKA|prc representa una opción especialmente adecuada para el desarrollo de trayectorias sobre superficies complejas, como los cangilones de las turbinas tipo Pelton, ya que permite combinar el modelado preciso de Rhino con la lógica paramétrica de Grasshopper para generar movimientos altamente personalizados y adaptables. Además, al ofrecer una simulación visual del movimiento del robot y su cinemática, permite verificar anticipadamente aspectos como orientación del efector final, accesibilidad y detección de colisiones. Si bien requiere una curva de aprendizaje inicial para dominar el flujo de trabajo paramétrico, KUKA|prc se destaca por ofrecer una integración fluida entre el diseño digital y la programación robótica avanzada, lo cual lo convierte en una herramienta poderosa para entornos de fabricación no convencionales o procesos que demandan alta adaptabilidad geométrica.

2.1.3. RobotMaster

Es un software de programación offline de robots industriales desarrollado originalmente por la empresa canadiense In-House Solutions. Surge como una solución CAM-robótica avanzada diseñada para integrar entornos CAD/CAM tradicionales con la programación de trayectorias robóticas complejas. Su enfoque central es permitir que las trayectorias generadas en software CAM, como Mastercam o SolidWorks, puedan ser adaptadas, optimizadas y trasladadas directamente a entornos de robótica industrial sin requerir una programación manual extensiva en el lenguaje nativo del robot [47].

Desde su concepción, RobotMaster ha buscado cerrar la brecha entre el mundo del diseño mecánico y la automatización robótica, ofreciendo una plataforma que traduce directamente

operaciones típicas de manufactura asistida por computadora —como fresado, taladrado, soldadura, desbaste o corte por láser— a trayectorias comprensibles y ejecutables por un robot industrial. El sistema permite simular en tiempo real todos los movimientos del robot, verificando colisiones, zonas inaccesibles y límites cinemáticos, con una representación visual clara y detallada [48]. Esto ha convertido a RobotMaster en una herramienta ampliamente adoptada en sectores como la industria aeroespacial, automotriz, naval y de moldes y matrices, donde la interacción precisa entre el diseño geométrico y la robótica es fundamental.

Una característica clave del software es su capacidad de generar código de robot personalizado a través de postprocesadores específicos para una amplia variedad de marcas industriales, como KUKA, ABB, FANUC, Yaskawa Motoman y Staubli, entre otras. Esta flexibilidad permite que las trayectorias CAM se adapten automáticamente a la cinemática y características específicas de cada robot, facilitando la adopción de robots como herramientas versátiles en procesos de manufactura avanzada [49]. Asimismo, RobotMaster ofrece algoritmos de optimización de orientación de herramienta, planificación de movimientos multi-eje y gestión inteligente de la cinemática redundante, funciones que lo distinguen de otros entornos más básicos de simulación robótica.

En resumen, RobotMaster representa una solución integral para la programación offline de robots industriales, al ofrecer una plataforma de alto nivel que combina el poder del diseño asistido por computadora con la lógica de control robótico. Su origen como puente entre el software CAM y la robótica lo convierte en una herramienta con un enfoque técnico sólido y una trayectoria consolidada en múltiples industrias que demandan automatización flexible y de alta precisión.

2.2. EVALUACIÓN DE ALTERNATIVAS

2.2.1. Criterios de evaluación entre software

Se evaluó los software teniendo en cuanta los siguientes parámetros para seleccionar el de mejor calificación. Esta evaluación se lo puede observar en la Tabla 1.

Tabla 1. *Criterios de Evaluación*

Ponderación	Características	RoboDK	Total	Rhinoceros	Total	RobotMaster	Total
20%	Costo de licencia	6	12%	8	16%	4	8%
20%	Adaptabilidad	5	10%	9	18%	3	6%
20%	Desarrollo de trayectorias sobre archivos .stl	2	4%	10	20%	0	0%
20%	Eficiencia del Post Procesador	9	18%	9	18%	9	18%
20%	Versión gratuita	3	6%	10	20%	1	2%
100%	Total		50%		92%		34%

GRAFICO 1

Análisis de alternativas mediante la comparación de costo de licencia teniendo en cuenta la suscripción de la licencia anual.

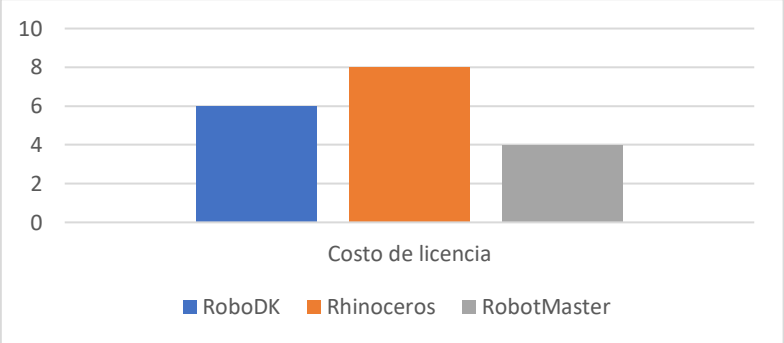


Figura. 1 *Análisis de costos*

GRAFICO 2

El análisis de adaptabilidad se observa en la figura 2 en la cual se buscó comparar la adaptabilidad del software con el entorno real.

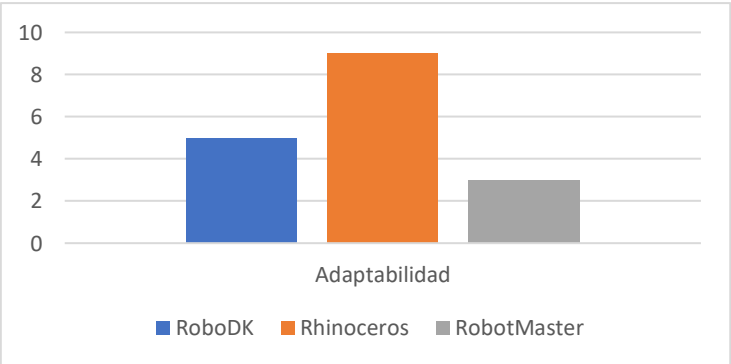


Figura. 2 *Análisis de Adaptabilidad*

GRAFICO 3

En la figura 3 se presenta la comparativa de softwares mediante el criterio de desarrollo de trayectorias sobre archivos .stl teniendo en cuenta que estos tipos de archivos son los que produce el escáner 3D.

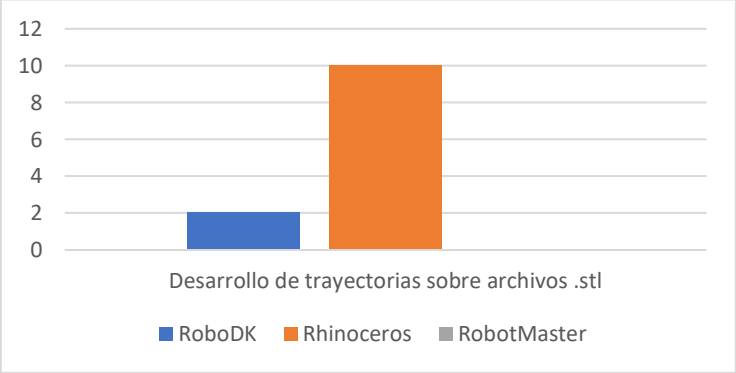


Figura. 3 *Desarrollo de trayectorias sobre archivos .stl*

GRAFICO 4

La eficiencia del Post Procesador se logra ver en la figura 4 en la cual se comparó que tan eficiente son los códigos KRL generados por los distintos software.

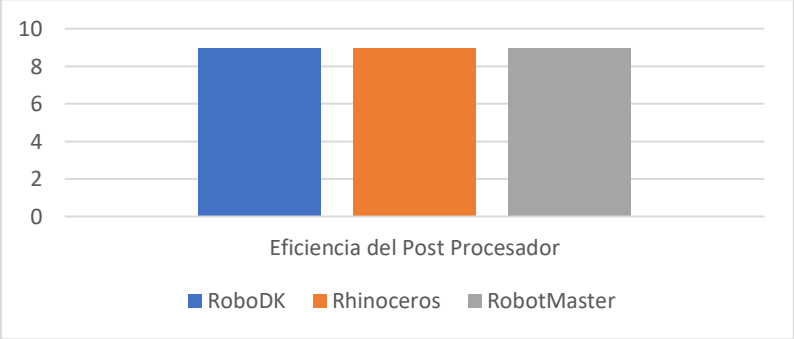


Figura. 4 *Eficiencia del Post Procesador*

GRAFICO 5

La versión gratuita se analizó en la figura 5 ya que las pruebas realizadas fueron generadas en las versiones de prueba o gratuitas que ofrecía cada software.

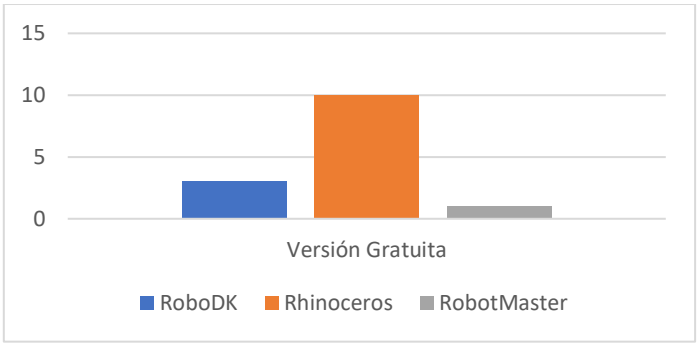


Figura. 5 *Versión gratuita*

Tras el análisis comparativo realizado entre las opciones de software evaluadas —RoboDK, Rhinoceros y Robotmaster— y considerando criterios como el costo de licencia, la adaptabilidad, la capacidad para desarrollar trayectorias directamente sobre archivos **.stl**, la eficiencia del post-procesador y la disponibilidad de versiones gratuitas o educativas, se concluye que Rhinoceros resulta ser la opción más adecuada para el desarrollo del presente proyecto.

Rhinoceros destaca no solo por su relación costo-beneficio favorable, sino también por su alta versatilidad en el manejo de geometrías complejas y su compatibilidad con múltiples plugins especializados, como Grasshopper y KUKA|prc, que permiten una programación avanzada y flexible del brazo robot. Esta capacidad de personalización y la facilidad para integrar procesos de diseño y simulación convierten a Rhinoceros en la herramienta ideal para la generación y control preciso de trayectorias, lo cual resulta fundamental para la aplicación de soldadura automatizada en componentes con superficies irregulares, como los álabes de turbinas.

CAPÍTULO III

DISEÑO Y SIMULACIÓN

Una vez seleccionado el software Rhinoceros por su plugin Grasshopper y su componente KUKA|prc el cual ofrece facilidad de manejo y una programación por bloques que permite programar de manera más visual e intuitiva.

Para la realización de pruebas se seleccionó una pieza metálica la cual se muestra en la figura 6. Esta simula la geometría de un cangilón tipo Pelton.

Figura. 6 *Pieza a soldar lateral*

3.1. ESCANEO 3D

Una vez fijada la pieza sobre la mesa giratoria para que se mantenga estática todo el tiempo se procedió a colocar targets los cuales ayudarán al escaneado 3D de la pieza. Los targets se posicionan con una separación de ±5cm aproximadamente cuando se necesita detalle y cuando es una superficie plana y que no es de tanto interes este valor tiende a aumentar. Se debe tener en cuenta que estos targets se deben formar triángulos ya que el escaner necesita de 3 targets para no perderse y trabajar de manera óptima. Al observar la figura 7 se aprecia como se colocaron los targets alrededor de la geometría.

Figura. 7 *Pieza a soldar con targets lateral*

Una vez colocado los targets alrededor de la pieza se prosigue a reconocer estos en el escáner 3D con la ayuda del programa VXElements.

Figura. 8 *Escaner 3D*

Una vez reconocido por el software los targets como puntos en el espacio se puede escanear para así lograr tener la malla a trabajar.

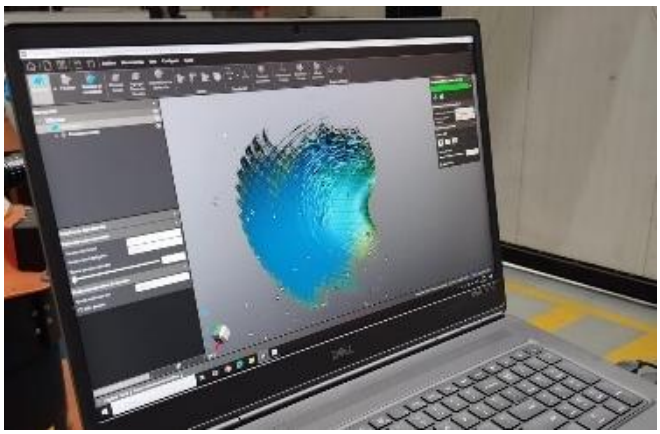


Figura. 9 *Creación de malla*

Una vez escaneada la pieza se procede a exportar la malla como un archivo .stl el cual servirá para tener la pieza a trabajar a disposición.

3.2. POST PROCESO DE LA MALLA ESCANEADA

Pero antes de ello se debe hacer un post procesado de la pieza escaneada para limpiar ruidos que se pudieron generar al momento de escanear y a su vez limpiar zonas extras que fueron escaneadas y que no son de interés. Como se observa en la figura 10 es la malla generada por el escáner sin post procesarle.

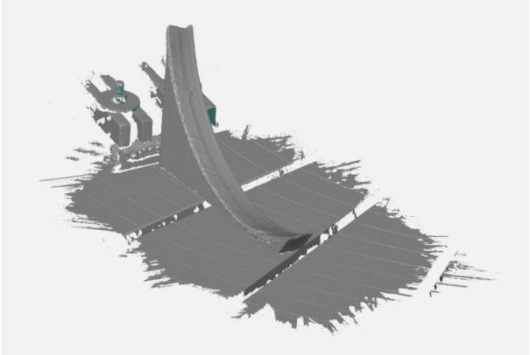


Figura. 10 *Pieza escaneada*

GOM Inspect es un software gratuito que permite limpiar mallas y centrarlas con un archivo nominal (archivo CAD) generado con las medidas de los planos de las piezas[50]. Así se logra limpiar y orientar la pieza. Al referirse orientar se habla de posicionar la pieza sobre un plano X-Y que en este caso se "asentó" la pieza sobre el plano X-Y la zona de la mesa giratoria ya que esta es la zona más plana y se pudo tomar como referencia ya que esta no tendría mucha desviación y será óptima para trabajar. En la figura 11 se aprecia la misma malla ya post procesada con la ayuda de software GOM Inspect.

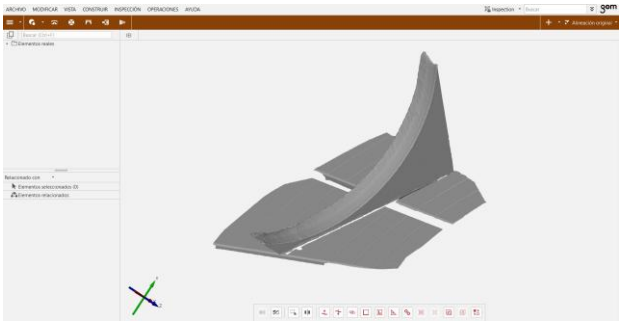


Figura. 11 *Pieza escaneada post procesada*

Una vez ya limpia y orientada la pieza es decir la zona plana centrada en un plano X-Y se la exportó la malla del software GOM Inspect como archivo. stl al programa Rhinoceros en el cual se lo trabajará para la generación de trayectorias.

3.3. PROGRAMACIÓN EN GRASSHOPPER

Una vez la pieza en Rhinoceros como se observa en la figura 12 se prosigue a trabajar con Grasshopper para la programación del brazo KUKA KR30-L16.

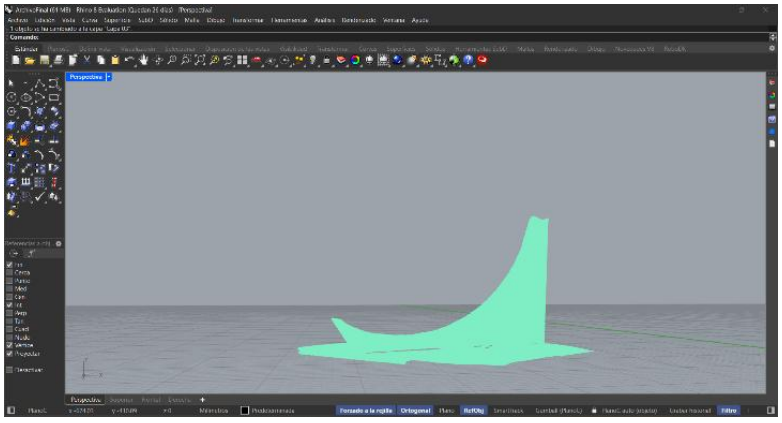


Figura. 12 *Pieza escaneada en Rhinoceros*

3.3.1. KUKA|prc

Este plug-in ayudará a la generación de trayectorias, pero primero se debe conocer de qué consta este componente para saber cómo utilizarlo.

Figura. 13 *KUKA|prc*

1. Core. Comandos de movimiento y bloque Core el cual es el núcleo de la programación.

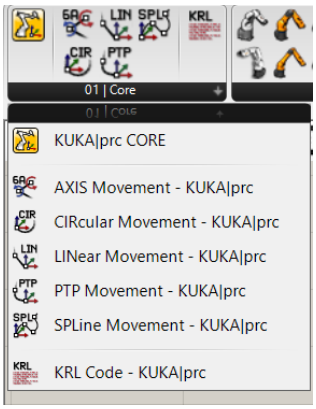


Figura. 14 *Core*

2. Virtual robot

Es donde se encuentra la librería de robots KUKA disponibles.

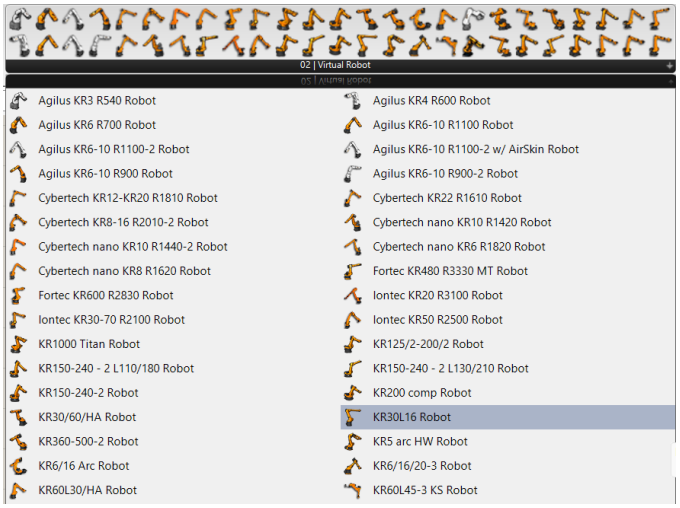


Figura. 15 *Virtual Robot*

3. Virtual Tool

Apartado donde consta todas la herramientas pre cargadas a su vez también consta de opciones para agregar herramientas al robot como también generar los distintos cambios de herramienta si el proceso lo amerita.

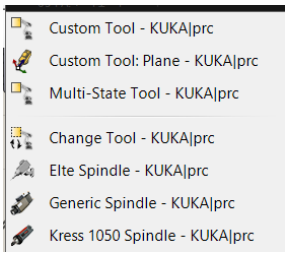


Figura. 16 *Virtual Tool*

4. Toolpath Utilities. Estas herramientas sirven para modificar o ajustar trayectorias que se conectan al input CMDS.

Figura. 17 *Toolpath Utilities*

5. Utilities. (Utilidades generales del sistema) Estas herramientas sirven para conectar lógicas, señales, comandos de espera, marcos, etc.

Figura. 18 *Utilities*

3.4. CREACIÓN DE ZONA DE TRABAJO

Como primer punto es colocar el brazo robótico en la zona de trabajo el cual es un brazo KUKA KR30-L16 el que se le encuentra en la librería de KUKA|prc. Para colocarlo se parte del bloque principal al momento de programar el cual es KUKA|prc Core. El cuál es el núcleo de la programación.

3.4.1 KUKA|prc Core

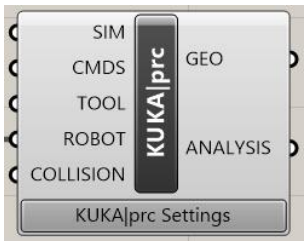


Figura. 19 *KUKA|prc Core*

Sus entradas:

SIM.(Simulation settings) El cual permite manipular el avance de la simulación de manera manual.

CMDS. (Commands) Es donde van conectados los comandos al robot es decir las ordenes que va a cumplir el robot como tipos de movimientos como son: Axial, Lineal, Circular, PTP y Spline. En resumen, aquí va conectado la trayectoria o secuencia de acciones que el robot debe realizar.

TOOL. (Tool settings) Es donde se conectan las herramientas que se acoplarán al robot. Es decir, aquí se define que herramienta será utilizada por el robot. También contiene información como posición, orientación de la herramienta, para ser claros, también contiene el TCP (Tool Center Point) de la herramienta el cual es importante en la realización de simulaciones precisas con el entorno real.

ROBOT. (Robot model) Es el robot que se define a usar en la simulación.

COLLISION. (Collision check settings) Se puede agregar superficies o mallas las cuales pueden ser obstáculos al momento de que realice los movimientos. Esto avisará que existe alguna colisión con dicha malla o superficie cargada.

Salidas:

GEO. (Geometry output) devuelve la geometría resultante de la simulación del robot. El cual es útil para visualizar el brazo en movimiento y su trayectoria.

ANALYSIS. (Analysis data) Muestra resultados analíticos como velocidad por punto, aceleración, errores cinemáticos, colisiones, etc. Sirve para validar si la trayectoria es factible y segura.

Dando clic en el botón inferior que dice KUKA|prc Settings se ingresa a un menú en el cual se pueden colocar distintos parámetros.

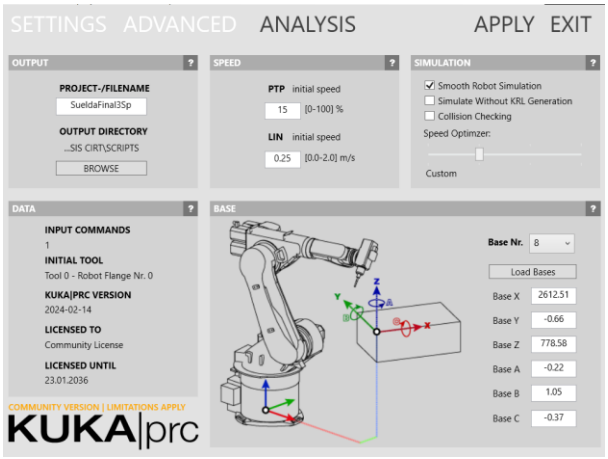


Figura. 20 *KUKA|prc Settings*

OUTPUT

- **PROJECT-/FILENAME.** Aquí se define el nombre del archivo .src que se generará para enviar al robot.
- **OUTPUT DIRECTORY:** Ruta donde se guardarán los archivos generados (SRC, DAT, TXT). Puedes cambiarla con el botón BROWSE.

SPEED

Aquí se define la velocidad inicial de los movimientos:

- **PTP (Point-to-Point):**Velocidad en movimientos de articulación. Rango: 0–100%.
- **LIN (Lineal):**Velocidad en movimientos lineales.Rango: 0.0 – 2.0 m/s.

SIMULATION

- **Smooth Robot Simulation** Habilita una animación fluida en la simulación de movimientos del robot en Rhino/Grasshopper.
- **Simulate Without KRL Generation** Solo simula, pero no genera código KRL. Ideal si solo se está probando trayectorias.
- **Collision Checking** (Solo en versión completa). Verifica colisiones entre robot, herramienta y entorno.
- **Speed Optimizer** Solo visible si se define varios comandos; ajusta cómo se suaviza la velocidad.

DATA

- **INPUT COMMANDS:** Cuántos comandos están siendo interpretados desde el bloque KUKA|prc CORE.
- **INITIAL TOOL:** El número de herramienta activa por defecto (ej. TOOL 0 es la brida).
- **KUKA|prc VERSION y LICENSE:** Información de tu versión y vigencia de la licencia.

BASE

- **Base Nr.** Es el número de base que estás configurando (en tu caso, Base 8).
- **Base X, Y, Z** La posición del origen de la base física respecto al robot.
- **Base A, B, C** Son los ángulos de rotación (en grados) en torno a los ejes Z, Y, X respectivamente. Esto define cómo está orientada la base (por ejemplo, si está inclinada, rotada, etc.).

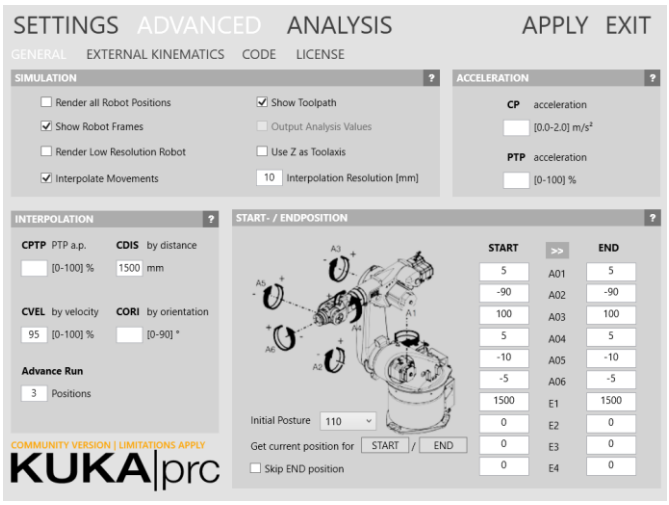


Figura. 21 *KUKA|prc Advanced*

SIMULATION

- **Render all Robot Positions** Muestra todas las posiciones del robot en la simulación (útil para ver toda la trayectoria).
- **Show Robot Frames** Muestra los ejes XYZ de cada punto de la trayectoria del TCP (útil para verificar orientación).
- **Render Low Resolution Robot** Muestra un modelo más simple del robot (para acelerar el rendimiento gráfico).
- **Interpolate Movements** Habilita el suavizado de movimientos entre puntos. Mejora visualización y trayectoria.
- **Interpolation Resolution [mm]** Define cada cuántos mm se interpolan los movimientos (por ejemplo, cada 10 mm se genera una posición intermedia).
- **Show Toolpath** Muestra una línea indicando por dónde se moverá la herramienta.
- **Output Analysis Values** Muestra datos de análisis como esfuerzo en motores o torque.
- **Use Z as Toolaxis** Orienta todos los planos generados de entrada asumiendo que el eje Z local es la dirección de la herramienta (muy útil para soldadura).

ACCELERATION

- **CP acceleration [0.0–2.0 m/s²]** Aceleración para movimientos lineales (LIN). Más aceleración → trayectorias más rápidas pero más bruscas.
- **PTP acceleration [0–100%]** Aceleración para movimientos PTP (Point to Point). Porcentaje del máximo permitido.

INTERPOLATION

- **CPTP (PTP Approximation)** Cuánto se aproxima el robot entre puntos PTP (0 = precisión exacta, 100 = máxima suavidad y velocidad).
- **CDIS (Approximation by Distance)** Aproximación en movimientos lineales (LIN) según distancia. Si CDIS = 5mm, el robot empieza a girar antes de llegar exactamente al punto.
- **CVEL (Approximation by Velocity)** Aproximación según velocidad del movimiento anterior.
- **CORI (Approximation by Orientation)** Máxima diferencia angular permitida para permitir aproximación.
- **Advance Run** Número de instrucciones que el robot puede leer "por adelantado". Afecta la suavidad de trayectorias. Valor típico: 3

START-/ENDPOSITION

Aquí se define las posturas articulares (ángulos de cada eje) para el inicio y/o fin de la simulación.

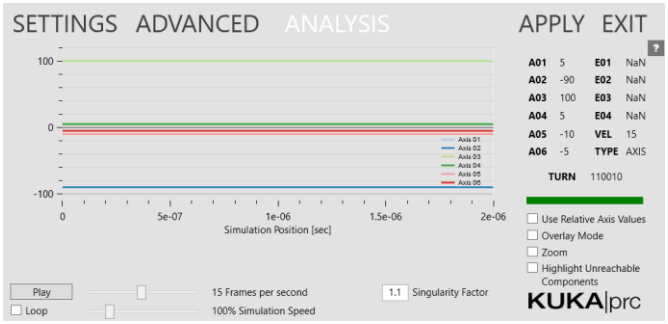


Figura. 22 *KUKA|prc Core Analysis*

Es donde se visualiza el comportamiento de cada eje a lo largo de la simulación.

Ahora bien, se procede a colocar el robot a usar en la creación de trayectorias. El proceso para colocar es seleccionar en el apartado de Virtual Robot el robot KUKA KR30-L16 y dicho bloque conectarlo al KUKA|prc Core a la entrada ROBOT y así es como el brazo robótico ingresa en la zona de trabajo como se puede observar en la figura 23.

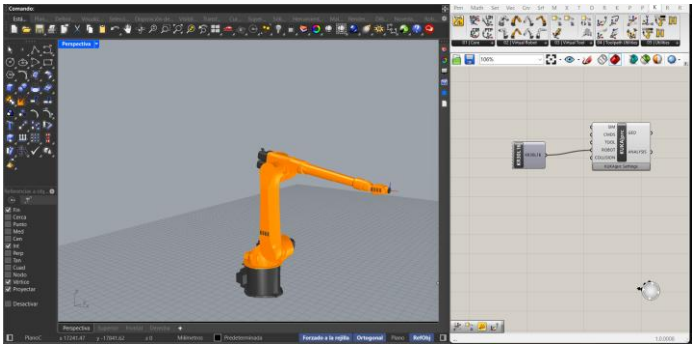


Figura. 23 *Área de trabajo virtual*

Una vez ingresado el robot se debe colocar la herramienta que se tiene en el robot real la cual es una antorcha BINZEL ABICOR TORCH NECK WH W500 para soldadura MIG. Dicha herramienta no consta en el apartado de TOOL en KUKA|prc pero si existe la opción de ingresar una herramienta externa al programa. El proceso para ingresar una herramienta a KUKA|prc es el siguiente:

1. Tener la malla o elemento CAD de la herramienta a colocar. En este caso se consiguió el elemento CAD del fabricante. Teniendo la herramienta de manera virtual debe ser importada al documento en Rhino. Una vez dentro de Rhino se puede utilizar en Grasshopper. Para usar en Grasshopper se debe seleccionar un bloque llamado Geometry en el cual se debe dar clic derecho y colocar la opción de “Set One Geometry”

y dar clic sobre la pieza CAD y así se selecciona la pieza como se muestra en la figura 24 y así ya está a disposición.

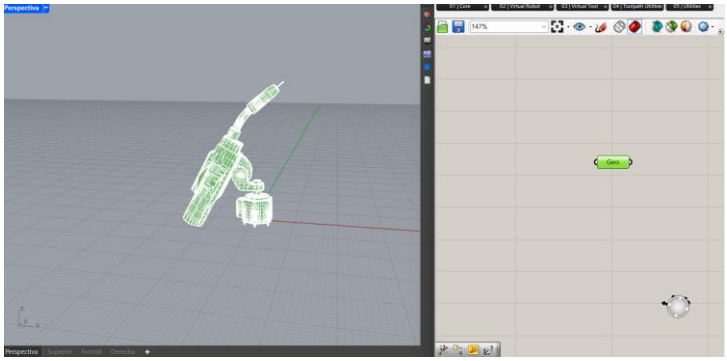


Figura. 24 *Herramienta Malla*

2. Se debe tener los valores del TCP (Tool Center Point) el cual es importante para que la herramienta esté centrada y a su vez generar una simulación precisa y exacta al entorno real. Y esto servirá para colocar los valores en el bloque de "CUSTOM TOOL" en el botón de "Set Tool"

   De no tener dichos datos, deben ser medidos mediante las opciones que menciona el fabricante en el manual.

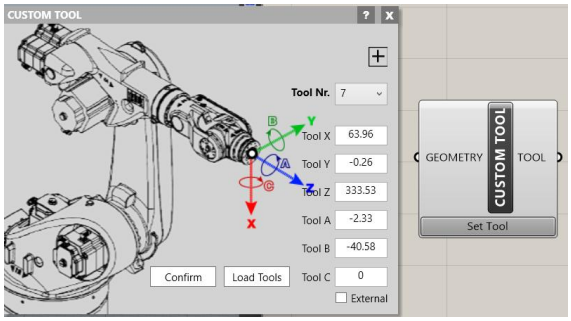


Figura. 25 *Valores de TCP*

3. Unir los bloques al KUKA|prc Core en la entrada de TOOL para así acoplar de manera exitosa la herramienta al robot.

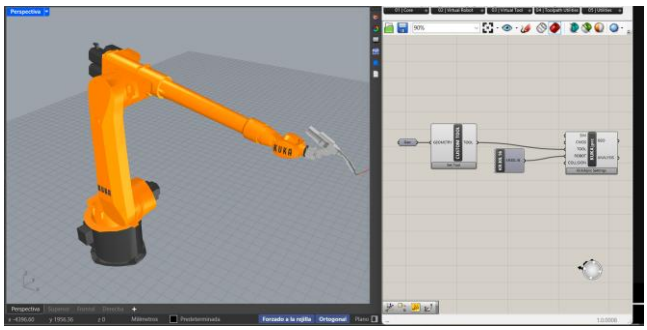


Figura. 26 *Colocación de Herramienta*

Una vez colocados el robot y la herramienta real en el entorno virtual, se procede a generar las trayectorias conforme a los requerimientos del proceso. No obstante, antes de realizar esta

etapa, es fundamental tener clara la definición y el manejo de los conceptos de TCP (Tool Center Point) y Base, los cuales deben ser estudiados y comprendidos en profundidad para garantizar una programación precisa y segura.

3.4.2. TCP (Tool Center Point)

Significa generar un sistema de coordenadas que tienen su origen en el punto de referencia de herramienta. Este punto de referencia se denomina TCP, mientras que el sistema de coordenadas es el sistema de coordenadas de herramienta. Durante la medición se guarda la distancia entre el origen del sistema de coordenadas de la herramienta (X, Y y Z) y el sistema de coordenadas de brida, además del giro entre si (ángulo A, B y C).

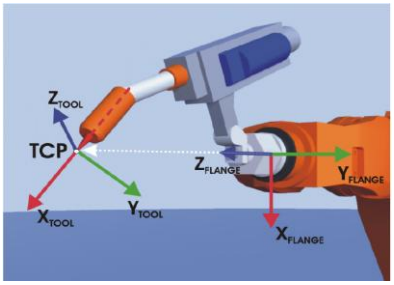


Figura. 27 *TCP*

Para este proceso se tiene 4 alternativas para medir. Los cuales vienen dado en el manual del brazo robótico.

Medición del TCP

- Método XYZ 4 puntos
- Método de referencia XYZ

Medición de la orientación

- Método ABC- Word
- Método ABC 2 puntos

Todo esto sirve para la medición de la herramienta y así generar procesos exactos y precisos.

3.4.3. Base

Significa crear un sistema de coordenadas en determinado punto del entorno del robot a partir del sistema universal de coordenadas. El objetivo consiste en aplicar los movimientos y las posiciones programadas del robot a este sistema de coordenadas. Los cantos definidos de los alojamientos de las piezas, las superficies, los cantos exteriores de los pallets o de la máquina son puntos de referencia muy útiles para el sistema de coordenadas base[51]. En resumen, es

generar un sistema de coordenadas en el cual va a ser el área de trabajo sobre el cual va a ejecutarse la programación, por ejemplo, como un llamado “cero pieza”.

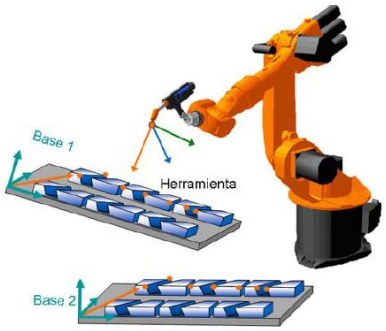


Figura. 28 *Base[52]*

Esta medición se realiza en dos pasos:

1. Determinación del origen de las coordenadas
2. Definición de la dirección de las coordenadas

Existen varias opciones para medir la base, los cuales son:

1. Método de los 3 puntos

En este se define 3 puntos de los cuales el primero define el origen que se aprecia el ejemplo en la figura 29.

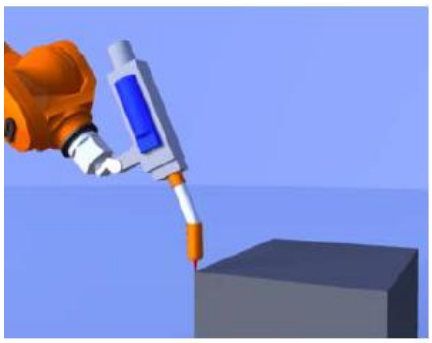


Figura. 29 *Medición Punto de Origen[52]*

El segundo punto se genera moviendo el robot para donde se quiere que sea la coordenada X como se visualiza en la figura 30.

Figura. 30 *Medición en X[52]*

El punto tres es moviéndonos en dirección hacia dónde se desee la coordenada Y como se observa en la figura 31.

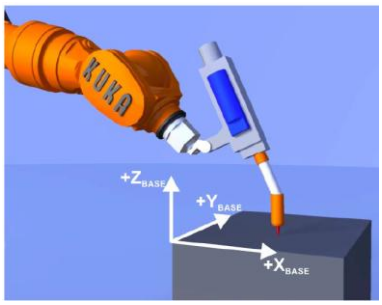


Figura. 31 *Medición en Y[52]*

2. Método indirecto

   El método indirecto se utiliza cuando no es posible llegar con el robot al origen de la base, por ej. porque se encuentra en el interior de una pieza o fuera del campo de trabajo del robot. Debe efectuarse el desplazamiento a 4 puntos de la base, cuyas coordenadas deben conocerse (datos CAD). La unidad de control del robot calcula la base utilizando estos puntos.

3. Entrada numérica

   Entrada directa de valores para la distancia al sistema de coordenadas universales (X, Y, Z) y del giro (A, B, C).

Se deben comprender estos conceptos clave para una correcta generación de trayectorias.

Comprendido los conceptos se puede comentar que los datos de TCP ingresados en el software fueron datos previamente medidos por el personal ya que esta herramienta es la que más suelen utilizar.

La figura 32 muestra los valores de TCP de la antorcha de soldadura.

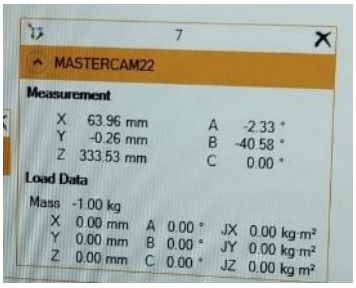


Figura. 32 *Valores Herramienta*

Dichos valores deben ser ingresados en Grasshopper como se observa en la figura 33.

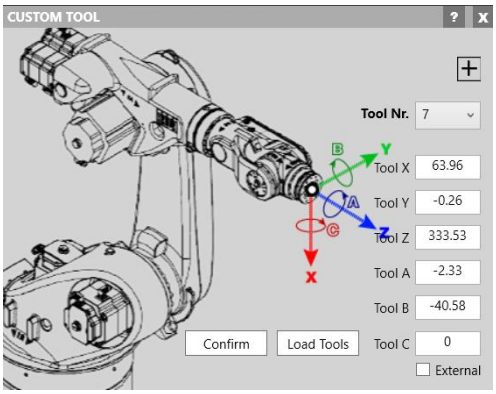


Figura. 33 *Valores Herramienta Virtual*

Para los valores de base se utilizó la mesa giratoria de la cabina de soldadura es decir el 8vo grado de libertad del robot. Esa medida fue tomada previamente por personal del CIRT en la cual el punto de origen es el centro de la mesa giratoria mediante el proceso de medición de 3 puntos. (No olvidarse de colocar el mismo número de herramienta en el programa y sea el mismo número de herramienta creado en el brazo KUKA).

La figura 34 muestra los valores de Base los cuales fueron medidos en la mesa giratoria.

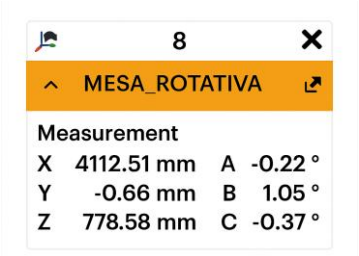


Figura. 34 *Valores Base Mesa Rotativa*

Dichos valores deben ser ingresados en Grasshopper como se aprecia en la figura 35. (No olvidarse de colocar el mismo número de base en el programa sea el mismo número de base creado en el brazo KUKA).

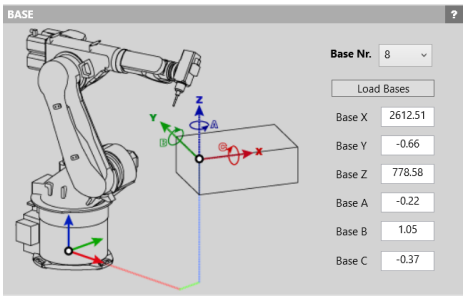


Figura. 35 *Valores Base Virtual*

La figura 36 muestra cómo se vería la base virtual en el entorno real.

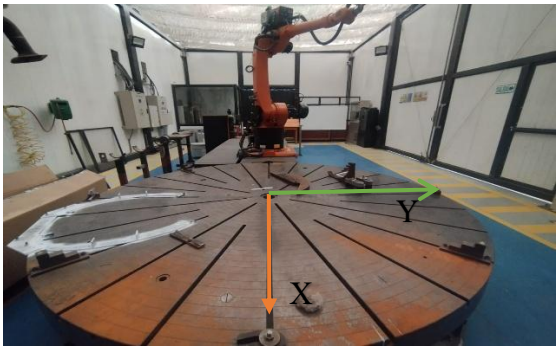


Figura. 36 *Posición Base*

Se debe tener en cuenta que el valor en X 4112.51mm, pero a este valor se lo resta 1500 debido que en la unidad lineal (E1) no se posicionó en su valor 0 ya que el robot tiende a estar muy alejado a la pieza es por esto por lo que se lo recorrió 1500mm para así estar más cerca a la mesa giratoria. En la figura 37 se comprende el movimiento realizado en la unidad lineal.

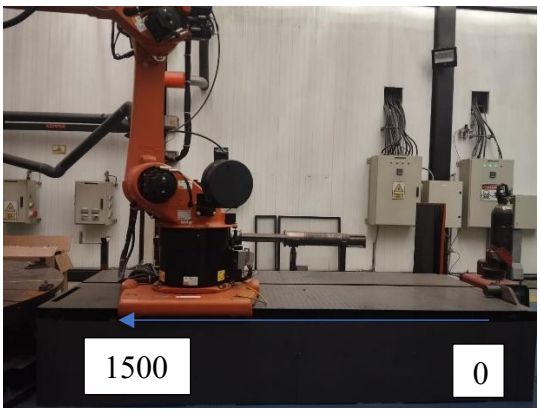


Figura. 37 *Desplazamiento Unidad Lineal*

3.5. PROGRAMACIÓN

Para la programación primero se ubicó la pieza en la posición real del entorno. Es decir, posicionado en la ubicación real sobre la mesa giratoria. Para entender de mejor manera la posición de la pieza se puede observar la figura 38 en la que se puede apreciar la ubicación en la que se encuentra con respecto al robot y toda el área de trabajo.

Figura. 38 *Posición pieza lateral*

En la figura 39 se puede apreciar la ubicación de la pieza a trabajar posicionandose de frente al robot.

Figura. 39 *Posición pieza vista frontal*

Como se puede observar se encuentra posicionado en una ubicación sobre la mesa giratoria. Las ranuras de la mesa giratoria fueron también escaneadas para ayudar al momento de centrar en el Rhino. Esto se logró proyectando las líneas de la ranura escaneada y a su vez se realiza un offset a la línea 20mm la cual es la medida de la ranura de la mesa.

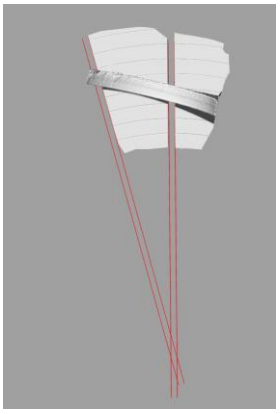


Figura. 40 *Líneas ranuras*

De estas dos líneas se creó una línea en el medio. Y es así como ambas líneas coinciden en un punto y dicho punto es el centro de la mesa la cual es el punto de origen de la Base.

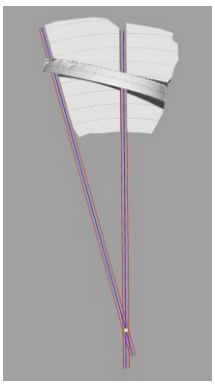


Figura. 41 *Líneas Centro de la Mesa*

Para lograr la posición exacta de la pieza se utilizó las ranuras alrededor del círculo las cuales dividían a la mesa en 24 partes iguales y la pieza se encuentra en la ranura 2 y 3 contando desde la división horizontal. La figura 42 explica la ubicación de la pieza respecto a las ranuras de la mesa.

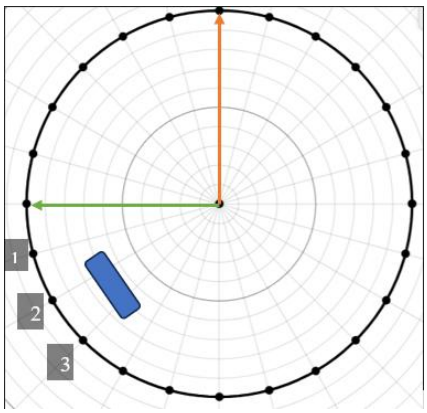


Figura. 42 *Posición Pieza sobre la mesa*

Es por eso por lo que la pieza fue girada 30º para así colocarle en la posición real en la cual se encuentra la pieza a soldar.

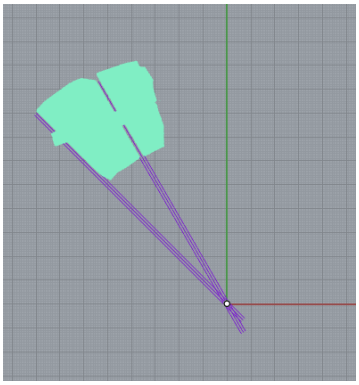


Figura. 43 *Posicionamiento final*

Concluyendo así la creación del entorno virtual del área de trabajo la cual simula de una manera precisa el área de trabajo real.

Ahora bien, se procede a dibujar una línea sobre la malla desde la vista Superior para posteriormente usar el comando _Project para poder proyectar dicha línea sobre la superficie de la malla. Estas líneas sobre la malla serán las trayectorias por donde se moverá el robot.

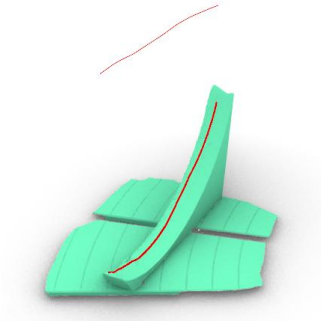


Figura. 44 *Línea sobre la malla*

Para la programación se debe trasladar al entorno de Grasshopper para crear el código.

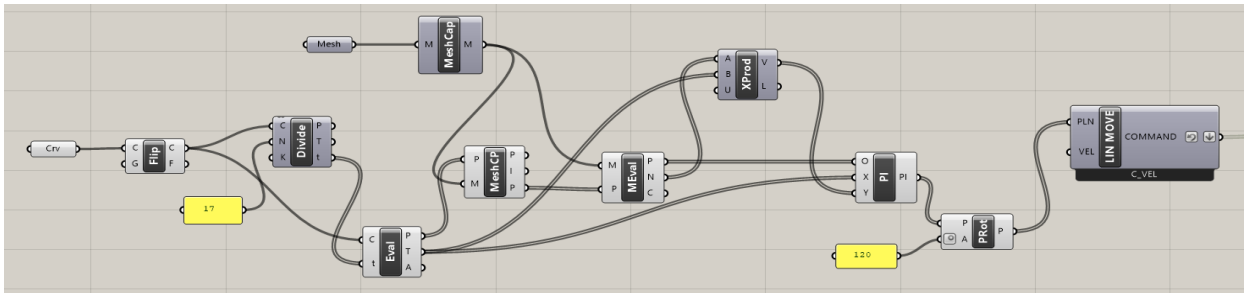


Figura. 45 *Código Grasshopper Línea sobre la malla*

Para la generación de trayectorias sobre superficies irregulares, se implementó un algoritmo en Grasshopper orientado al control de movimiento de un brazo robótico. El procedimiento parte de una curva de entrada que representa la trayectoria base, la cual es sometida a un ajuste de dirección mediante el componente Flip Curve, asegurando la consistencia del sentido de

recorrido. A continuación, se realiza una división de dicha curva en un número definido de segmentos discretos utilizando el bloque Divide Curve, obteniéndose así los puntos de control, parámetros normalizados y vectores tangentes asociados.

Con el objetivo de adaptar la trayectoria a la geometría superficial, se emplea una malla previamente generada a partir del objeto de interés. A través del componente Mesh Closest Point, se determina el punto más cercano sobre la malla para cada punto de la curva dividida. Posteriormente, con Mesh Evaluate, se extraen las coordenadas exactas y vectores normales correspondientes a cada uno de estos puntos sobre la superficie mapeada.

La orientación de la herramienta en cada posición se define mediante un sistema de referencia local. El eje X del plano se determina a partir de la tangente de la curva original, evaluada en los mismos parámetros de subdivisión mediante Evaluate Curve. A partir del producto vectorial entre dicha tangente y la normal superficial (obtenida del Mesh Evaluate), se calcula el eje Y, garantizando así la ortogonalidad del marco de referencia. El eje Z se corresponde directamente con la normal de la malla. Con esta información, se construyen planos ortogonales localizados sobre la superficie utilizando Construct Plane, tomando como origen el punto sobre la malla y como ejes los vectores mencionados.

Para evitar colisiones o permitir la ejecución de procesos como la soldadura aérea, se realiza una rotación de los planos calculados mediante el componente Rotate Plane, aplicando un giro de 120° a lo largo del eje normal. Finalmente, los planos resultantes son ingresados al bloque LIN MOVE del plugin KUKA|prc, el cual genera los movimientos lineales del robot en modo interpolado continuo (C_VEL), garantizando una trayectoria fluida sin detenciones entre puntos.

Para seguridad se creó un punto seguro a donde debe movilizarse el robot después de terminar la soldadura.

Teniendo como resultado un código KRL generado en un archivo .src el cual se observa en la figura 46.

```
DEF SueldaFinal3Sp ( )


INI

STARTPOSITION - BASE IS 8, TOOL IS 7, SPEED IS 15%, POSITION IS A1 5,A2 -90,A3 100,A4 5,A5 -10,A6 -5,E1 1500,E2 0

LIN SPEED IS 0.25 m/sec, INTERPOLATION SETTINGS IN FOLD

PTP {X -719.193, Y 709.12, Z 45.711, A 159.721, B 69.448, C -158.329, E1 0, E2 0, E3 0, E4 0, S 'B110'} C_PTP
LIN {X -719.193, Y 709.12, Z 45.711, A 159.721, B 69.448, C -158.329, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -693.815, Y 718.304, Z 31.399, A -87.285, B 60.4, C -49.575, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -665.396, Y 727.641, Z 22.775, A -155.51, B 77.359, C -109.845, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -635.085, Y 733.815, Z 18.327, A 149.226, B 70.09, C -158.772, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -605.185, Y 740.707, Z 13.222, A 121.022, B 83.502, C 163.793, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -575.347, Y 750.008, Z 13.004, A 79.769, B 79.84, C 122.083, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -545.611, Y 759.378, Z 15.965, A 65.695, B 78.088, C 107.439, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -516.481, Y 768.638, Z 22.749, A 34.744, B 73.399, C 76.435, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -488.591, Y 778.388, Z 33.064, A 61.703, B 68.523, C 99.712, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -461.53, Y 788.213, Z 45.374, A 48.162, B 65.168, C 85.957, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -435.476, Y 796.086, Z 60.798, A 21.815, B 53.552, C 66.015, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -410.979, Y 802.354, Z 79.158, A 36.931, B 54.404, C 78.681, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -387.555, Y 806.582, Z 99.342, A 25.546, B 43.629, C 73.177, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -365.889, Y 809.166, Z 121.769, A 20.597, B 36.433, C 68.296, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -346.813, Y 811.442, Z 146.42, A 21.928, B 32.45, C 68.252, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -330.263, Y 813.416, Z 172.676, A 23.753, B 28.137, C 68.179, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -316.58, Y 814.158, Z 200.792, A 28.004, B 23.345, C 70.488, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -306.03, Y 814.704, Z 230.008, A 18.016, B 14.455, C 63.84, E1 0, E2 0, E3 0, E4 0} C_VEL
LIN {X -530, Y 820, Z 290, A 90, B 72, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
PTP {A1 5, A2 -90, A3 100, A4 5, A5 -10, A6 -5, E1 1500, E2 0, E3 0, E4 0} C_PTP

END
```

Figura. 46 *Código KRL línea sobre la malla*

Para comprender de mejor manera el código presentado se lo explicará línea por línea.

DEF: Define el nombre del procedimiento. En este caso: SueldaFinal3Sp.

INI: Sección de inicialización.

Comentarios descriptivos. Informan que:

- La base activa es la nº 8.
- El TCP (Tool Center Point) o herramienta es la nº **7.**
- Velocidad inicial: 15%
- Posición articular de arranque (A1-A6 y E1-E2).

"LIN SPEED IS 0.25 m/sec, INTERPOLATION SETTINGS IN FOLD"

Comentario informativo: la velocidad de movimiento lineal es 0.25 m/s.

"PTP {X -719.193, Y 709.12, Z 45.711, A 159.721, B 69.448, C -158.329, E1 0, E2 0, E3 0, E4 0} C_PTP"

- Es el movimiento inicial.
- Movimiento tipo PTP (Point To Point) hacia una posición cartesiana específica.
- Posición en coordenadas XYZ + orientación en ángulos de Euler (ABC).
- **E1-E4**: Ejes externos (en este caso están todos en 0).
- **C_PTP**: Indicador de aproximación continua tipo PTP (movimiento fluido).

“LIN {X ..., Y ..., Z ..., A ..., B ..., C ..., E1 ..., E2 ..., E3 ..., E4 ...} C_VEL”

- Serie de movimientos de trayectoria.
- Son movimientos LINEALES (LIN) entre puntos consecutivos, ideales para trayectorias de soldadura.
- **C_VEL**: Aproximación continua con velocidad constante, útil para mantener un cordón uniforme durante la soldadura.
- Cada línea define una posición en el espacio 3D (posición + orientación).

“PTP {A1 5.2, A2 -90, A3 100, A4 5, A5 -10, A6 -5, E1 1500, E2 0, E3 0, E4 0} C_PTP”

Retorno a la posición inicial en modo PTP a la posición articular inicial definida en la sección de inicio.

“END”

Finaliza el programa.

Pero como se mencionó antes el E1 está movido 1500mm dicho valor debe ser reemplazado en el software OrangeEdit el cual ayuda a modificar códigos y es un entorno de manipulación de códigos KRL.

Es así como al aplastar Ctr+R se despliega una ventana donde se coloca que valor se desea buscar y por qué valor desea cambiar que en este caso se busca (E1 0) y se lo reemplaza por (E1 1500) y al dar clic en “Replace all” y así es como se reemplazarán esos valores en todo el documento como se observa en la Figura 47.

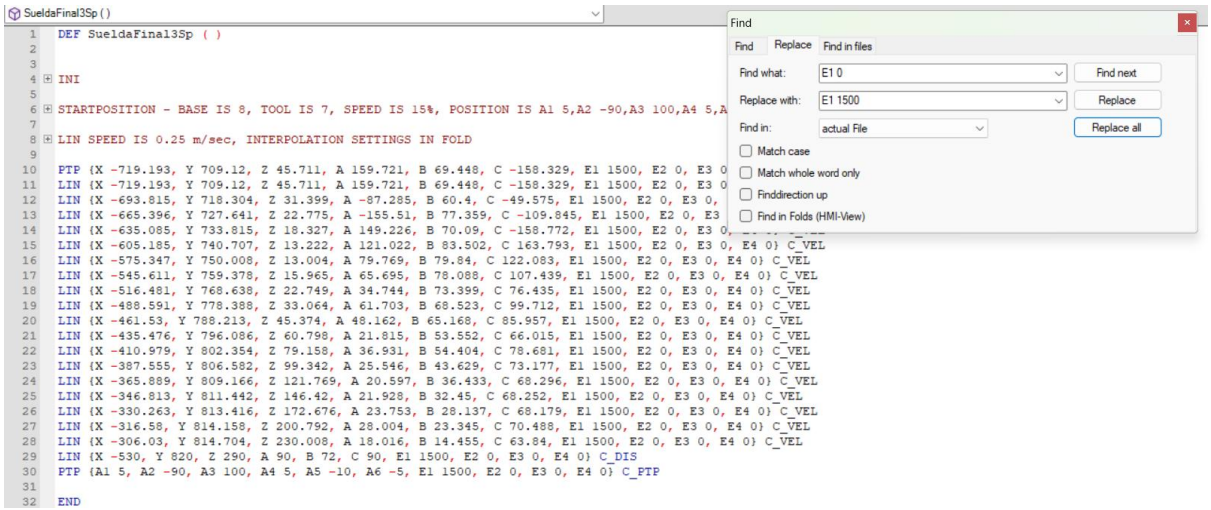


Figura. 47 *Remplazo de valores*

Dicho programa que se está manipulando se trata de un documento .src pero para la programación y vinculación con el paquete tecnológico ArcTech es necesario de un documento

.dat en el que se debe crear en OrangeEdit en el cual se declara las posiciones, y declarar líneas de código del paquete de soldadura (Se debe tener en cuenta que los dos documentos deben tener el mismo nombre tanto el .src como el .dat). Todo esto ayudará con la vinculación de programación ArcTech. Ya que este se moviliza a partir de llamar puntos y comandos previamente guardados en el control del robot (documento .dat).

Grasshopper y KUKA|prc genera solo un documento .src más no un documento .dat es por eso por lo que se ha visto en la necesidad de crearlo manualmente, es decir, colocar manualmente las declaraciones necesarias en el documento .dat pero dicho archivo se puede generar con la ayuda de OrangeEdit y cualquier inteligencia artificial en este caso Gemini IA ya que después de varias pruebas e investigaciones se determinó líneas de código que son base para todos los programas a generar. Simplemente se deben reemplazar ciertos valores con valores del código y es aquí donde ayuda la IA. Para comprender de mejor manera se realizará la explicación de las líneas de código y como deben ser reemplazadas dependiendo la necesidad.

Se declara los puntos generados por KUKA|prc en el documento .dat donde primero teniendo los puntos del documento .src se procede a declarar.

Punto en documento .src

LIN {X -719.193, Y 709.12, Z 45.711, A 159.721, B 69.448, C -158.329, E1 1500, E2 0, E3 0, E4 0} C_VEL

Declaración que se debe colocar en el archivo .dat

DECL E6POS P1={ X -719.193, Y 709.12, Z 45.711, A 159.721, B 69.448, C -158.329, E1 1500, E2 0, E3 0, E4 0}

Este proceso se debe repetir con todos los puntos generados en él .src. pero esto se ve resuelto al pedir a la IA. En la IA se cargan todos los puntos que se tiene en él .src y se le pide que cambie los “LIN” por “DECL E6POS P1”, luego por “DECL E6POS P2”, y así sucesivamente. Esto se debe copiar y pegar en el archivo .dat. Logrando así declarar todos los puntos.

Línea de código base

DECL FDAT FP1={TOOL_NO 7,BASE_NO 8,IPO_FRAME #BASE}

Tener en cuenta que el número de herramienta y base sea la que se vaya a usar en dicho programa, caso contrario reemplazar. En este caso se estaba trabajando con la herramienta (Tool) 7 y la Base 8.

De esta línea se debe reemplazar el FP1 por FP2, FP3 hasta el número de puntos que se tenga en dicha programación.

Línea de código

DECL DAT LCPDAT1={ACC100.000,APO_DIST100.000,APO_FAC 50.0000,AXIS_VEL100.000,AXIS_ACC100.000,ORI_TYP#VAR,CIRC_TYP #BASE, JERK_FAC 50.0000,GEAR_JERK 50.0000,EXAX_IGN 0}

De esta línea se debe reemplazar el LCPDAT1 por LCPDAT2, LCPDAT3 hasta el número de puntos que se tenga en dicha programación. Tener en cuenta que siempre después de TYP diga #VAR y no #CONSTANT ya que eso implica que el ángulo de la antorcha sea variable durante la trayectoria y no constante lo cual puede perjudicar al trabajar con superficies no planas.

Línea de código

DECL PDAT PPDAT1={VEL 100.000,ACC 100.000,APO_DIST 100.000,GEAR_JERK 50.0000,EXAX_IGN 0}

De esta línea se debe reemplazar el PPDAT1 por PPDAT2, PPDAT3 hasta el número de puntos que se tenga en dicha programación.

3.6. VINCULACIÓN CON TECNOLOGÍA ARCTECH

Se comienza por declarar los WDAT (Weld Data Set) este es el nombre de la estructura general de soldadura el cual contiene datos de AS (Arc Start) en este vienen los parámetros de encendido del arco. AW (Arc Weld) este contiene los parámetros de soldadura activa es decir cuando el arco ya este encendido. AC (Arc Crater) este viene con los parámetros para apagar el arco correctamente.

Para vincular esta tecnología se partió de un archivo creado por los soldadores de manera manual mediante el SmartPath el cual se muestra en la figura 48. Y se lo exportó el archivo .src y el .dat a la computadora para mejor visualización.

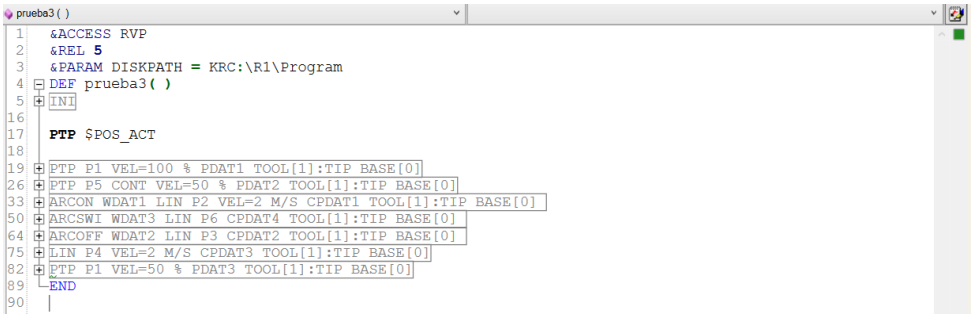


Figura. 48 *Archivo .src*

En la figura 49 se aprecia las declaraciones de todos los parámetros de soldadura que contiene el archivo .dat.

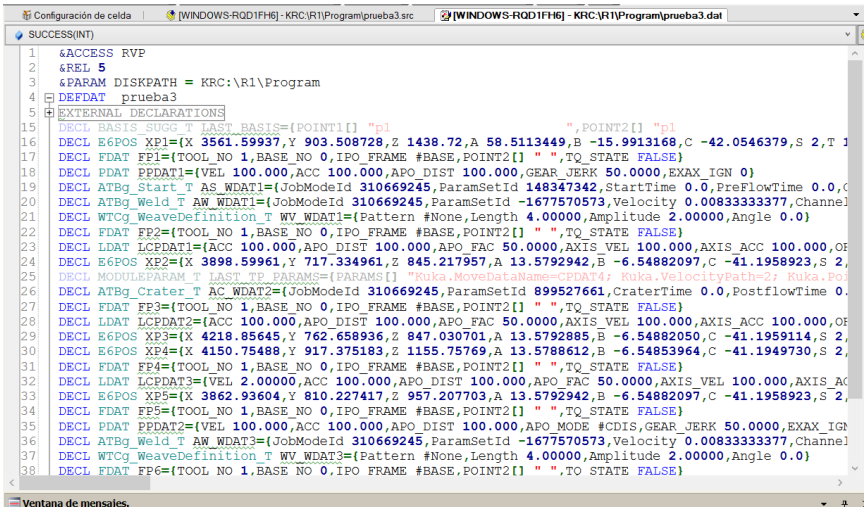


Figura. 49 *Archivo.dat*

Se seleccionó este programa ya que cuenta con los 3 comandos principales del paquete tecnológico ArcTech los cuales son:

ARC ON. La instrucción ARC ON contiene el movimiento hacia la posición de encendido, así como los parámetros de encendido, de soldadura y de oscilación. La instrucción ARC ON finaliza cuando el arco eléctrico está encendido y los parámetros de soldadura conectados.

ARC SWITCH. La instrucción ARC SWITCH se utiliza para dividir una costura en varias secciones. Una instrucción ARC SWITCH contiene los parámetros de movimiento, soldadura y oscilación para una de las secciones. Siempre se da un posicionamiento aproximado del punto de destino. Es utilizado cuando se necesita pasar a otro punto o movimiento sin apagar el arco eléctrico y variando los parámetros de soldadura dependiendo la necesidad del usuario.

ARC OFF. Este comando finaliza el proceso de soldadura en la posición del cráter final. El cráter final se llena en la posición del cráter final.

Y como se pudo apreciar en las figuras 50 y 51 esta programación sirvió de base para poder crear el código para la pieza, pero sin olvidar que el código que se tomó como referencia fue para soldar en superficies planas y se le adecuó para que suelde en una superficie tridimensional.

De dicho código se utiliza las declaraciones de los WDAT que se encontraban en el archivo .dat que en este caso eran 3 WDAT1 que se usó para el encendido del arco (ARC ON) el WDAT2 para el apagado del arco (ARC OFF) y finalmente WDAT3 que fue para el cambio de punto a otro punto manteniendo el arco encendido (ARC SWITCH). Los cuales se los copió todas las líneas que tuvieran información acerca de los WDAT y pegó en el documento .dat.

Una vez listo el archivo .dat con toda la información a utilizar declarada se pasó a trabajar con el archivo .src para la programación de los comandos de soldadura. Los cuales también fueron reutilizados y acoplados del documento base creado por los especialistas soldadores y que fue extraído para su visualización y uso.

Para la programación de los códigos de soldadura se partió por entender que en el primer punto declarado es decir el P1 es donde se desea que se encienda el arco y se mantenga encendido pasando por el resto de los puntos para finalmente apagarse al llegar el punto final. Es decir, en el P1 se necesita el comando ArcOn, en el resto de los puntos hasta el 17 se necesita un ArcSwitch y finalmente en el punto 18 se necesita un ArcOff.

Se copió la estructura de los comandos Arc ya que es todo un Fold más no solo una línea. Al hablar de Fold es una estructura especial que se utiliza para organizar y simplificar el código en el SmartPath o en el editor de KUKA (por ejemplo, WorkVisual).

El término FOLD no es una instrucción funcional del robot; es más bien una marca de estructura que le indica al sistema (y al programador) que un bloque de instrucciones puede plegarse o desplegarse visualmente. Para entender de mejor manera se presentan la figura 50 que es el FOLD sin desplegar.

```
ARCON WDAT1 LIN P2 VEL=2 M/S CPDAT1 TOOL[1]:TIP BASE[0]
ARCSWI WDAT3 LIN P6 CPDAT4 TOOL[1]:TIP BASE[0]
ARCOFF WDAT2 LIN P3 CPDAT2 TOOL[1]:TIP BASE[0]
```

Figura. 50 *Código sin desplegar*

En la figura 51 se muestra lo que contiene el FOLD ya desplegado.

```
;FOLD ARCON WDAT1 LIN P2 Vel=2 m/s CPDAT1 Tool[1]:TIP Base[0] ;%{PE}
;FOLD Parameters ;%{h}
;Params IlfProvider=kukaroboter.arctech.arconlin; Kuka.PointName=P2; Kuka.BlendingEnabled=False; Kuka.MoveDataName=CPDAT1;
;FOLD Parameters ArcTechAdvanced ;%{h}
;Params IgnitionErrorStrategy=1; WeldErrorStrategy=1
;ENDFOLD ArcTechAdvancedFoldEnd
;ENDFOLD
$BWDSTART = FALSE
LDAT_ACT = LCPDAT1
FDAT_ACT = FP2
BAS(#CP_PARAMS, 2.0)
TRIGGER WHEN DISTANCE = 1 DELAY = ATBg_PreDefinitionTime DO Arc_DefineStrikeParams(1, AS_WDAT1) PRIO = -1
Arc_DefineCpPattern(#OffInAdvance, WV_WDAT1, TRUE)
LIN XP2
Arc_On(1, AS_WDAT1, ATBg_StartErrSetField[1], AW_WDAT1, WV_WDAT1, ATBg_WeldErrSetField[1], #StdArcOn, " ")
Arc_DefineCpPattern(#OnInAdvance, WV_WDAT1, TRUE)
;ENDFOLD
;FOLD ARCSWI WDAT3 LIN P6 CPDAT4 Tool[1]:TIP Base[0] ;%{PE}
;FOLD Parameters ;%{h}
;Params IlfProvider=kukaroboter.arctech.arcswilin; Kuka.PointName=P6; Kuka.BlendingEnabled=True; Kuka.MoveDataName=CPDAT4;
;ENDFOLD
$BWDSTART = FALSE
LDAT_ACT = LCPDAT4
LDAT_ACT.APO_Dist = ATBg_APODistanceArcTech
FDAT_ACT = FP6
BAS(#CP_PARAMS, ATBg_BAS_VELDefinition)
TRIGGER WHEN DISTANCE = 1 DELAY = 0 DO Arc_Swi(1, AW_WDAT3, WV_WDAT3, ATBg_WeldErrSetField[1]) PRIO = -1
LIN XP6 C_Dis C_Dis
ATB_Definition(AW_WDAT3)
Arc_DefineCpPattern(#OnInAdvance, WV_WDAT3, TRUE)
;ENDFOLD
;FOLD ARCOFF WDAT2 LIN P3 CPDAT2 Tool[1]:TIP Base[0] ;%{PE}
;FOLD Parameters ;%{h}
;Params IlfProvider=kukaroboter.arctech.arcofflin; Kuka.PointName=P3; Kuka.BlendingEnabled=False; Kuka.MoveDataName=CPDAT2
;ENDFOLD
$BWDSTART = FALSE
LDAT_ACT = LCPDAT2
```

Figura. 51 *Código desplegado FOLD*

Y como se aprecia la estructura de programación de los comandos es fija solo cambia dependiendo que WDAT se usa y en qué punto se va a usar.

3.6.1. ESTRUCTURA ARCON

En la figura 52 se observa la estructura del ArcOn el cual se le adecuó para la aplicación cambiando por los datos que se va a utilizar es decir Tool 7 y Base 8 y el punto 2 donde hará el encendido del arco. Se mantendrá el uso del WDAT1.

```
;FOLD ARCON WDAT1 LIN P2 Vel=2 m/s CPDAT1 Tool[1]:TIP Base[0] ;%{PE}
;FOLD Parameters ;%{h}
;Params IlfProvider=kukaroboter.arctech.arconlin; Kuka.PointName=P2; Kuka.BlendingEnabled=False; Kuka.MoveDat
;FOLD Parameters ArcTechAdvanced ;%{h}
;Params IgnitionErrorStrategy=1; WeldErrorStrategy=1
;ENDFOLD ArcTechAdvancedFoldEnd
;ENDFOLD
$BWDSTART = FALSE
LDAT_ACT = LCPDAT1
FDAT_ACT = FP2
BAS(#CP_PARAMS, 2.0)
TRIGGER WHEN DISTANCE = 1 DELAY = ATBg_PreDefinitionTime DO Arc_DefineStrikeParams(1, AS_WDAT1) PRIO = -1
Arc_DefineCpPattern(#OffInAdvance, WV_WDAT1, TRUE)
LIN XP2
Arc_On(1, AS_WDAT1, ATBg_StartErrSetField[1], AW_WDAT1, WV_WDAT1, ATBg_WeldErrSetField[1], #StdArcOn, " ")
Arc_DefineCpPattern(#OnInAdvance, WV_WDAT1, TRUE)
;ENDFOLD
```

Figura. 52 *Estructura ArcOn*

Explicación del código para comprender de mejor manera:

“;FOLD ARCON WDAT1 LIN P2 Vel=2 m/s CPDAT1 Tool[1]:TIP Base[0] ;%{PE}”

- **ARCON WDAT1**: Activa el arco de soldadura con los parámetros definidos en la estructura WDAT1 (estructura de datos de soldadura: voltaje, corriente, etc.).
- **LIN P2**: Movimiento lineal hacia el punto P2.
- **Vel=2 m/s**: Velocidad de soldadura de 2 m/s.
- **CPDAT1**: Estructura de datos de trayectoria (coordenadas, velocidad, aproximación, etc.).
- **Tool[1]**: Herramienta nº1 (por ejemplo, la antorcha).
- **Base[0]**: Sistema de coordenadas base.
- ;%{PE}: Marca especial de los FOLD, generada por el sistema (no se edita manualmente).

“;FOLD Parameters ;%{h}

;Params tlfProvider=kukaroboter.arctech.arconlin; ...”

Estos son parámetros internos que configuran el comportamiento del encendido del arco:

- tlfProvider: proveedor lógico del arco (comunicación interna con el sistema ArcTech).
- Kuka.PointName=P2: referencia del punto al que se moverá.
- Kuka.BlendingEnabled=False: no hay blending (suavizado entre movimientos).

- Kuka.MoveDataName=CPDAT1: datos de movimiento.

“;FOLD Parameters ArcTechAdvanced ;%{h}

;Params IgnitionErrorStrategy=1; WeldErrorStrategy=1”

- **IgnitionErrorStrategy=1**: cómo actuar ante un fallo de encendido del arco (por ejemplo, reintentar o detener).
- **WeldErrorStrategy=1**: estrategia ante error de soldadura.

“$BWDSTART = FALSE

LDAT_ACT = LCPDAT1

FDAT_ACT = FP2

BAS(#CP_PARAMS, 2.0)”

- **$BWDSTART**: evita que el programa se ejecute en retroceso (backward run).
- **LDAT_ACT**: asigna datos de movimiento actuales desde LCPDAT1.
- **FDAT_ACT**: asigna datos del punto final (FP2).
- **BAS(#CP_PARAMS, 2.0)**: llama a una rutina estándar de KUKA para configurar velocidad/cartesiano con una velocidad de 2 m/s.

“TRIGGER WHEN DISTANCE = 1 DELAY = ATBg_PreDefinitionTime DO Arc_DefineStrikeParams(...)”

- **TRIGGER**: es un disparador.
- **WHEN DISTANCE = 1**: al estar a 1 mm del punto objetivo...
- **DELAY = ATBg_PreDefinitionTime**: se espera el tiempo de preencendido.
- **DO**: ejecuta lo siguiente:

“Arc_DefineStrikeParams(1, AS_WDAT1) ...”

- Define los parámetros de encendido del arco antes de llegar al punto.

“LIN XP2”

Movimiento lineal hasta el punto XP2 (puede ser una posición declarada en el archivo .DAT).

“Arc_On(1, AS_WDAT1, ATBg_StartErrSetField[1], AW_WDAT1, WV_WDAT1, ATBg_WeldErrSetField[1], $StdArcOn, " ")”

Arc_On: enciende el arco.

Parámetros:

- 1: canal del generador.
- AS_WDAT1: datos de encendido del arco.
- AW_WDAT1: parámetros de soldadura.
- WV_WDAT1: parámetros de velocidad del arco.
- $StdArcOn: configuración estándar para ArcOn.
- " ": sin comentario adicional.

"Arc_DefineCpPattern(#OnInAdvance, WV_WDAT1, TRUE)"

- Define el patrón de soldadura al encender el arco con anticipación.
- #OnInAdvance: es un tipo de evento programado para que el arco esté listo justo al llegar al punto.

3.6.2. ESTRUCTURA ARCSWITCH

Esta misma estructura del ArcSwitch que se aprecia en la figura 53 solo se lo cambió el WDAT3 por WDAT2 y se los reemplazó todos los puntos por los puntos a necesidad es decir desde el P3 hasta P17.

```
;FOLD ARCSWI WDAT3 LIN P6 CPDAT4 Tool[1]:TIP Base[0] ;%{PE}
;FOLD Parameters ;%{h}
;Params IlfProvider=kukaroboter.arctech.arcswilin; Kuka.PointName=P6; Kuka.BlendingEnabled=True; Kuka.MoveDataN
;ENDFOLD
$BWDSTART = FALSE
LDAT_ACT = LCPDAT4
LDAT_ACT.APO_Dist = ATBg_APODistanceArcTech
FDAT_ACT = FP6
BAS(#CP_PARAMS, ATBg_BAS_VELDefinition)
TRIGGER WHEN DISTANCE = 1 DELAY = 0 DO Arc_Swi(1, AW_WDAT3, WV_WDAT3, ATBg_WeldErrSetField[1]) PRIO = -1
LIN XP6 C_Dis C_Dis
ATB_Definition(AW_WDAT3)
Arc_DefineCpPattern(#OnInAdvance, WV_WDAT3, TRUE)
;ENDFOLD
```

Figura. 53 *Estructura ArcSwitch*

";FOLD ARCSWI WDAT3 LIN P6 CPDAT4 Tool[1]:TIP Base[0] ;%{PE}"

- **ARCSWI WDAT3**: Ejecuta un cambio de parámetros de soldadura (WDAT3) sin apagar el arco.
- **LIN P6**: Movimiento lineal hacia el punto P6.
- **CPDAT4**: Datos de movimiento (velocidad, aproximación, etc.).
- **Tool[1]**: Herramienta activa (por ejemplo, antorcha).
- **Base[0]**: Sistema de coordenadas base.

";FOLD Parameters ;%{h}

;Params tlfProvider=kukaroboter.arctech.arcswi; ...”

- PointName=P6: nombre del punto destino.
- BlendingEnabled=True: se permite interpolación entre movimientos, lo cual ayuda a no detener la soldadura.

“$BWDSTART = FALSE

LDAT_ACT = LCPDAT4

LDAT_ACT.APO_Dist = ATBg_APODistanceArcTech

FDAT_ACT = FP6”

- **$BWDSTART**: evita ejecución hacia atrás.
- **LDAT_ACT**: activa los datos de movimiento LCPDAT4.
- **LDAT_ACT.APO_Dist**: define la distancia de aproximación continua.
- **FDAT_ACT**: activa los datos del punto final (FP6).

“BAS(#CP_PARAMS, ATBg_BAS_VELDefinition)”

Configura la velocidad y otros parámetros mediante una rutina del sistema BAS().

“TRIGGER WHEN DISTANCE = 1 DELAY = 0 DO Arc_Swi(1, AW_WDAT3, WV_WDAT3, ATBg_WeldErrSetField[1]) PRIO = -1”

Cuando falte 1 mm para llegar al punto, realiza un cambio de parámetros de soldadura:

- Arc_Swi: función que cambia en caliente (hot-swap) los parámetros sin apagar el arco.
- Canal 1, nuevos parámetros AW_WDAT3, WV_WDAT3 (corriente, voltaje, etc.)
- ATBg_WeldErrSetField[1]: estrategia en caso de error.

“LIN XP6 C_Dis C_Dis”

- Movimiento lineal hacia el punto XP6.
- **C_Dis**: se mantiene la aproximación continua (el robot puede iniciar el próximo movimiento antes de terminar completamente este punto).

“ATB_Definition(AW_WDAT3)

Arc_DefineCpPattern(#OnInAdvance, WV_WDAT3, TRUE)”

- **ATB_Definition(AW_WDAT3)**: declara que el nuevo set de parámetros de soldadura **AW_WDAT3** está activo desde este punto.
- **Arc_DefineCpPattern(...)**: define el patrón del proceso con los nuevos parámetros de velocidad del arco WV_WDAT3, con anticipación (#OnInAdvance).

Teniendo así la estructura completa de los ArcSwitch que se necesitaba. Generando los Folds comprimidos que se visualizan en la figura 54.

```
ARCSWI WDAT2 LIN P11 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P12 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P13 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P14 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P15 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P16 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P17 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
```

Figura. 54 *Código implementado ArcTech*

3.6.3. ESTRUCTURA ARCOFF

En esta estructura como se visualiza en la figura 55 solo se reemplazó por WDAT2 por WDAT1 y por el P18 el cual es el punto final.

```
;FOLD ARCOFF WDAT1 LIN P5 CPDAT2 Tool[7]:MASTERCAM22 Base[19]:pruebaSoldadura ;%{PE}
;FOLD Parameters ;%{h}
;Params IlfProvider=kukaroboter.arctech.arcofflin; Kuka.PointName=P5; Kuka.BlendingEnabled=False; Kuka.MoveDataName=CPDAT2;
;Kuka.MovementParameterFieldEnabled=True; ArcTech.WdatVarName=WDAT1
;ENDFOLD
$BWDSTART = FALSE
LDAT_ACT = LCPDAT2
FDAT_ACT = FP5
BAS(#CP_PARAMS, ATBg_BAS_VELDefinition)
LIN XP5
Arc_Off(1, AC_WDAT1)
;ENDFOLD
```

Figura. 55 *Estructura ArcOff*

";FOLD ARCOFF WDAT1 LIN P5 CPDAT2 Tool[7]:MASTERCAM22 Base[19]:pruebaSoldadura ;%{PE}

;FOLD Parameters: %{{h}

;Params ...

;ENDFOLD"

ARCOFF WDAT1 LIN P5 CPDAT2: Describe la acción que se ejecuta:

- ARCOFF: Apaga el arco de soldadura.
- WDAT1: Nombre del conjunto de datos del arco (ArcTech).
- LIN P5 CPDAT2: Movimiento lineal hacia el punto P5 usando los datos CPDAT2.

Tool[7]: Herramienta activa número 7.

Base[19]: Base de trabajo número 19.

pruebaSoldadura: Comentario descriptivo o nombre del procedimiento.

Las líneas con ;Params ... contienen parámetros que indican:

- Kuka.PointName = P5: Punto destino.
- Kuka.MoveDataName = CPDAT2: Parámetros de movimiento.
- Kuka.BlendingEnabled = False: Sin suavizado de trayectoria.
- ArcTech.WeldVarName = WDAT1: Datos del arco usados.

“$BWDSTART = FALSE”

Desactiva el inicio en retroceso (útil para reanudar movimientos desde un punto anterior).

“LDAT_ACT = LCPDAT2

FDAT_ACT = FP5”

Asigna los datos de movimiento (LDAT_ACT) y posición (FDAT_ACT) actuales:

- LCPDAT2: Parámetros de movimiento.
- FP5: Punto de destino.

“BAS(#CP_PARAMS, AT6=BAS_VELDefinition)”

Llama a una rutina estándar de KUKA (BAS) para aplicar parámetros base de movimiento (velocidad, aceleración, etc.).

“LIN XP5”

Ejecuta un movimiento lineal hacia la posición XP5.

“Arc_Off(1, AC_WDAT1)”

Apaga el arco de soldadura. El parámetro 1 generalmente representa el número de canal (si hay múltiples procesos de soldadura). AC_WDAT1 es el nombre de los datos de soldadura para apagar el arco.

“:ENDFOLD”

Fin del programa.

Se usó el WDAT1 y WDAT2 ya que estos se los configuró con los parámetros óptimos para la soldadura sobre la pieza.

La configuración de valores de WDAT se los genera manualmente en el SmartPath de la siguiente manera. Al abrir el código en el robot mediante el Smartpath se puede posicionar el fichero en donde dice WDAT y colocar modificar y se desplegará la siguiente ventana donde se deberán colocar los valores a elección. Dependiendo de que comando se necesita modificar se desplegará diferentes ventanas donde se podrán colocar los valores a elección.

3.6.4. VENTANA ARCON

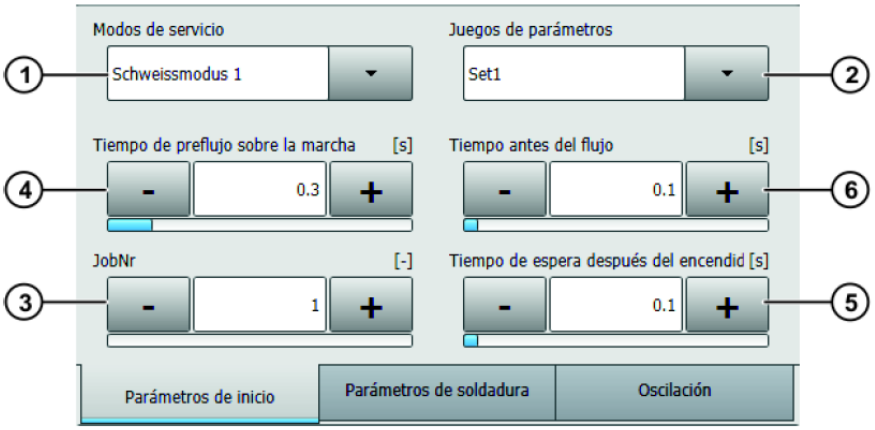


Figura. 56 *Ventana Parámetros ArcOn[53]*

1) Modos de servicio son valores que no se deberán cambiar
2) Juegos de parámetros son valores que no se deberán cambiar
3) Job se trata de las distintas configuraciones existentes en la soldadura con parámetros como amperaje, velocidad del alambre, etc.
4) Tiempo antes de flujo Tiempo de preflujo sobre la marcha es el tiempo antes del inicio de la soldadura (ARC ON) en el que ya fluye gas.
5) Tiempo de espera después del encendido es el tiempo de espera desde el encendido del arco eléctrico hasta el inicio del movimiento.
6) Tiempo de preflujo es el tiempo antes de encender el arco eléctrico durante el que fluye gas.

3.6.5. *VENTANA PARÁMETROS DE SOLDADURA*

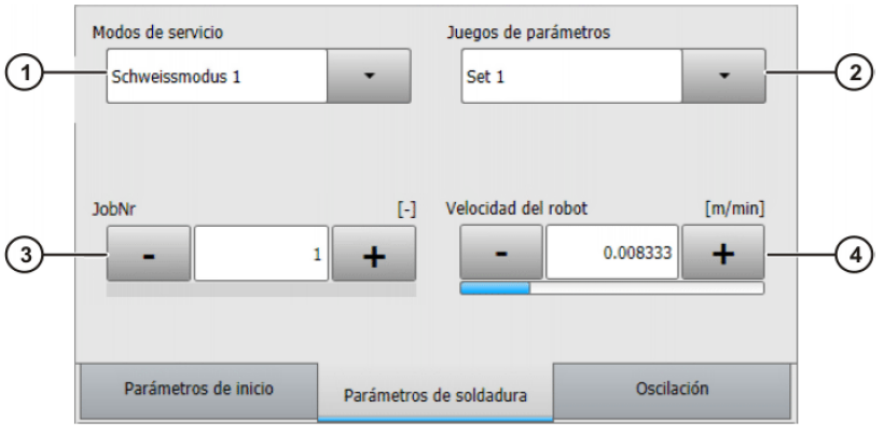


Figura. 57 *Parámteros de Soldadura[53]*

1) Modos de servicio son valores que no se deberán cambiar.
2) Juegos de parámetros son valores que no se deberán cambiar.
3) Job se trata de las distintas configuraciones existentes en la soldadura con parámetros como amperaje, velocidad del alambre, etc.
4) Velocidad del robot al momento de soldadura.

3.6.6. *VENTANA OSCILACIÓN*

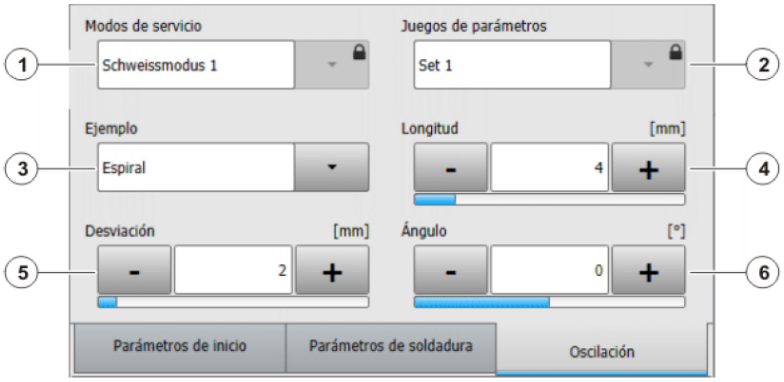


Figura. 58 *Parámetros de Oscilación[53]*

Esta ventana aparece si se elige un tipo de oscilación en el cordón

1) Modos de servicio son valores que no se deberán cambiar.
2) Juegos de parámetros son valores que no se deberán cambiar.
3) Seleccionar el tipo de oscilación.
4) Longitud de oscilación es decir amplitud.
5) Amplitud de oscilación Desviación lateral es la altura del tipo de oscilación.
6) Ángulo es el giro del plano de oscilación.

3.6.7. VENTANA ARCOFF

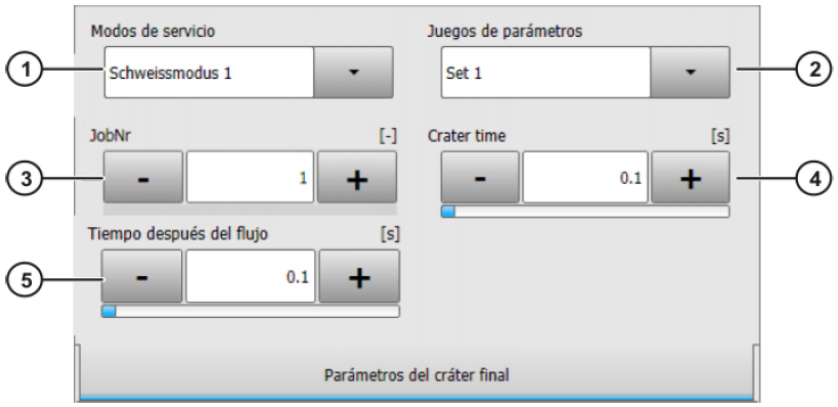


Figura. 59 *Parámetros ArcOff[53]*

1) Modos de servicio son valores que no se deberán cambiar.
2) Juegos de parámetros son valores que no se deberán cambiar.
3) Job se trata de las distintas configuraciones existentes en la soldadura con parámetros como amperaje, velocidad del alambre, etc.
4) Tiempo de cráter final es el tiempo durante el que el robot permanece parado en el punto de destino de la instrucción ARC OFF.
5) Tiempo de retardo al desconectar el gas.

Modificando todos estos valores y se logró tener la programación final de soldadura. En la figura 60 se observa la programación del archivo .src.

```
ARCON WDAT1 LIN P2 Vel=2 m/s CPDAT1 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P3 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P4 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P5 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P6 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P7 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P8 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P9 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P10 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P11 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P12 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P13 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P14 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P15 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P16 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P17 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCOFF WDAT1 LIN P18 CPDAT2 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
```

Figura. 60 *Programación final de Soldadura*

En la figura 61 se aprecia las declaraciones hechas en el archivo .dat.

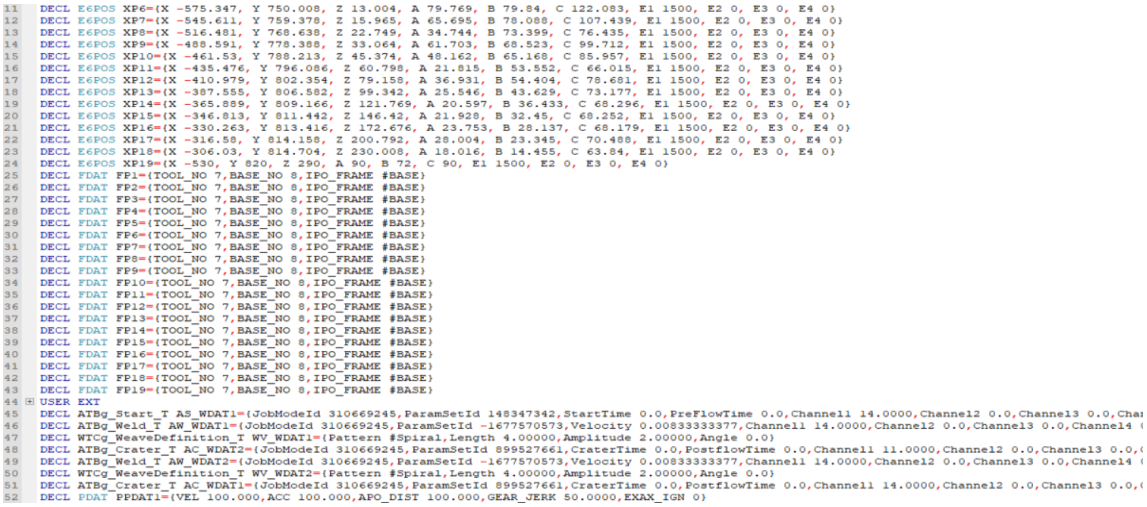
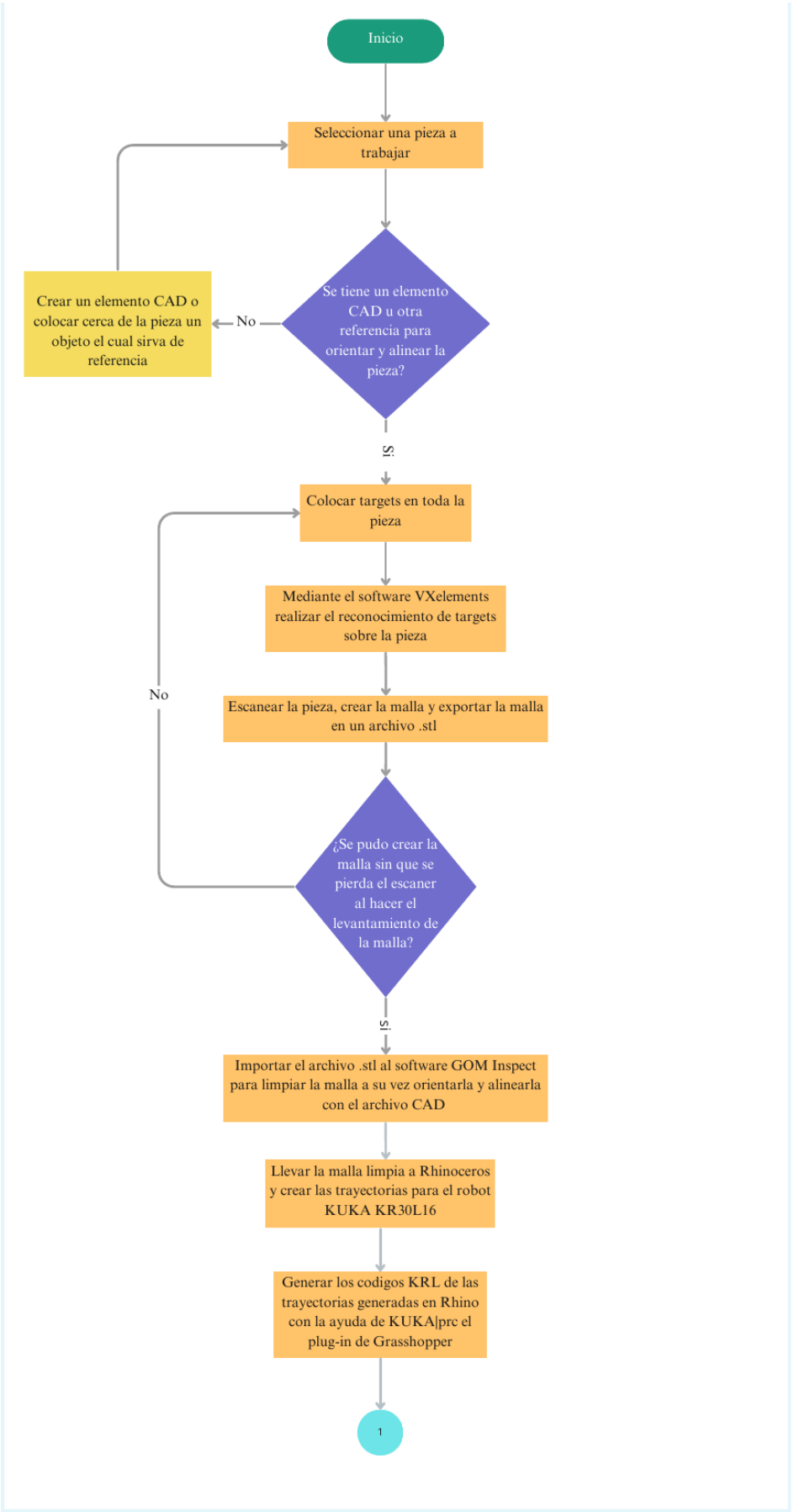
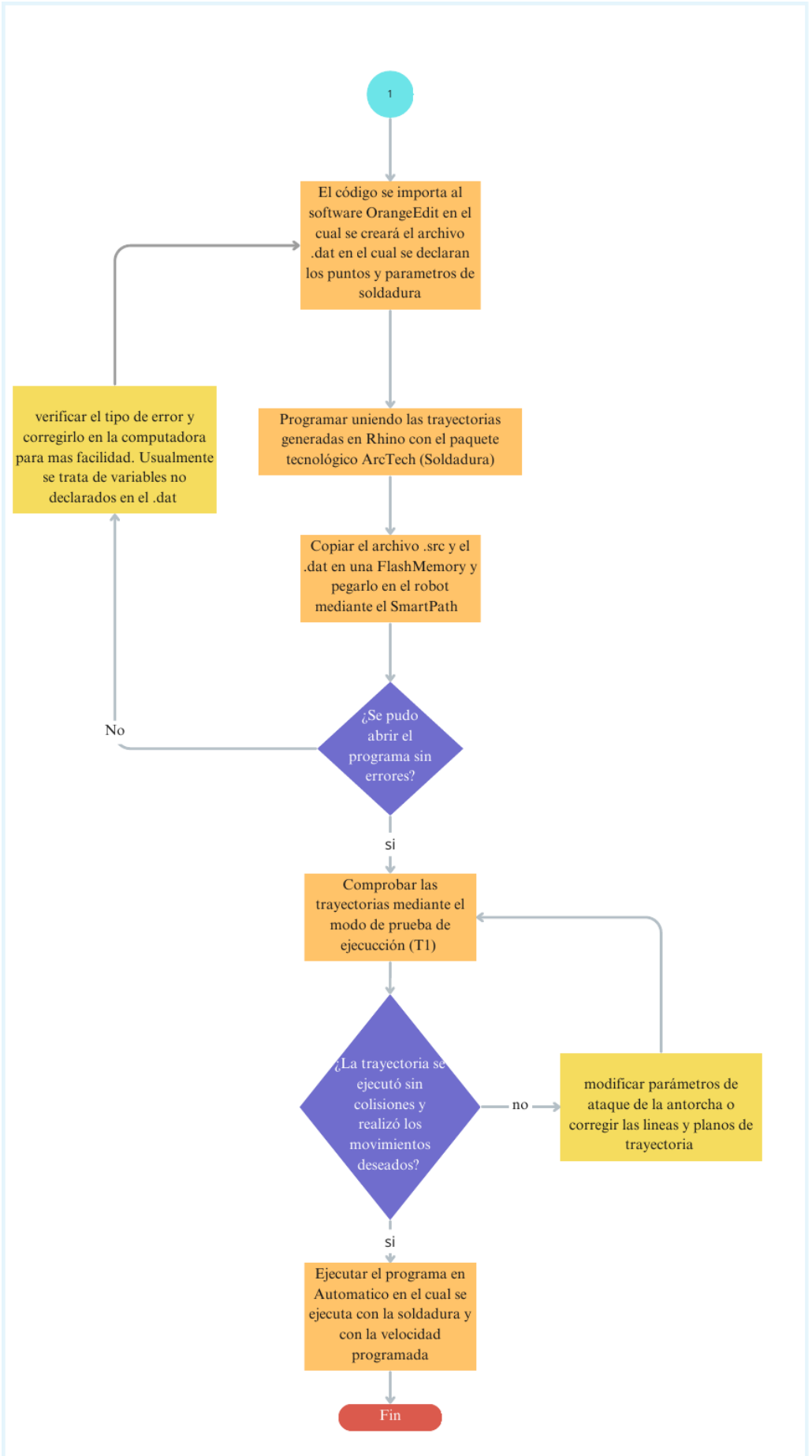


Figura. 61 *Archivo .dat final*

Los dos documentos tanto el .src como él .dat se lo deben pasar a una FlashMemory para así cargarlo en el robot mediante el SmartPath.

3.7. DIAGRAMA DE FLUJO




1
El código se importa al software OrangeEdit en el cual se creará el archivo .dat en el cual se declaran los puntos y parametros de soldadura
Programar uniendo las trayectorias generadas en Rhino con el paquete tecnológico ArcTech (Soldadura)
Copiar el archivo .src y el .dat en una FlashMemory y pegarlo en el robot mediante el SmartPath
¿Se pudo abrir el programa sin errores?
No
verificar el tipo de error y corregirlo en la computadora para mas facilidad. Usualmente se trata de variables no declarados en el .dat
si
Comprobar las trayectorias mediante el modo de prueba de ejecucción (T1)
¿La trayectoria se ejecutó sin colisiones y realizó los movimientos deseados?
no
modificar parámetros de ataque de la antorcha o corregir las lineas y planos de trayectoria
si
Ejecutar el programa en Automatico en el cual se ejecuta con la soldadura y con la velocidad programada
Fin

CAPÍTULO IV

PRUEBAS Y RESULTADOS

Se puso como objetivo el seguir las líneas de la mesa giratoria. Al referirse como líneas de la mesa giratoria se habla a las ranuras que tiene dicha mesa las cuales tienen como punto de inicio el centro y como final el perímetro de la mesa. Esto fue útil ya que como se mencionó el punto de origen de la base es el centro de la mesa giratoria. Para eso se comenzó por dibujar un círculo el cual posteriormente se lo dividió con dos líneas las cuáles serán las ranuras de la mesa (esto fue dibujado en Rhino manualmente).

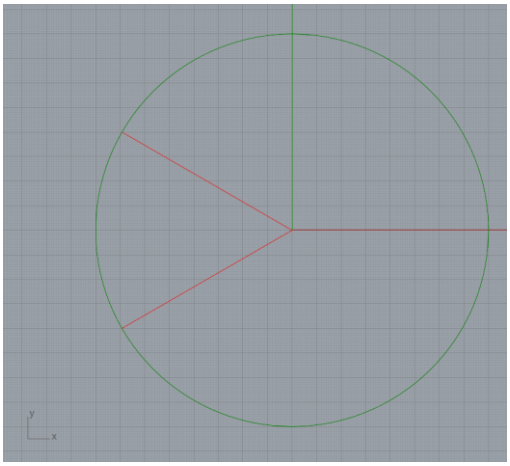


Figura. 62 *Líneas virtuales sobre la mesa*

Como se puede observar se ha realizado las líneas en el X negativo ya que esta parte es la más cercana al robot. Posteriormente se lo eleva en Z 400mm para que el movimiento sea en un área segura y alejada de la superficie como se observa en la figura 63.

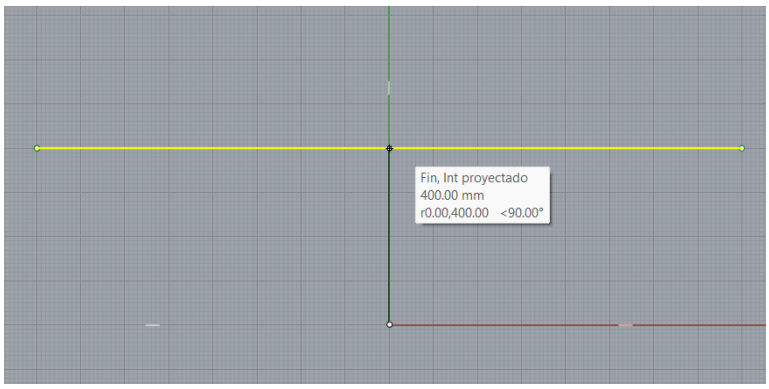


Figura. 63 *Línea movida en Z*

Posteriormente a dicho proceso se comenzó a trabajar en Grasshopper para programar las trayectorias. Y se realizó el siguiente código.

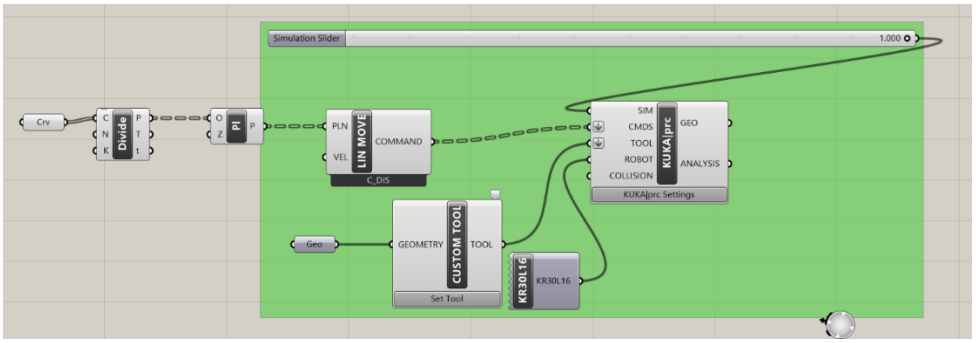


Figura. 64 *Código Grasshopper líneas virtuales*

En el cual se selecciona ambas líneas en el bloque “Crv”, después dividir esta línea en puntos para después generar planos en estos puntos los cuales ayudarán a orientar al robot. Finalmente se conecta a un comando Lineal y se manda al bloque de KUKA|prc el cual generará la trayectoria como se observa en la figura 65.

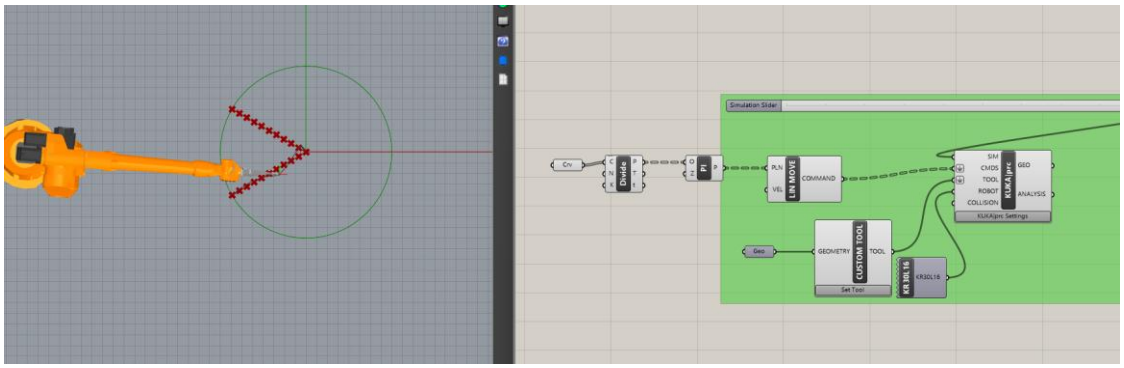


Figura. 65 *Trayectoria de Líneas Virtuales*

El código KRL que se visualiza en la figura 66 es el producto de la programación en Grasshopper.

```
DEF IntentoMesaFin ( )

INI

STARTPOSITION - BASE IS 8, TOOL IS 7, SPEED IS 15%, POSITION IS A1 5,A2 -90,A3 100,A4 5,

LIN SPEED IS 0.25 m/sec, INTERPOLATION SETTINGS IN FOLD

PTP {X 0, Y 0, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0, S 'B110'} C_PTP
LIN {X 0, Y 0, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -69.272, Y 39.997, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -138.544, Y 79.995, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -207.817, Y 119.992, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -277.089, Y 159.99, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -346.361, Y 199.987, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -415.633, Y 239.985, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -484.906, Y 279.982, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -554.178, Y 319.98, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -623.45, Y 359.977, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -692.722, Y 399.975, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 0, Y 0, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -69.277, Y -39.994, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -138.553, Y -79.988, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -207.83, Y -119.982, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -277.107, Y -159.976, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -346.384, Y -199.97, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -415.66, Y -239.964, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -484.937, Y -279.958, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -554.214, Y -319.952, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -623.491, Y -359.946, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X -692.767, Y -399.94, Z 400, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
PTP {A1 5, A2 -90, A3 100, A4 5, A5 -10, A6 -5, E1 1500, E2 0, E3 0, E4 0} C_PTP

END
```

Figura. 66 *Código KRL líneas virtuales*

Se lo cargó en el robot y se lo corrió comprobando así la correcta generación de trayectorias mediante el software Rhino y Grasshopper con la ayuda del plug-in KUKA|prc.

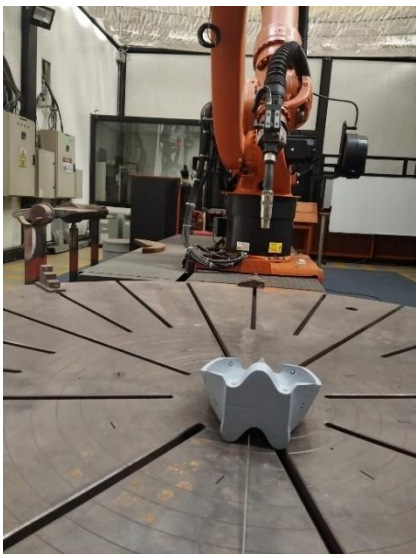


Figura. 67 *Resultado de trayectoria de líneas virtuales*

4.1. PRUEBA SOBRE GEOMETRÍAS ESCANEADAS

Se levantó el escaneo de un rodete de la central Coca Codo Sinclair a escala.

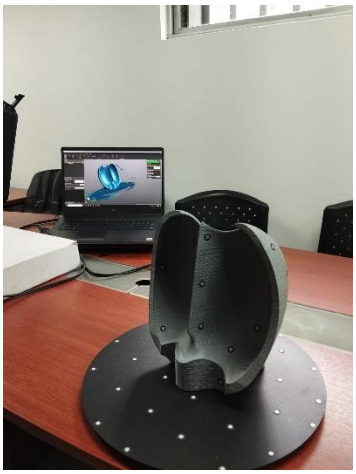


Figura. 68 *Escaneado de Cangilón a escala*

Se procedió a limpiar la malla para tenerla lo más parecido a la original como se la aprecia en la figura 69.

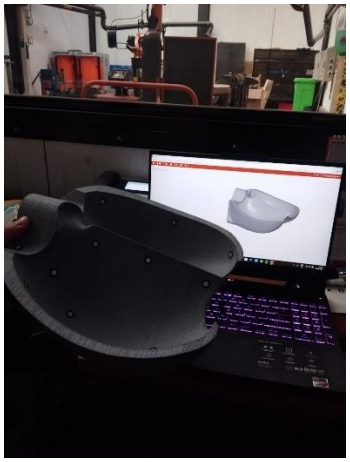


Figura. 69 *Archivo post procesado de Cangilón a escala*

Se lo pasó al software Rhino para trabajar en Grasshopper y realizar las trayectorias sobre la malla como se observa en la figura 70.

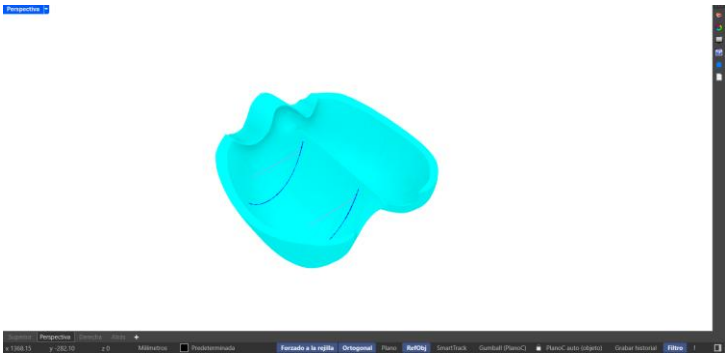


Figura. 70 *Malla Cangilón en Rhino*

A la malla se le dibujaron líneas para después proyectarle sobre la malla y así poder generar trayectorias entre dichas líneas como se aprecia en la figura 71.

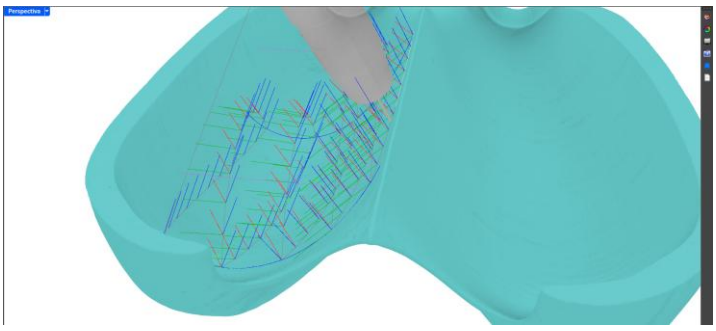


Figura. 71 *Trayectorias sobre malla de Cangilón*

Se creó un programa en el cual a partir de las dos líneas se crearon 4 líneas perpendiculares teniendo como inicio y fin las líneas previamente dibujadas. Y a partir de estas líneas generadas se crearon arcos para moverse entre líneas y así conseguir una trayectoria fluida.

Se midió una base sobre el cangilón mediante el método de tres puntos y se la guardó como en el espacio de Base 14 y se utilizó la herramienta número 7 la cual es la antorcha de soldadura MIG de 22.5º.

Y mediante programación en Grasshopper se logró formar dicha trayectoria la cual se puede observar en la figura 72.

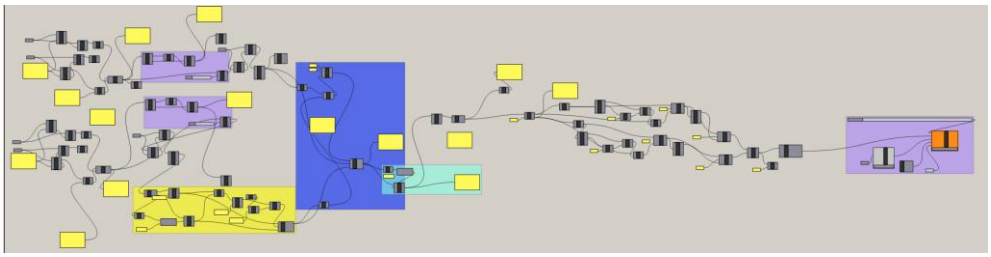


Figura. 72 *Código Grasshopper de trayectoria de Cangilón a escala*

Los códigos KRL que se visualizan en la figura 73 los cuales se generaron mediante Grasshopper y KUKA|prc.

```
DEF PruebaCuchara1 ( )

INI

STARTPOSITION - BASE IS 14, TOOL IS 7, SPEED IS 15%, POSITION IS A1 5,A2 -90,A3 100,A4 5,A5 -10,A6 -5,E1 1500,E2 0,E3 0,E4 0

LIN SPEED IS 0.25 m/sec, INTERPOLATION SETTINGS IN FOLD

PTP {X 54.339, Y 42.898, Z -46.586, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0, S 'B110'} C_PTP
LIN {X 54.339, Y 42.898, Z -46.586, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 65.461, Y 45.279, Z -51.054, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 76.914, Y 46.395, Z -55.165, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 88.642, Y 46.349, Z -58.599, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 100.522, Y 45.417, Z -61.318, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 112.47, Y 43.856, Z -63.371, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 124.433, Y 41.68, Z -64.616, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 136.231, Y 38.621, Z -65.544, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 147.913, Y 35.042, Z -65.567, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 159.451, Y 31.249, Z -64.207, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 170.687, Y 27.06, Z -61.844, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 170.687, Y 27.06, Z -61.844, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 162.69, Y 26.413, Z -49.926, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 152.351, Y 25.576, Z -39.987, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 140.142, Y 24.589, Z -32.48, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 126.621, Y 23.495, Z -27.749, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 112.407, Y 22.345, Z -26.01, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 98.149, Y 21.192, Z -27.342, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 84.499, Y 20.088, Z -31.685, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 72.082, Y 19.083, Z -38.841, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 61.465, Y 18.224, Z -48.481, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 53.133, Y 17.55, Z -60.165, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 53.133, Y 17.55, Z -60.165, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 63.952, Y 15.201, Z -65.118, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 75.154, Y 12.323, Z -68.773, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 86.682, Y 9.549, Z -71.335, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 98.465, Y 7.107, Z -72.871, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 110.366, Y 4.795, Z -73.34, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 122.337, Y 2.847, Z -73.01, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 134.344, Y 1.361, Z -72.082, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 146.295, Y 0.059, Z -70.446, A 125, B 65, C 90, E1 0, E2 0, E3 0, E4 0} C_DIS
```

Figura. 73 *Código KRL de trayectoria de Cangilón a escala*

Se carga en el robot y se lo pone en marcha sin ningún error ni colisión como se observa en la figura 74 comprobando así la generación de trayectorias sobre superficies escaneadas 3D.

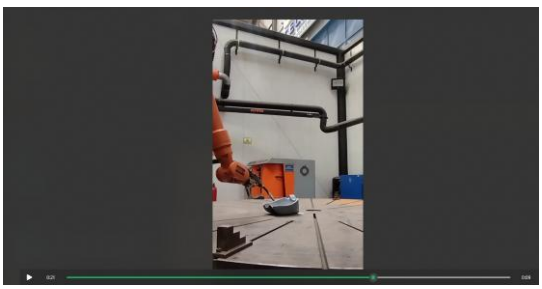


Figura. 74 *Prueba de funcionamiento*

4.2. PRUEBAS DE SOLDADURA ROBOTIZADA

Se genera como primera prueba una trayectoria lineal sobre una superficie plana para comprobar la correcta programación y vinculación entre trayectorias y el paquete tecnológico ArcTech.

A continuación, el procedimiento de generación de trayectoria lineal con código KRL utilizando TOOL 7 y una Base creada sobre la placa metálica todo esto realizado mediante la programación en Grasshopper como se puede apreciar en la figura 75.

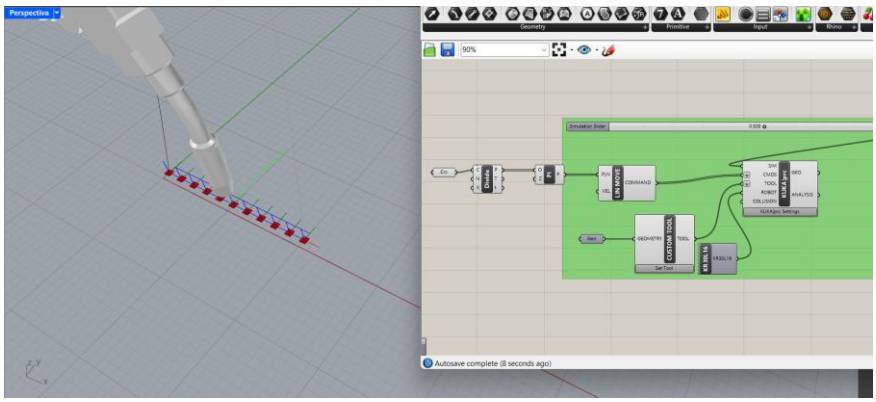


Figura. 75 *Trayectoria Soldadura en Línea*

Una vez se realizó las simulaciones en Grasshopper se procede a generar el código KRL como se puede apreciar en la figura 76.

```
DEF TestSoldar1 ( )

INI

STARTPOSITION - BASE IS 19, TOOL IS 7, SPEED IS 15%, POSITION IS A1 5,A2 -90,A3 100,A4 5,A5 -10,A6 -5,E1 1500,E2 0,E3 0,E4 0

LIN SPEED IS 0.25 m/sec, INTERPOLATION SETTINGS IN FOLD

PTP {X 0, Y 0, Z 10, A 0, B 90, C -0, E1 1500, E2 0, E3 0, E4 0, S 'B110'} C_PTP
LIN {X 0, Y 0, Z 10, A 0, B 90, C -0, E1 1500, E2 0, E3 0, E4 0} C_DIS
LIN {X 210, Y 0, Z 10, A 0, B 90, C -0, E1 1500, E2 0, E3 0, E4 0} C_DIS
LIN {X 420, Y 0, Z 10, A 0, B 90, C -0, E1 1500, E2 0, E3 0, E4 0} C_DIS
LIN {X 630, Y 0, Z 10, A 0, B 90, C -0, E1 1500, E2 0, E3 0, E4 0} C_DIS
LIN {X 840, Y 0, Z 10, A 0, B 90, C -0, E1 1500, E2 0, E3 0, E4 0} C_DIS
PTP {A1 5, A2 -90, A3 100, A4 5, A5 -10, A6 -5, E1 1500, E2 0, E3 0, E4 0} C_PTP

END
```

Figura. 76 *Código KRL de trayectoria de soldadura en línea*

Se le creó un archivo .dat llamado TestSoldar1 que se aprecia en la figura 77 para así conectar los dos archivos. En este archivo .dat se declararon los puntos y los parámetros de soldadura a su vez se redujo el valor en Z para que tenga mejor contacto con la superficie.

```
DEFDAT SoldarLinea PUBLIC
EXTERNAL DECLARATIONS
BASISTECH EXT
DECL E6POS P1 = {X 0, Y 0, Z 4.999, A 0, B 90, C -0, E1 1500, E2 0, E3 0, E4 0}
DECL E6POS P2 = {X 85, Y 0, Z 4.999, A 0, B 90, C -0, E1 1500, E2 0, E3 0, E4 0}
DECL E6POS P3 = {X 170, Y 0, Z 4.999, A 0, B 90, C -0, E1 1500, E2 0, E3 0, E4 0}
DECL E6POS P4 = {X 255, Y 0, Z 4.999, A 0, B 90, C -0, E1 1500, E2 0, E3 0, E4 0}
DECL E6POS P5 = {X 340, Y 0, Z 4.999, A 0, B 90, C -0, E1 1500, E2 0, E3 0, E4 0}
USER EXT
DECL ATBg_Start_T AS_WDAT1={JobModeId 310669245,ParamSetId 148347342,StartTime 0.0,PreFlowTime 0.0,Channel1 10.0000,Channel2 0.0,Channel3 0.0,Channel4
DECL ATBg_Weld_T AW_WDAT1={JobModeId 310669245,ParamSetId -1677570573,Velocity 0.00833333377,Channel1 10.0000,Channel2 0.0,Channel3 0.0,Channel4 0.0,Ch
DECL WTCg_WeaveDefinition_T WV_WDAT1={Pattern #None,Length 4.00000,Amplitude 2.00000,Angle 0.0}
DECL ATBg_Crater_T AC_WDAT2={JobModeId 310669245,ParamSetId 899527661,CraterTime 0.0,PostflowTime 0.0,Channel1 10.0000,Channel2 0.0,Channel3 0.0,Channe
DECL BASIS_SUGG_T LAST_BASIS={POINT1[] "P2",POINT2[] "P2",CP_PARAMS[] "WDAT4",PTP_PARAMS[] "PDAT0",CONT[] " ",CP_VEL[] "2.0",PTP_VEL[] "100"}
DECL FDAT FP1={TOOL_NO 7,BASE_NO 19,IPO_FRAME #BASE}
DECL LDAT LWDAT3={VEL 0.5,ACC 100,APO_DIST 100,APO_FAC 50,ORI_TYP #VAR}
DECL FDAT FP2={TOOL_NO 7,BASE_NO 19,IPO_FRAME #BASE}
DECL FDAT FP3={TOOL_NO 7,BASE_NO 19,IPO_FRAME #BASE}
DECL FDAT FP4={TOOL_NO 7,BASE_NO 19,IPO_FRAME #BASE}
DECL LDAT LWDAT4={VEL 2.0,ACC 100,APO_DIST 100,APO_FAC 50,ORI_TYP #VAR}
DECL LDAT LCPDAT3={ACC 100.000,APO_DIST 100.000,APO_FAC 50.0000,AXIS_VEL 100.000,AXIS_ACC 100.000,ORI_TYP #VAR,CIRC_TYP #BASE,JERK_FAC 50.0000,GEAR_JERK
DECL LDAT LCPDAT1={VEL 2.00000,ACC 100.000,APO_DIST 100.000,APO_FAC 50.0000,AXIS_VEL 100.000,AXIS_ACC 100.000,ORI_TYP #VAR,CIRC_TYP #BASE,JERK_FAC 50.0
DECL PDAT PPDAT1={ACC 100.000,APO_DIST 100.000,APO_MODE #CPTP,GEAR_JERK 50.0000}
DECL E6POS XP1 = {X 0, Y 0, Z 4.999, A 0, B 90, C -0, E1 1500, E2 0, E3 0, E4 0}
DECL E6POS XP2 = {X 85, Y 0, Z 4.999, A 0, B 90, C -0, E1 1500, E2 0, E3 0, E4 0}
DECL E6POS XP3 = {X 170, Y 0, Z 4.999, A 0, B 90, C -0, E1 1500, E2 0, E3 0, E4 0}
DECL E6POS XP4 = {X 255, Y 0, Z 4.999, A 0, B 90, C -0, E1 1500, E2 0, E3 0, E4 0}
DECL E6POS XP5 = {X 340, Y 0, Z 4.999, A 0, B 90, C -0, E1 1500, E2 0, E3 0, E4 0}
DECL MODULEPARAM_T LAST_TP_PARAMS={PARAMS[] "Kuka.MoveDataPtpName=PDAT1; Kuka.VelocityPtp=100; Kuka.PointName=P3; Kuka.BlendingEnabled=False; Kuka.Fram
ENDDAT
```

Figura. 77 *Archivo .dat Soldaura en línea*

Consiguiendo una soldadura sin fallas y un cordón continuo sin apagarse como se pueden observar en figura 78 que es el resultado obtenido. Consiguiendo así vincular la trayectoria generada en Grasshopper y el paquete tecnológico de soldadura ArcTech.

Figura. 78 *Cordón de soldadura*

4.3. SOLDADURA NO LINEAL Y CON VARIACIÓN DE AMPERAJE

Se generó trayectorias en “S” para comprobar que tal se comportaba el soldar en trayectorias curvas es por eso que se dibujaron unas curvas en el software Rhino y se hizo la respectiva programación y simulación del movimiento como se observa en la figura 79.

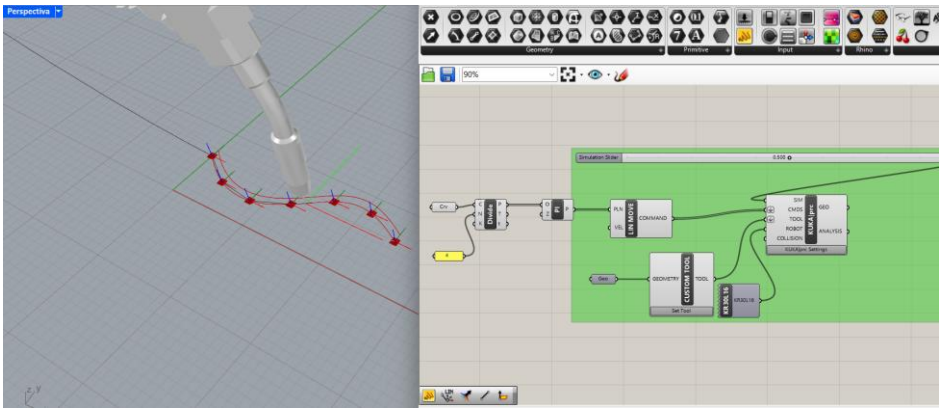


Figura. 79 *Código Grasshopper de trayectoria soldadura en S*

Ya comprobada la trayectoria mediante la simulación se procede a generar el códido KRL que se observa en la figura 80.

```
INI

STARTPOSITION - BASE IS 8, TOOL IS 7, SPEED IS 15%, POSITION IS A1 5,A2 -90,A3 100,A4 5,A5 -10,A6 -5,E1 1500,E2 0,E3 0,E4 0

LIN SPEED IS 0.25 m/sec, INTERPOLATION SETTINGS IN FOLD

PTP {X 0, Y 60, Z 0, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0, S 'B110'} C_PTP
LIN {X 0, Y 60, Z 0, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 38.564, Y 35.28, Z 0, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 83.577, Y 29.495, Z 0, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 125.614, Y 47.329, Z 0, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 163.912, Y 72.756, Z 0, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 208.298, Y 79.013, Z 0, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
LIN {X 250, Y 60, Z 0, A 0, B 90, C -0, E1 0, E2 0, E3 0, E4 0} C_DIS
PTP {A1 5, A2 -90, A3 100, A4 5, A5 -10, A6 -5, E1 1500, E2 0, E3 0, E4 0} C_PTP

END
```

Figura. 80 *Código KRL de trayectoria de soldadura en S*

A su vez en esta prueba se buscó variar el amperaje a lo largo de la curva con el objetivo de simular superficies tridimensionales ya que cuando se tiene superficies tridimensionales se va variando el amperaje dependiendo si se trabaja en bajada, plano o en subida. Al variar amperaje se busca corregir el problema de goteo al soldar.

Para esto se generaron distintos WDAT los cuales también se cambiaron los JOBs es decir el grupo de parámetros guardados en una variable los cuales se generan directamente en la soldadura y la configuración fue la siguiente:

- WDAT1 con JOB14 con un amperaje de 181A se usó para ArcOn
- WDAT2 con JOB11 con un amperaje de 278Ase usó para ArcOff
- WDAT3 con JOB10 con un amperaje de 251A se usó para ArcSwitch
- WDAT4 con JOB15 con un amperaje de 350A se usó para ArcSwitch
- WDAT5 con JOB14 con un amperaje de 278A se usó para ArcSwitch

Obteniendo así el archivo .src final que se aprecia en la figura 81 con la variación de comandos ArcTech.

```
INI

STARTPOSITION - BASE IS 19, TOOL IS 7, SPEED IS 15%, POSITION IS A1 5,A2 -90,A3 100,A4 5,A5 -10,A6 -5,E1 1500,E2 0,E3 0,E4 0

LIN SPEED IS 0.25 m/sec, INTERPOLATION SETTINGS IN FOLD

PTP {X 0, Y 60, Z 0, A 0, B 90, C -0, E1 1500, E2 0, E3 0, E4 0, S 'B110'} C_PTP
ARCON WDAT1 LIN P1 Vel=2 m/s CPDAT1 Tool[7]:MASTERCAM22 Base[19]:pruebaSoldadura
ARCSWI WDAT3 LIN P2 CPDAT4 Tool[7]:MASTERCAM22 Base[19]:pruebaSoldadura
ARCSWI WDAT3 LIN P3 CPDAT4 Tool[7]:MASTERCAM22 Base[19]:pruebaSoldadura
ARCSWI WDAT4 LIN P4 CPDAT4 Tool[7]:MASTERCAM22 Base[19]:pruebaSoldadura
ARCSWI WDAT4 LIN P5 CPDAT4 Tool[7]:MASTERCAM22 Base[19]:pruebaSoldadura
ARCSWI WDAT5 LIN P6 CPDAT4 Tool[7]:MASTERCAM22 Base[19]:pruebaSoldadura
ARCOFF WDAT2 LIN P7 CPDAT2 Tool[7]:TIP Base[19]
PTP {A1 5, A2 -90, A3 100, A4 5, A5 -10, A6 -5, E1 1500, E2 0, E3 0, E4 0} C_PTP

END
```

Figura. 81 *Código KRL implementado tecnología ArcTech*

Teniendo un resultado exitoso. En la figura 82 se aprecia el resultado con la variación de amperaje a lo largo del cordón y todo esto sin que se apagase el arco.

Figura. 82 *Resultado de soldadura en S*

4.4. PRUEBA SOBRE SUPERFICIE TRIDIMENSIONAL

De lo mencionado en el capítulo 3 aquí se llevó a cabo la prueba en la superficie tridimensional teniendo como parámetro de base la Base 8 y como herramienta TOOL 7. Se realizaron líneas sobre la malla en el programa Rhino que serán las trayectorias de soldadura como se aprecia en la figura 84.

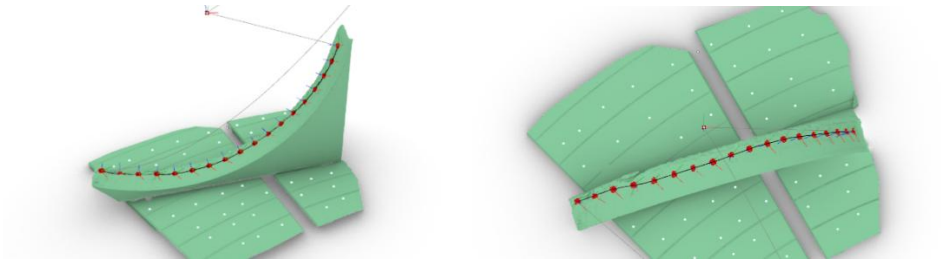


Figura. 84 *Trayectoria en pieza de superficie curva*

También se colocó como WDAT1 el JOB14 con un amperaje de 181A puesto que dicha configuración fue seleccionada para ArcOn y ArcOff debido a varios criterios analizados con los especialistas soldadores. Para el resto de trayectoria se utilizó el WDAT2 con JOB13 el cual permitiría ver el cambio de amperaje al inicio y final. Dicho cambio entre WDAT se aprecia en la figura 85. A esto se agregó que la Oscilación del cordón sea Espiral con una desviación de 4mm y una amplitud de 2mm para así también comprobar que todas las funciones que sean necesarias trabajen de manera correcta.

```
STARTPOSITION - BASE IS 8, TOOL IS 7, SPEED IS 15%, POSITION IS A1 5,A2 -90,A3 100,A4 5,A5 -10,A6 -5,E1 1500,E2 0,E3 0,E4 0

LIN SPEED IS 0.25 m/sec, INTERPOLATION SETTINGS IN FOLD

PTP {X -719.193, Y 709.12, Z 45.711, A 159.721, B 69.448, C -158.329, E1 1500, E2 0, E3 0, E4 0, S 'B110'} C_PTP

LIN XP2 C_VEL

ARCON WDAT1 LIN P2 Vel=2 m/s CPDAT1 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura

ARCSWI WDAT2 LIN P3 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P4 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P5 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P6 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P7 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P8 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P9 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P10 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P11 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P12 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P13 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P14 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P15 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P16 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura
ARCSWI WDAT2 LIN P17 CPDAT4 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura

ARCOFF WDAT1 LIN P18 CPDAT2 Tool[7]:MASTERCAM22 Base[8]:pruebaSoldadura

LIN XP19 C_DIS
PTP {A1 5, A2 -90, A3 100, A4 5, A5 -10, A6 -5, E1 1500, E2 0, E3 0, E4 0} C_PTP

END
```

Figura. 85 *Código KRL de trayectoria y soldadura en pieza de superficie curva*

En la figura 86 se aprecia el momento en el que se corrió el programa y con la ayuda de un especialista soldador analizar el comportamiento mientras el cordón se encuentra encendido.

Figura. 86 *Proceso de soldadura en superficie curva*

Como se visualiza en las figura 87 se logró soldar sobre la superficie escaneada y sobre todo se puede apreciar la variación de amperaje al inicio y al final del cordón. Cumpliendo también la trayectoria generada y visualizada en el software. Cumpliendo así la implementación de trayectorias generadas mediante escaneo 3D y vincular con el paquete tecnológico ArcTech.

Figura. 87 *Resultado de soldadura en superficie curva vista lateral*

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

- A partir de la revisión exhaustiva de los manuales técnicos y la investigación realizada, se logró establecer una integración efectiva entre el escáner 3D, el brazo robot KUKA y el entorno de programación. Esta integración permitió garantizar un flujo de trabajo continuo y preciso, asegurando un ajuste geométrico con una desviación inferior a ±0.2 mm en el proceso de alineación de la malla 3D respecto a la pieza real. Dicho nivel de precisión resulta fundamental para aplicaciones de reparación y mantenimiento de componentes críticos, como los álabes de turbinas Pelton, donde un error mínimo podría comprometer el rendimiento hidráulico y la vida útil del componente.

- La implementación del planificador de trayectoria lineal para el brazo robot KUKA demostró ser una solución eficaz y altamente precisa para la recuperación de superficies desgastadas por abrasión. Este planificador permitió definir trayectorias con errores máximos de posicionamiento inferiores a ±0.05 mm, en concordancia con la precisión de repetibilidad del robot. Además, durante las pruebas de simulación y análisis mecánico, no se registraron interrupciones ni desviaciones significativas, asegurando la estabilidad y la calidad del cordón de soldadura aplicado en las superficies reconstruidas.

- Las pruebas de validación en campo confirmaron el correcto funcionamiento del sistema integrado, logrando cordones de soldadura con variaciones dimensionales menores a ±0.5 mm respecto al perfil teórico. Gracias a la alta repetibilidad del brazo robot KUKA KR 30 L16 (±0.05 mm), se obtuvo un control geométrico del cordón superior al 95 % de conformidad con los parámetros definidos en el software de simulación. Esta precisión permitió reducir los tiempos de intervención manual en aproximadamente un 40 %, mejorando la calidad superficial y aumentando la vida útil del componente intervenido. Además, la metodología implementada evidencia un potencial significativo para su aplicación industrial en procesos de mantenimiento y recuperación de piezas con geometrías complejas.

RECOMENDACIONES

- Al momento de realizar el escaneo 3D de componentes para los cuales no se dispone de un modelo CAD previo, se incluya siempre una geometría de referencia conocida. Esta práctica facilita el centrado y la orientación precisa de la pieza durante el procesamiento de la malla, garantizando una correcta alineación en las etapas posteriores de planificación de trayectoria y programación del robot.

- Es fundamental realizar una lectura detallada y comprensión profunda de los manuales técnicos del robot KUKA y del módulo adicional ArcTech antes de iniciar cualquier desarrollo o implementación. Conocer la estructura de programación, las funciones específicas y las limitaciones del sistema permite optimizar la planificación, prevenir errores y aprovechar de forma completa las capacidades del equipo.

- No olvidar la importancia de mantener una actitud abierta y proactiva hacia el aprendizaje de nuevos software y herramientas. La integración de diferentes plataformas y entornos de simulación —como Grasshopper, KUKA|prc, y software de análisis mecánico— contribuye significativamente al desarrollo de soluciones más robustas y versátiles, fomentando el crecimiento técnico y personal del profesional.

- Resulta esencial comprender tanto el flujo de programación en Grasshopper como la lógica y sintaxis del lenguaje KRL (KUKA Robot Language). Esta doble comprensión permite identificar y corregir posibles errores de trayectoria o lógica antes de la carga y ejecución en el robot físico, evitando contratiempos durante las pruebas y asegurando la seguridad del proceso.

- Como trabajo futuro, se recomienda el desarrollo de un intérprete que permita procesar automáticamente los datos contenidos en los archivos de tipo **.**src y convertirlos al formato **.**dat. Esta herramienta facilitaría la gestión y modificación de parámetros específicos de las trayectorias y puntos de trabajo del robot, optimizando el flujo de programación y reduciendo significativamente el tiempo de ajuste manual en el código KRL.

- Se recomienda, como trabajo futuro, la exploración y prueba de distintos tipos de interpoladores y planificadores de trayectoria, ya que la utilización de otros algoritmos podría ofrecer mejores resultados en términos de suavidad de movimiento, precisión en la deposición del material y adaptación a geometrías complejas. Evaluar la implementación de interpoladores no lineales, como los basados en splines o movimientos de tipo circular, permitiría optimizar la calidad del cordón de soldadura y reducir esfuerzos innecesarios en las articulaciones del robot. Esta investigación adicional aportaría mayor versatilidad al sistema y abriría nuevas posibilidades para aplicaciones avanzadas en procesos de reparación y fabricación automatizada.

REFERENCIAS

[1] Gutiérrez Román Jorge Elias, "Mantenimiento de turbinas hidráulicas Pelton y Francis de medianas potencias", Alicia. Consultado: el 29 de junio de 2025. [En línea]. Disponible en: UUNI_ce47b5e5c865312dd25f67dc81d73280

[2] Gonzalo Castillo, "La soldadura robotizada provee una amplia gama de beneficios para los procesos productivos de la industria.", InnovacionDigital360. Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://www.innovaciondigital360.com/industria-4-0/que-es-la-soldadura-robotizada/

[3] Abid Haleem Singh *et al.*, "Exploring the potential of 3D scanning in Industry 4.0: An overview", Sciencedirect. Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://www.sciencedirect.com/science/article/pii/S2666307422000171?via%3Dihub

[4] Tony Hopkins, "Riesgos de salud ocasionados por humos de soldadura y cómo reducirlos.", Nederman. Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://www.nederman.com/es-mx/knowledge-center/welding-and-cancer

[5] Sofia Beguería, "Operación y mantenimiento: Turbina-Generador", ceupe. Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://www.ceupe.com/blog/operacion-y-mantenimiento-de-turbinas-hidraulicas.html

[6] EDS Robotics, "Soldadura robotizada", EDS Robotics.

[7] Shining 3D, "Escaneado 3D para la planificación de trayectorias de robots con la serie SHINING 3D FreeScan", Shining 3D. Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://www.shining3d.com/es/escaneado-3d-para-la-planificación-de-trayectorias-de-robots-con-la-serie-shining-3d-freescan

[8] Repsol, "¿Qué es una central hidráulica o hidroeléctrica?", Repsol. Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://www.repsol.com/es/energia-futuro/futuro-planeta/central-hidraulica/index.cshtml

[9] V. Singh, S. S.-R. and S. E. Reviews, y undefined 2017, "Operation of hydro power plants-a review", *Elsevier*, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://www.sciencedirect.com/science/article/pii/S1364032116309200

[10] Å. K.-F. energy y undefined 2014, "Hydroelectric power", *Elsevier*, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://www.sciencedirect.com/science/article/pii/B9780080994246000211

[11] Enel Green Power, "Turbina hidroeléctrica", Enel Green Power. Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://www.enelgreenpower.com/es/learning-hub/energias-renovables/energia-hidroelectrica/turbina-hidroelectrica

[12] … D. M. Nordeste. U. N. de S. y undefined 2005, "Micro Centrales Hidroeléctricas", *listas.exa.unne.edu.ar*, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: http://listas.exa.unne.edu.ar/fisica/maestria/modulo2/microturbinas/apuntemch.pdf

[13] V. M.-H. of Turbomachinery y undefined 2003, “Hydraulic Turbines”, *taylorfrancis.com*, doi: 10.1201/9780203911990-23/HYDRAULIC-TURBINES-DAHSHINA-MURTY.

[14] R. Arndt, L. C.-E. Conversion, y undefined 2017, “Hydraulic turbines”, *taylorfrancis.com*, doi: 10.1201/9781315374192-12/HYDRAULIC-TURBINES-ROGER-ARNDT-LEONARDO-CHAMORRO.

[15] P. Akademia Baru, M. A. Khattak, N. S. Mohd Ali, N. H. Zainal Abidin, N. S. Azhar, y M. H. Omar, “Common type of turbines in power plant: a review”, *akademiabaru.com*, vol. 3, núm. 1, pp. 77–100, 2016, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://akademiabaru.com/doc/ARASETV3_N1_P77_100.pdf

[16] L. L. Ladero, “Flexibilidad operacional de las turbinas hidráulicas tipo Francis.”, 2020, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://upcommons.upc.edu/handle/2117/189022

[17] A. C.-S. C. UFSC. O. from http://www y undefined 2003, “Turbinas hidráulicas e condutos forçados”, *academia.edu*, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://www.academia.edu/download/55841650/turb-hidr-2003.pdf

[18] M. Alexis López Montes, J. A. Bonilla Porras, y al M. Ing Víctor Palma Valderrama Ing Jesús Gallegos Silva, “TURBINAS HIDRÁULICAS”, *dicyg.fi-c.unam.mx*, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: http://dicyg.fi-c.unam.mx:8080/labhidraulica/manual-de-practicas/hidraulica-de-maquinas/hmp3.pdf

[19] C. Plana, “Turbomáquinas hidráulicas: turbinas hidráulicas, bombas, ventiladores”, 2009, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://books.google.com/books?hl=es&lr=&id=eKZUEAAAQBAJ&oi=fnd&pg=PA1&dq=tipo+de+turbinas+hidraulicas&ots=l4GqXI3luE&sig=WLTlBeoyu1LPX9euOIlkxRI1VTo

[20] Visitación Buendia, “Fundamentos de desgaste”, SlidePlayer.

[21] L. D. Millan, L. Kazatchkov, A. Fedorov, y H. P. De Vecchi, “Estudio De Problemas De Fluctuaciones De Presión En Una Turbina Hidraulica De Tipo Francis.”, *amcaonline.org.ar*, 2003, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://amcaonline.org.ar/ojs/index.php/mc/article/download/834/787

[22] E. A.-I. J. of H. & Technology y undefined 2019, “Cavitation in Hydraulic Turbines.”, *iieta.org*, doi: 10.18280/ijht.370140.

[23] X. Escaler, E. Egusquiza, M. Farhat, … F. A.-M. systems and, y undefined 2006, “Detection of cavitation in hydraulic turbines”, *Elsevier*, vol. 20, pp. 983–1007, 2006, doi: 10.1016/j.ymssp.2004.08.006.

[24] J. G.-C. J. of C. Engineering y undefined 2001, “Hydraulic turbine efficiency”, *cdnsciencepub.com*, vol. 28, núm. 2, pp. 238–253, 2001, doi: 10.1139/L00-102.

[25] L. Jeffus, “Soldadura. Principios y aplicaciones”, 2009, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://books.google.com/books?hl=es&lr=&id=rHynAxzh0iEC&oi=fnd&pg=PP1&dq=soldadura+&ots=byLN9odXhS&sig=RSDVTp0YzcSe5voHP5LlwYbqyrw

[26] P. M. Solá, “Soldadura industrial: clases y aplicaciones”, 1992, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://books.google.com/books?hl=es&lr=&id=KoEH9EkR48gC&oi=fnd&pg=PA7&dq=soldadura+&ots=34Kv9xB9et&sig=r114sLaZwYWW_1o1FhkUDeCgvA4

[27] P. Rodriguez, “Manual de soldadura”, 2013, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://books.google.com/books?hl=es&lr=&id=EyiXDwAAQBAJ&oi=fnd&pg=PA7&dq=soldadura+&ots=ZANF_oEPT4&sig=a706FhwpQAhSVRoHGlMCg6CQKNg

[28] C. U.-M. Actual y undefined 2010, “Soldadura GMAW-MIG/MAG”, *academia.edu*, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://www.academia.edu/download/56888458/procesos_soldadura.pdf

[29] F. C. Torres, “Control de Calidad en los Procesos de Soldadura FCAW-SMAW”, 2014, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://alicia.concytec.gob.pe/vufind/Record/UNSA_b67807a6b36e7763b53c861079dc1e79

[30] D. Cuesta, J. H.- Tekhnê, y undefined 2016, “Diseño y construcción de un brazo robótico de 6 GDL”, *revistas.udistrital.edu.co*, vol. 13, núm. 1, pp. 73–82, 2016, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://revistas.udistrital.edu.co/index.php/tekhne/article/view/11427

[31] M. Molina Cárdenas, P. Pedroza Barrios, K. Mauricio Gaitán Moreno, J. Fernando Salgado Arismendy, y M. Camila Ordóñez Ávila, “Diseño y Construcción del Prototipo de un Brazo Robótico con Tres Grados de Libertad, como Objeto de Estudio”, *dialnet.unirioja.es*, vol. 10, núm. 18, pp. 1909–2458, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://dialnet.unirioja.es/servlet/articulo?codigo=5478783

[32] R. Kelly, R. Carelli, C. S.-V. C. M. de, y undefined 2004, “Sobre control cinemático de impedancia en robots industriales”, *ebanov.inaut.unsj.edu.ar*, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: http://ebanov.inaut.unsj.edu.ar/publicaciones/Ca1658_04.pdf

[33] G. Labanda, “Desarrollo de un sistema interactivo de gestión integral de parámetros de influencia en procedimientos de soldadura robotizada para procesos de soldeo por arco”, 2005, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://dialnet.unirioja.es/servlet/dctes?codigo=2518

[34] J. R. Villar, “Diseño, programación y simulación de un sistema para soldadura robotizada”, 2021, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://riunet.upv.es/handle/10251/173730

[35] H. Mühe, A. Angerer, A. Hoffmann, y W. Reif, “On reverse-engineering the KUKA Robot Language”, sep. 2010, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://arxiv.org/pdf/1009.5004

[36] Y. Tong, … Z. X.-C. on C. D. and R., y undefined 2019, “Application of robotic arm technology in intelligent construction”, *Springer*, pp. 331–345, 2020, doi: 10.1007/978-981-13-8153-9_30.

[37] M. Ríos, G. G.-R. B. Redipe, y undefined 2022, “Desarrollo de habilidades técnicas en ingeniería de software aplicando ingeniería inversa.”, *revista.redipe.org*, Consultado: el

29 de junio de 2025. [En línea]. Disponible en: http://revista.redipe.org/index.php/1/article/view/1661

[38] M. B. Rodriguez, "Ingeniería inversa aplicada: metodología y aplicaciones industriales", 2011, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: https://repository.eafit.edu.co/bitstream/handle/10784/9080/Mario_BetancurRodriguez_2011.pdf

[39] M. T.-R. T. energía y undefined 2019, "Plan de control y aseguramiento de la calidad para la recuperación de un rodete de turbina Francis de una central hidroeléctrica", *scielo.senescyt.gob.ec*, Consultado: el 29 de junio de 2025. [En línea]. Disponible en: http://scielo.senescyt.gob.ec/scielo.php?pid=S2602-84922019000100057&script=sci_arttext

[40] CELEC EP, "El Centro de Investigación y Recuperación de Turbinas (CIRT) entrega sus servicios a la empresa privada". [En línea]. Disponible en: https://www.celec.gob.ec/noticias/el-centro-de-investigacion-y-recuperacion-de-turbinas-cirt-entrega-sus-servicios-a-la-empresa-privada/

[41] "Guía Básica - Documentación RoboDK". Consultado: el 20 de julio de 2025. [En línea]. Disponible en: https://robodk.com/doc/es/Basic-Guide.html

[42] "Conoce RoboDK". Consultado: el 20 de julio de 2025. [En línea]. Disponible en: https://www.dypromac.com/noticias/conoce-robodk/

[43] "Rhino - Rhino en Arquitectura, Ingeniería y Construcción". Consultado: el 20 de julio de 2025. [En línea]. Disponible en: https://www.rhino3d.com/es/for/architecture/

[44] "Grasshopper para Rhino: ¿qué es y cómo se utiliza?" Consultado: el 20 de julio de 2025. [En línea]. Disponible en: https://www.echeverrimontes.com/blog/grasshopper-para-rhino-que-es-y-como-se-utiliza

[45] "KUKA PRC y sus aplicaciones. | Makermex | Impresoras 3D". Consultado: el 20 de julio de 2025. [En línea]. Disponible en: http://makermex.com/blog/manufactura-digital-1/post/kuka-prc-y-sus-aplicaciones-538

[46] "KUKA | prc | Parametric Robot Control for Grasshopper Software". Consultado: el 20 de julio de 2025. [En línea]. Disponible en: https://robotsinarchitecture.com/kuka-prc/

[47] "Robotic Welding | Robotmaster". Consultado: el 20 de julio de 2025. [En línea]. Disponible en: https://www.robotmaster.com/applications/welding/

[48] "Robotmaster | Programación offline para robots industriales » Camcut". Consultado: el 20 de julio de 2025. [En línea]. Disponible en: https://www.camcut.fi/es/softwares/robotmaster/

[49] "Robotmaster - TECICAM". Consultado: el 20 de julio de 2025. [En línea]. Disponible en: https://www.tecicam.com/robotmaster/

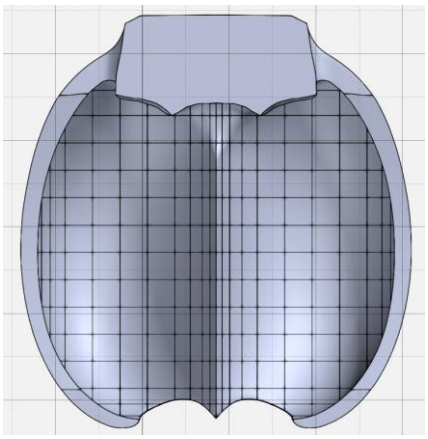
[50] K. Roboter GmbH, "Programación de robots 1 KUKA System Software 8 Documentación para la formación", 2011.

[51] K. Roboter GmbH, "Programación de robots 1 KUKA System Software 8 Documentación para la formación", 2011.

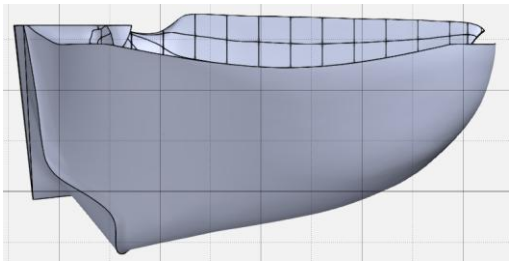
[52] “KUKA System Technology KUKA.ArcTech Basic 1.4 Para KUKA System Software 8.3 Versión: KST ArcTech Basic 1.4 V1”, 2015.

ANEXOS

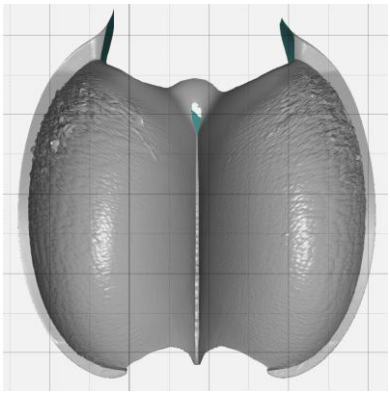
Anexo 1 Elemento nominal cangilón



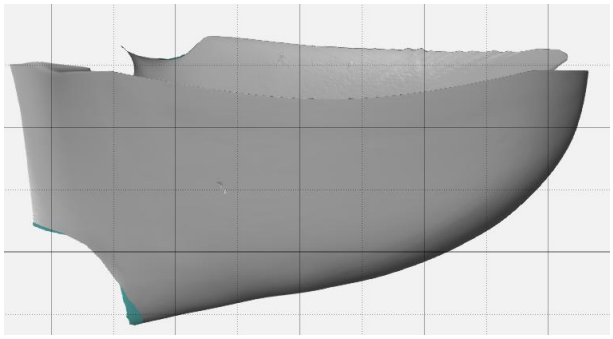
Anexo 2 Cuchilla nominal cangilón



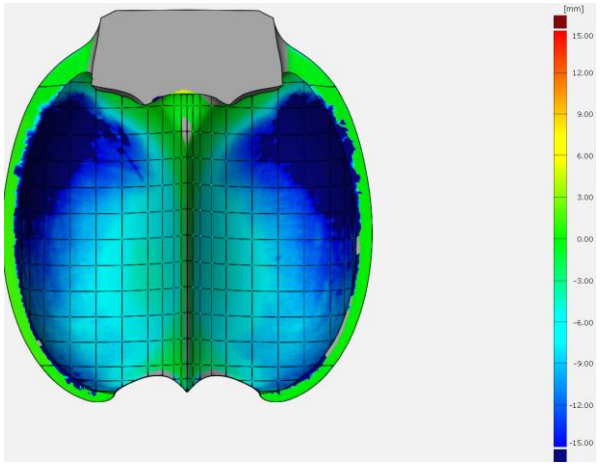
Anexo 3 Malla cangilón cavitado escaneado



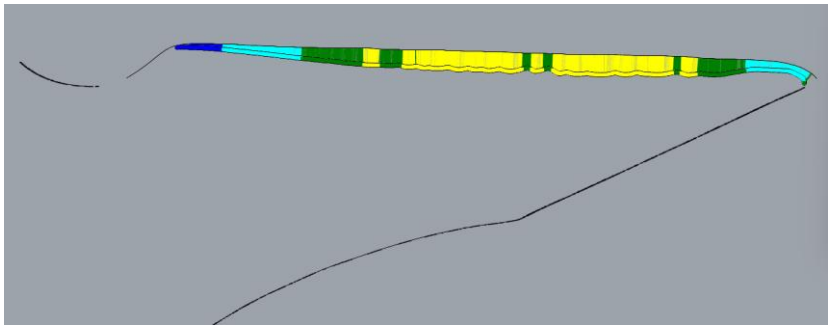
Anexo 4 Cuchilla escaneada cavitada cangilón



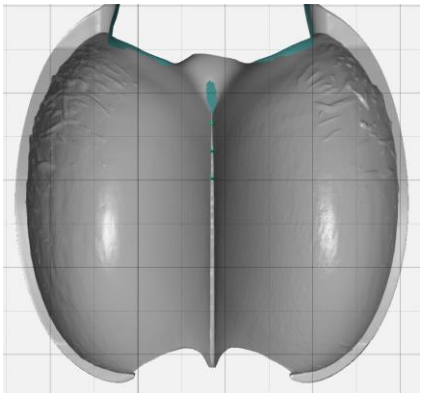
Anexo 5 Comparativa entre nominal (CAD) y real (malla cangilón cavitado)



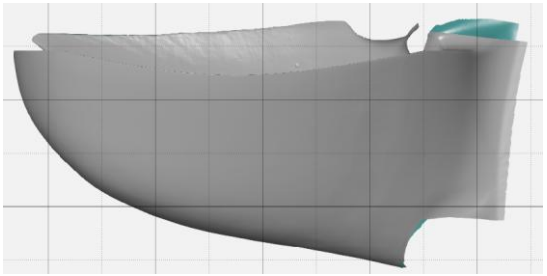
Anexo 6 Comparativa entre cuchilla nominal con la real



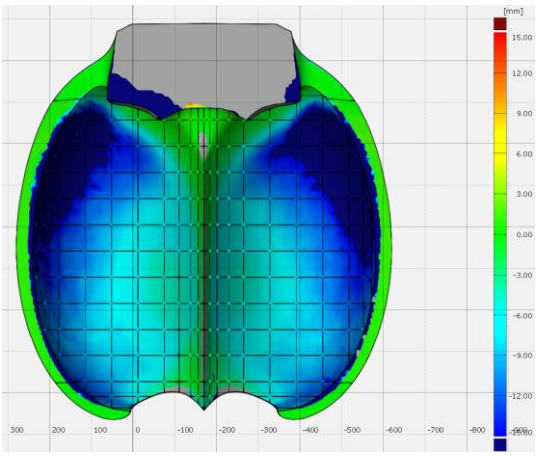
Anexo 7 Malla cangilón pulido escaneado



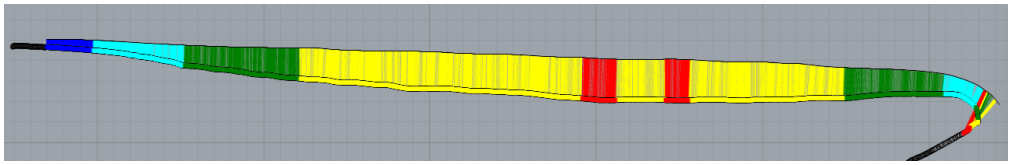
Anexo 8 Malla cuchilla pulida



Anexo 9 Comparativa entre nominal (CAD) y real (malla cangilón pulido)



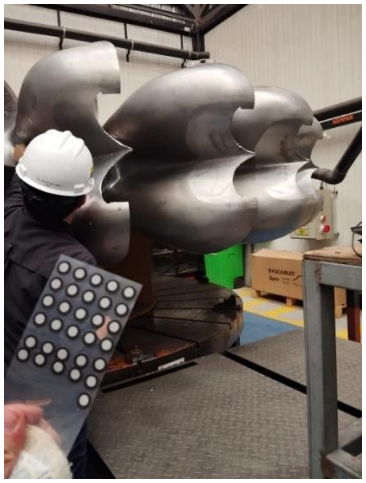
Anexo 10 Comparativa entre cuchilla nominal y cuchilla real



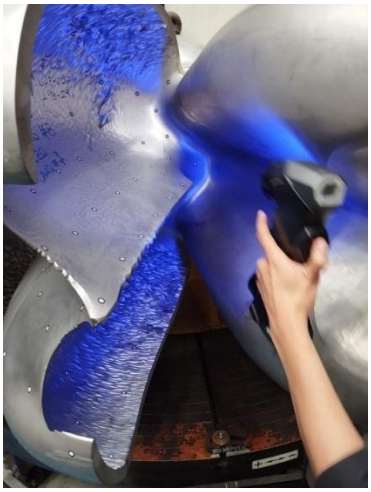
Anexo 11 Cangilón Cavitado

Anexo 12 Cangilón pulido

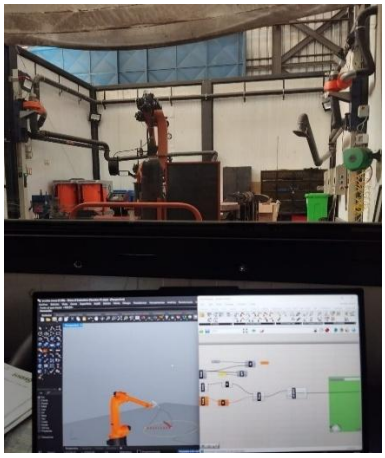
Anexo 13 Colocación de targets en los cangilones a escanear



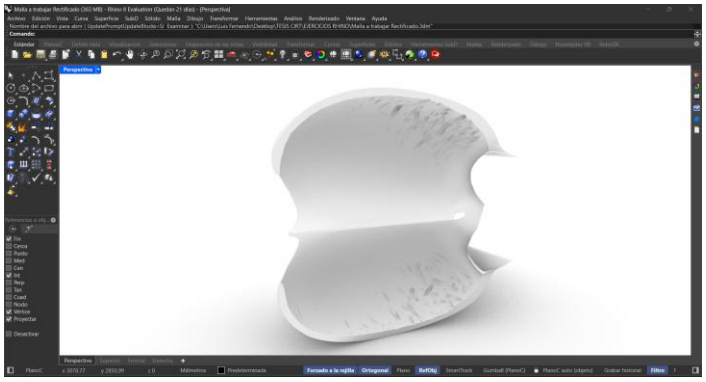
Anexo 14 Escaneo de rodete cavitado



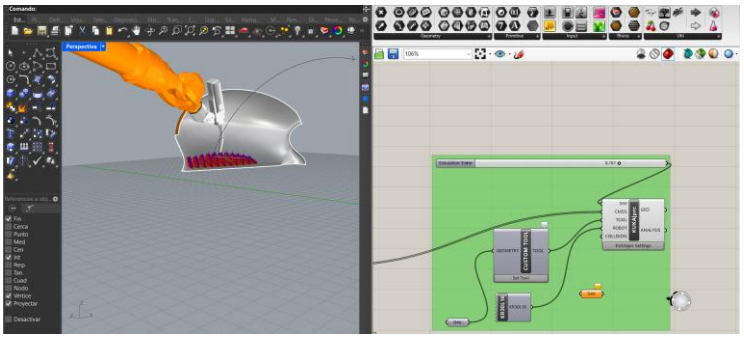
Anexo 15 Pruebas entre simulación y entorno real



Anexo 16 Cangilón escaneado dentro del entorno Rhinoceros



Anexo 17 Simulación de trayectoria sobre cangilón escaneado



Anexo 18 Pruebas en RoboDK