

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
SEDE VIÑA DEL MAR – JOSÉ MIGUEL CARRERA

DISEÑO Y PROPUESTA DE INTEGRACIÓN ENTRE VISIÓN ARTIFICIAL Y ROBOT KUKA
AGILUS

Trabajo de Titulación para optar al Título de
Ingeniero de Ejecución en CONTROL E
INSTRUMENTACIÓN INDUSTRIAL

Alumno:

Moisés Emmanuel Moyano Vega

Profesor Guía:

Martín García Gutiérrez

2025



CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☒ Memoria o trabajo de título; ☐ Tesis de Postgrado;

Título del trabajo: DISEÑO Y PROPUESTA DE INTEGRACIÓN ENTRE VISIÓN ARTIFICIAL Y ROBOT KUKA AGILUS

Nombre del candidato(a): Moisés Moyano Vega

Carrera / Grado: INGENIERÍA DE EJECUCIÓN EN CONTROL E INSTRUMENTACIÓN INDUSTRIAL

Campus: Viña del Mar; Departamento: ELINF

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Martín García Gutierrez, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL

☒ El trabajo **NO contiene información que amerite confidencialidad** y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (embargo) por:

☐ 6 meses; ☐ 12 meses; ☐ 2 años; ☐ 3 años; ☐ 5 años; ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 05-09-2025

; Firma:

Estudiante o Candidato(a):

Fecha: 05-09-2025

; Firma:

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.

RESUMEN

KEYWORDS: BRAZO ROBÓTICO, MATLAB, PROCESAMIENTO DE IMÁGENES, KUKA ETHERNET KRL.

El siguiente proyecto tiene como objetivo explorar la integración de la visión artificial y el procesamiento de imágenes en tiempo real como estímulo de entrada para el accionamiento de un brazo robótico KUKA Agilus KR6 R900, con la finalidad de poder crear nuevas funciones de control sobre la máquina a través de interpretación de colores, además de instancias de aprendizaje para futuros estudiantes con las tecnologías implementadas en la universidad, como resultado dar un paso más al proceso de trabajo colaborativo humano-máquina, aprovechando el uso de los recursos de softwares como MATLAB en el reconocimiento, análisis y procesamiento de datos.

Se utilizará la librería de MatLab “IMAGE ACQUISITION toolbox” para obtener imágenes y procesarlas en tiempo real, de esta manera poder crear un seguimiento de objetivo para proporcionar una coordenada que puede ser interpretada por el controlador KRC4 del Robot, así de esta manera se busca proporcionar movimientos al actuador, el controlador obtendría datos directamente del procesamiento de Matlab a través de un servidor TCP/IP proporcionado por la librería “Kuka Ethernet KRL” y de esta manera acercar un paso más a la interacción de las máquinas y nuestro mundo a través de los datos recopilados y transformados provenientes de nuestro entorno.

INDICE

DEDICATORIA	1
INTRODUCCIÓN	2
CAPITULO 1: PROBLEMÁTICA Y OBJETIVOS.....	4
1 PROBLEMÁTICA Y OBJETIVOS.....	5
1.1 ANTECEDENTES GENERALES.....	5
1.2 UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA.....	6
1.3 KUKA AGILUS KR6 R900.....	7
1.4 MATLAB	8
1.4.1 IMAGE ACQUISITION toolbox	9
1.4.2 Comunicación TCP/IP	10
1.5 PROBLEMÁTICA	11
1.5.1 Descripción del problema	11
1.5.2 Importancia de resolverlo	12
1.6 PROPUESTA	12
1.7 REQUERIMIENTOS	12
1.8 ALTERNATIVAS DE SOLUCIÓN.....	15
1.8.1 Alternativa N°1 “Implementar secuencias a través de KUKA SmartPAD”	15
1.8.2 Alternativa N°2 “Búsqueda de integración del robot KUKA Agilus con visión artificial utilizando los recursos existentes”	16
1.8.3 Alternativa N°3 “Integración del robot KUKA Agilus con visión artificial utilizando los mejores recursos”	16
1.9 EVALUACIÓN DE LAS ALTERNATIVAS PROPUESTAS	17
1.10 OBJETIVOS	18
1.10.1 Objetivo general	19
1.10.2 Objetivos específicos	19
CAPITULO 2: PRUEBAS DE INTEGRACIÓN ENTRE SISTEMAS	20

2	PRUEBAS DE INTEGRACIÓN ENTRE SISTEMAS.....	22
2.1	<i>IDENTIFICACIÓN KUKA AGILUS KR6 R900</i>	22
2.1.1	Controlador KR C4.....	22
2.1.2	Manipulador de 6 ejes	25
2.1.3	Kuka SmartPad	27
2.1.4	Lenguaje de programación KRL	29
2.1.5	Software de programación WorkVisual 6.0.....	30
2.2	<i>COMUNICACIÓN ENTRE CÁMARAS Y MATLAB</i>	32
2.2.1	IMAGE ACQUISITION toolbox	33
2.3	<i>SCRIPT DE PROCESADO DE IMÁGENES EN TIEMPO REAL</i>	35
2.3.1	Captura y conversión	35
2.3.2	Procesado de imágenes	37
2.3.3	Exportar datos desde MATLAB	38
2.4	<i>CREACIÓN DE LA ZONA DE OPERACIÓN</i>	40
2.4.1	Disposición física del área de trabajo	40
2.4.2	Creación del espacio 3D de trabajo en SmartPad	42
2.5	<i>PROGRAMACIÓN DE ROBOT KUKA EN WORKVISUAL 6.0</i>	44
2.5.1	Configuración del software y enlace con controlador KRC4	45
2.5.2	Incorporación de librerías.....	51
2.5.3	Creación de ciclo de trabajo en coordenadas \$BASE	52
2.5.4	Relación de transformación Pixel a [mm] de movimiento	56
2.6	<i>KUKA ETHERNET KRL</i>	60
2.6.1	Comunicación TCP IP con MATLAB	61
2.6.2	Programar XML para obtención y entrega de datos	62
2.6.3	Integrar variables obtenidas a programación KRL.....	63
	CAPITULO 3: EVALUACIÓN DE RECURSOS Y OPCIONES DE MEJORA.....	65
3	EVALUACIÓN DE RECURSOS Y OPCIONES DE MEJORA	64
3.1	<i>RECURSOS MATERIALES</i>	64

3.1.1 Recursos materiales de la universidad	64
3.1.2 Recursos materiales anexos y/o externos	65
3.2 <i>RECURSOS HUMANOS</i>	68
3.3 <i>OPCIONES DE MEJORA</i>	70
CONCLUSIONES	73
BIBLIOGRAFÍA	74

ÍNDICE DE FIGURAS

Figura 1-1. Universidad Técnica Federico Santa María – Sede José Miguel Carrera

Figura 1-2. Brazo robótico KUKA Agilus KR6 R900

Figura 1-3. MATLAB

Figura 1-4. IMAGE ACQUISITION toolbox

Figura 2-1. Esquema controlador KR C4

Figura 2-2. Controlador KR C4

Figura 2-3. Ejes de operación KUKA Agilus R6 900

Figura 2-4. KUKA Agilus R6 900 Laboratorio

Figura 2-5. KUKA SmartPad

Figura 2-6. Instrucción de ejemplo KRL

Figura 2-7. Entorno WorkVisual 6.0

Figura 2-8. Enlace DroidCam Client

Figura 2-9. Enlace Irium Webcam v2.8.1

Figura 2-10. Declaración de variables a dispositivos de video MATLAB

Figura 2-11. Selección de dispositivo IMAGE ACQUISITION toolbox

Figura 2-12. Triángulo de conversión HSV

Figura 2-13. Conversión RGB a HSV

Figura 2-14. Ejemplo de funcionamiento Script MATLAB

Figura 2-15. Prueba de comunicación entre MATLAB y Hercules SETUP

Figura 2-16. Calibración de orientación de Smartphone con APP Inclinator

Figura 2-17. Virtualización de la zona de trabajo

Figura 2-18. Creación de \$BASE de trabajo con asistente en SmartPad

Figura 2-19. Sistemas de coordenadas Robot KUKA Agilus R6 900

Figura 2-20. Requisitos del sistema de WorkVisual 6.0

Figura 2-21. Configuración de red del ordenador

Figura 2-22. Exploración de Proyectos WorkVisual 6.0

Figura 2-23. Interfaz de trabajo WorkVisual 6.0

Figura 2-24. Administrador de catálogos de WorkVisual 6.0

Figura 2-25. Árbol de dispositivos del proyecto en WorkVisual 6.0

Figura 2-26. Árbol de elementos geométricos en WorkVisual 6.0

Figura 2-27. Explorador de archivos del proyecto en WorkVisual 6.0

Figura 2-28. Gestor de paquetes en WorkVisual 6.0

Figura 2-29. Creación del archivo de programación en WorkVisual 6.0

Figura 2-30. Demostración del interpretador interno del Robot

Figura 2-31. Demostración del programa KRL y posición de ejes para punto HOME en SmartPad

Figura 2-32. Escalado y corrección de OFFSET en MATLAB para exportar coordenadas al Robot

Figura 2-33. Ícono de instalación del proyecto

Figura 2-34. Carga del Proyecto a la célula Robot

Figura 2-35. Diagrama del interpretador EKI a KRL

Figura 2-36. Esquema de envío, recepción e interpretación de datos con EKI en KRL

Figura 2-37. Diagrama de roles de comunicación TCP/IP

Figura 2-38. Configuración estándar de archivo XML

Figura 2-39. XML "EXT" para el traspaso de información entre MATLAB y EKI RKL

Figura 3-1. Cotización Samsung S20 Fe

Figura 3-2. Cotización Trípodes

Figura 3-3. Cotización Notebook

Figura 3-4. Cotización cámara industrial Datalogic P20C

Figura 3-5. Cotización paquete KUKA.PLC.mxAutomation 3.3

Figura 3-6. Cotización WorkStation

Figura 3-7. Cotización Monitor 27" para WorkStation

Figura 3-8. Cotización PLC Siemens S7-1500

ÍNDICE DE TABLAS

Tabla 1-1. Requerimientos del proyecto

Tabla 1-2. Tabla de puntuación

Tabla 1-3. Tabla de evaluación de alternativas

Tabla 2-1. Descripción del controlador KR C4

Tabla 2-2. Datos de los ejes del manipulador KR6 R900

Tabla 2-3. Datos técnicos manipulador KUKA KR6 R900

Tabla 2-4. Característica de KUKA SmartPad

Tabla 2-5. Valores HSV a analizar en el Script

Tabla 2-6. Muestreo de coordenadas con SmarPad

Tabla 2-7. Factor de escalado para desplazamiento en \$BASE

Tabla 2-8. Límites de movimiento en \$BASE del Robot

Tabla 2-9. Ajuste de OFFSET y posiciones límites para el manipulador en \$BASE

Tabla 3-1. Recursos de la universidad

Tabla 3-2. Recursos anexos y externos

Tabla 3-3. Recursos humanos utilizados en horas de estudio

Tabla 3-4. Recursos humanos utilizados en el desarrollo del proyecto

Tabla 3-5. Cotización de elementos para opciones de mejora del proyecto

SIGLAS Y SIMBOLOGÍAS

MATLAB:	“MATrix LABoratory”
KRL:	“KUKA Robot Language”
UTFSM:	“Universidad Técnica Federico Santa María”
2D:	“Dos dimensiones”
3D:	“Tres dimensiones”
USB:	“Universal Serial Bus”
Wi-Fi:	“Wireless Fidelity”
KUKA:	“Keller und Knappich Augsburg”
ETH:	“Ethernet”
TCP/IP:	“Transmission Control Protocol/Internet Protocol”
XML:	“Extensible Markup Language”
[mm]:	“milímetros”
RGB:	“Red Green Blue”
HSV:	“Hue Saturation Value” / “Matiz Saturación Valor”

DEDICATORIA

Dedico todo este esfuerzo a mi esposa Kizzy, por su paciencia, apoyo, amor y compromiso con mi desarrollo profesional, a mi amada hija Ignacia que siempre ilumina los días con su creatividad, también a mis padres por su apoyo incondicional frente a las adversidades de la vida.

Fue gracias al apoyo, al sustento emocional y al tiempo que me han brindado lo que hizo posible la realización de este proyecto de titulación. Muchas gracias.

INTRODUCCIÓN

La robótica hoy en día representa un avance trascendental en el último siglo y su integración está cada vez es más presente en la industria y la cotidianeidad de nuestras vidas.

Los comienzos de la robótica datan desde la antigüedad donde las primeras confecciones correspondían a estatuas móviles y/o mecánicas enfocadas en el entretenimiento y lo religioso, posterior a esto comienzan los avances ya en el siglo XIII el inventor y matemático musulmán Al-Jazari diseñó y creó máquinas automáticas que incluían relojes de agua y dispositivos mecánicos que se asemejaban a animales y músicos, con el paso del tiempo en la época del renacimiento aparece el famoso inventor Leonardo da Vinci quien esbozó diseños de máquinas que se asemejan a humanos y animales, aunque no los construyó, sus ideas presentaron las bases para el futuro. El siguiente hito importante corresponde a la revolución industrial, dando el puntapié inicial a la creación de las primeras máquinas automatizadas en fábricas, como telares mecánicos y máquinas de vapor, estos dispositivos no eran robots en el sentido moderno, pero introdujeron la automatización en la producción. Ya en el siglo XX nace el término “robot” acuñado por primera vez en la obra de teatro “R.U.R” (Rossum’s Universal Robots) escrita por Karel Capek en 1920, en esta obra los robots eran seres artificiales contruidos por los humanos, desde este punto los avances fueron más frecuentes ya que en el año 1954 George Devol y Joseph Engelberger crearon el primer robot industrial, llamado Unimate, que se utilizó en una línea de montaje de General Motors. Y posterior en el año 1961 el ingeniero Joseph F. Engelberger fundó la primera compañía de robótica, llamada "Unimation," lo que impulsó el desarrollo de la robótica industrial.

Ya en las últimas décadas la robótica se enfoca en el diseño, construcción, programación y operación de robots, siendo estos máquinas y dispositivos automáticos que pueden realizar tareas de manera autónoma y también controlada o asistida por humanos. Estas tareas pueden variar desde actividades simples, como ensamblar piezas en una línea de producción, hasta tareas más complejas, entrando al campo de la medicina, como la cirugía asistida por robots y también en el campo de la exploración de terrenos peligrosos en el espacio o en la Tierra, además con el constante desarrollo de la inteligencia artificial y la visión artificial, es posible la integración segura

entre humanos y robots, esto ha revolucionado la fabricación y otras industrias, además la visión artificial han mejorado en gran medida la capacidad de los robots para realizar tareas autónomas y adaptarse a entornos de cambios. La historia de la robótica plasma el avance y el ingenio humano en su búsqueda constante de crear máquinas que imiten y mejoren nuestras propias capacidades.

En resumen, la robótica se centra en la creación y el desarrollo de máquinas inteligentes capaces de realizar una amplia gama de tareas, lo que la convierte en un campo fundamental en la automatización y la mejora de la eficiencia en diversos sectores de la industria, la medicina y la sociedad.

CAPITULO 1: PROBLEMÁTICA Y OBJETIVOS

1 PROBLEMÁTICA Y OBJETIVOS

En este capítulo se describen las proyecciones y el alcance de las ideas propuestas para lograr el objetivo de la integración efectiva entre el entorno real y las acciones realizadas por el robot KUKA a través del procesamiento de imágenes en tiempo real, esto con la finalidad de poder establecer un vínculo y una interacción más inmediata entre personas y máquinas.

1.1 ANTECEDENTES GENERALES

En primera instancia se menciona la visión artificial, que representa una disciplina cuyo objetivo principal reside en el procesamiento y análisis de imágenes provenientes del entorno real, con el propósito de permitir comprender el entorno por a través de sistemas informáticos. La noción de visión artificial surgió en la década de los años sesenta del siglo pasado, y a lo largo del tiempo ha ido adquiriendo cada vez más relevancia y funciones, hasta convertirse en uno de los métodos más efectivos para la identificación de imágenes en la actualidad. Su principal aplicación se centra en el reconocimiento de patrones y se aplica en un amplio espectro, como la vigilancia y la seguridad, el reconocimiento facial y el funcionamiento de sistemas robóticos, para comprender mejor el comportamiento de la visión artificial es que se debe analizar su componente esencial y fundamental, el píxel, que representa a la mínima unidad que conforma una imagen digital. En resumen, la visión artificial implica una serie de operaciones matemáticas aplicadas a los píxeles de la imagen, las cuales varían según el objetivo deseado en un momento específico, estas operaciones se denominan filtros y se llevan a cabo mediante el uso de máscaras previamente configuradas para detectar similitudes, patrones y movimientos.

En segundo lugar, se hace presente la robótica que combina conceptos y técnicas de diversas disciplinas, como la mecánica, la electrónica, la informática y la ingeniería de control. Su objetivo principal es crear máquinas que puedan llevar a cabo tareas de manera eficiente, precisa y, en muchos casos, autónoma. Los robots pueden tener diversas formas y tamaños, desde brazos robóticos industriales hasta vehículos autónomos, drones y últimamente robots humanoides.

Con la intención de unificar ambos campos se considera la integración de sistemas, en este caso entrelazando la comunicación entre la visión artificial que corresponde a una mirada del entorno digitalizada y la ejecución de tareas complejas y laboriosas cómo es la robótica, llevando a ambas a un plano de intercomunicación constante en un mismo lenguaje que les permita interactuar sin la presencia de instrucciones intermedias por el usuario para la ejecución de cada movimiento, esta función se explorará utilizando el poderoso software de ingeniería MATLAB con sus propias herramientas de intercomunicación: IMAGE ACQUISITION TOOLBOX y comunicación TCP/IP, para ser receptada en el robot KUKA directamente por el controlador KR C4 buscando interpretar los datos al utilizar la librería “KUKA ETHERNET KRL” (Eki), otorgando así un rasgo de mayor autonomía en la ejecución de tareas avanzadas y en la integración del trabajo colaborativo entre humanos y máquinas.

1.2 UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA

La Universidad Técnica Federico Santa María (UTFSM) es una prestigiosa institución de educación superior en Chile, con una rica historia de desarrollo y destacada en la formación de ingenieros y profesionales en ciencias y tecnología. Fue fundada el 21 de febrero de 1926 por Don Federico Santa María Carrera, un destacado empresario chileno que tenía un profundo interés en la educación técnica y científica. Su visión era crear una institución que ofreciera una educación de alta calidad en ingeniería y tecnología para apoyar el desarrollo industrial y tecnológico de Chile, a lo largo de los años, la UTFSM ha experimentado un crecimiento significativo. Ha establecido campus en otras ciudades de Chile, como Viña del Mar, Santiago y Concepción, para atender a una población estudiantil diversa y en crecimiento.

Actualmente en la sede de Viña del Mar se han incorporado constantes actualizaciones a la infraestructura y al desarrollo e implementación de nuevas tecnologías, ampliando en gran medida la calidad de los laboratorios y de la educación en pequeña, mediana y gran escala, pero ello a su vez se presenta como un desafío para la docencia al incorporar estos avances y nuevos recursos para los estudiantes, es por ello que en este proyecto se aborda la inclusión de

importantes recursos obtenidos por la universidad que aún no han sido explotados cómo por ejemplo los brazos robóticos KUKA Agilus KR6 R900 que se encuentran en el laboratorio de Electrotecnia.



Fuente: Búsqueda por Google

Figura 1-1. Universidad Técnica Federico Santa María – Sede José Miguel Carrera

1.3 KUKA AGILUS KR6 R900

El KUKA Agilus KR6 R900 es un robot industrial compacto fabricado por KUKA Robotics. Este robot es parte de la serie Agilus, caracterizada por su agilidad y tamaño reducido, ideal para aplicaciones industriales que requieren alta precisión y velocidad. Con una carga útil de hasta 6 [kg] y un alcance de aproximadamente 901 [mm], el KR6 R900 es versátil y capaz de realizar tareas como ensamblaje, manipulación de piezas y paletización con rapidez y precisión. Los robots KUKA, incluyendo el Agilus KR6 R900, se programan básicamente utilizando su propio lenguaje “KUKA Robot Language (KRL)” con sus propias interfaces de programación. Además, se les dota de sistemas de seguridad para garantizar un entorno de trabajo seguro. Este robot se utiliza en diversas industrias, desde la automotriz y electrónica hasta la alimentaria, donde su capacidad para realizar tareas precisas y rápidas es esencial.

En resumen, el KUKA Agilus KR6 R900 es un robot industrial compacto y ágil diseñado para una variedad de aplicaciones en la industria. Su capacidad de carga útil, velocidad y precisión lo hacen valioso en entornos de producción que requieren automatización precisa y eficiente.



Fuente: www.kuka.com

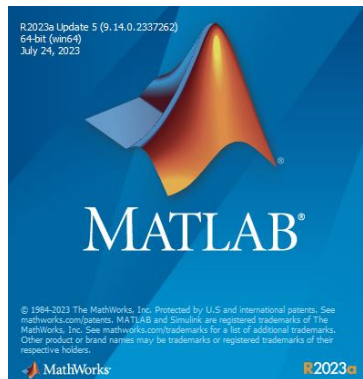
Figura 1-2. Brazo robótico KUKA Agilus KR6 R900

1.4 MATLAB

MATLAB, "MATrix LABoratory," es un poderoso software de programación y cálculo numérico ampliamente utilizado en ingeniería, matemáticas, ciencias y otros campos técnicos y científicos. MATLAB proporciona un entorno de desarrollo interactivo que permite a los usuarios realizar cálculos, programación y simulaciones de manera eficiente. Incluye un editor de scripts, un espacio de trabajo para gestionar variables y una interfaz gráfica de usuario (GUI) para crear aplicaciones personalizadas, además utiliza un lenguaje de programación propio (M) que es

especialmente adecuado para el procesamiento de matrices y datos numéricos. Es conocido por su simplicidad y versatilidad en la manipulación de datos, también ofrece una amplia biblioteca de funciones y herramientas para realizar una variedad de tareas, desde análisis matemáticos y estadísticos hasta visualización de datos y desarrollo de algoritmos.

MATLAB es conocido por su capacidad para generar gráficos y visualizaciones de datos de alta calidad. Los usuarios pueden crear gráficos 2D y 3D, diagramas, imágenes y animaciones para comunicar sus resultados de manera efectiva, se utiliza ampliamente en aplicaciones de procesamiento de señales y control, como diseño de filtros, simulación de sistemas dinámicos y control de sistemas automáticos, por último, se puede integrar con otros lenguajes de programación, como C/C++, Python, Java entre otros.



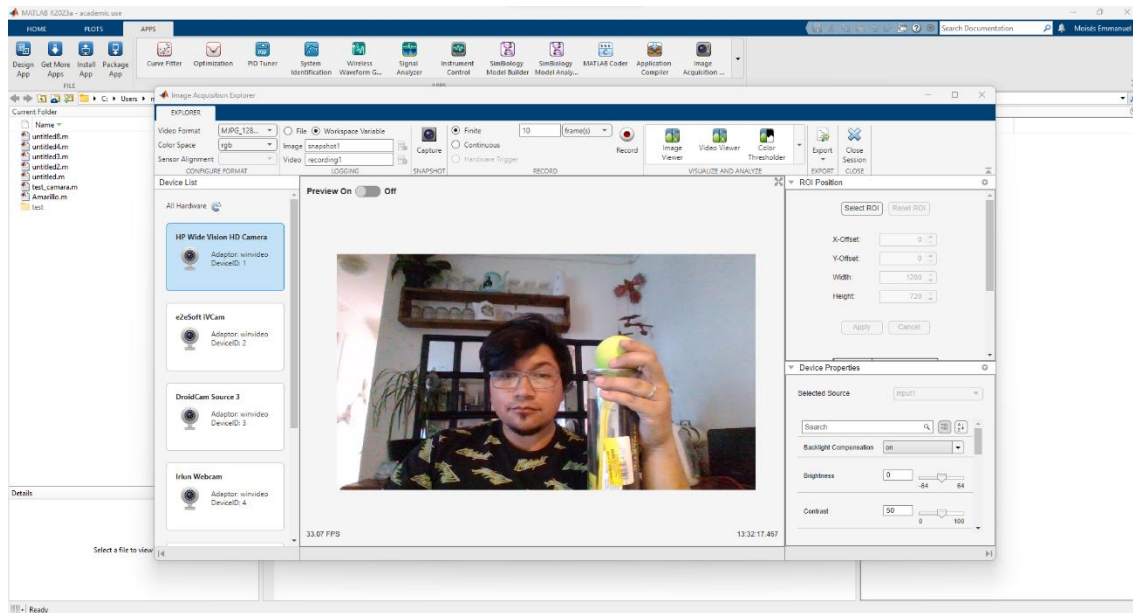
Fuente: recorte de pantalla

Figura 1-3. MATLAB

1.4.1 IMAGE ACQUISITION toolbox

La librería “IMAGE ACQUISITION toolbox” es un set de aplicaciones adicionales de MATLAB, este se utiliza para la obtención de imágenes de un dispositivo conectado vía USB a la computadora o vía Wi-Fi por diversos softwares o controladores, en conjunto con el procesamiento por matrices de las imágenes obtenidas, de esta manera mapea los píxeles en orden de matrices de bits para poder darle utilidad a la interpretación de imágenes, creando máscaras de filtros de color y morfológicas, pudiendo utilizar las variaciones en la toma de las

imágenes y la decodificación de los valores de estas mismas para establecer comportamientos según las necesidades de la programación.



Fuente: captura de pantalla

Figura 1-4. IMAGE ACQUISITION toolbox

1.4.2 Comunicación TCP/IP

TCP/IP es un conjunto de protocolos de comunicación que permiten la conexión entre dispositivos de red en internet. La sigla TCP/IP significa Protocolo de Control de Transmisión/Protocolo de Internet.

La comunicación TCP/IP para la transmisión de datos se establecerá dentro de la misma rutina de trabajo de MATLAB, para ello se asignará cómo cliente a la computadora y cómo servidor al controlador KR C4, incorporando las características de la librería KUKA Ethernet KRL, se buscará la adquisición de datos a través de un archivo "XML" y de esta manera se podrían obtener las variables en el programa principal del Robot KUKA.

1.5 PROBLEMÁTICA

El siguiente proyecto se encarga de atenuar el problema existente en la ausencia de actividades a realizar con el brazo robótico KUKA, ya que es un recurso relativamente nuevo en la universidad y no se han realizado clases ni talleres para explorar sus usos con los alumnos, es por ello que a causa de las extensas jornadas que poseen los docentes en la universidad, se ha tomado a modo de desafío y con el propósito de poder generar instancias de acercamiento al robot KUKA entre el alumnado y las carreras afines, el generar actividades de integración de sistemas y operación avanzada con dicha máquina a través de visión artificial, tomando como núcleo el vasto procesamiento y poder de integración que posee el software MATLAB utilizado durante la carrera.

1.5.1 Descripción del problema

Ahondando en las particularidades del sistema, como antecedentes se pueden considerar que el brazo robótico KUKA KR6 R900 se encuentra armado sobre una base con fines educativos, la estructura es móvil, posee cuatro ruedas con freno las cuales realizan la labor de soporte de una estructura de material resistente (fierro), pero se encuentra recubierta de material liviano, como lo son placas de madera aglomerada, dentro de esta estructura se encuentra su controlador KR C4 incorporando un computador industrial con interfaz gráfica Windows que posee salida de video a monitor para la programación e interfaz de acceso al operador como mouse y teclado, además de interfaz de comunicación USB, Ethernet y EtherCAT.

El brazo robótico KUKA incorpora una interfaz de operación llamada "Kuka SmartPAD", siendo un dispositivo manual de entrada para manipular el robot localmente con una distancia prudente de seguridad gracias a su extensión de cable de comunicación de 3[m] aproximadamente, este dispositivo tiene entradas de interacción a través de pulsadores y triggers, cuenta con una pantalla de 8 pulgadas para visualizar las instrucciones manuales y automáticas dentro de su lenguaje de programación KRL, esta interfaz, refleja directamente la

programación dentro del controlador, ya que pasa a ser un dispositivo de entrada para entre el operador / programador y el controlador KR C4.

Este equipo se encuentra en el Laboratorio de Electricidad de la UTFSM sede Viña del Mar y cómo se menciona en el apartado anterior aún no ha sido utilizado por el personal estudiantil.

1.5.2 Importancia de resolverlo

La importancia de poder resolver esta problemática yace en dar utilidad a los recursos obtenidos por la universidad, además de lograr integrar satisfactoriamente los conocimientos obtenidos durante la carrera de Ingeniería de Ejecución en Control e Instrumentación Industrial, y dar cabida a la exploración de este recurso educativo de alta calidad proporcionado a los alumnos y al personal docente, de esta manera implementar actividades variadas que se puedan realizar con el robot KUKA y la integración de visión artificial, tal como apunta este proyecto en particular, dando cabida la capacidad de trabajo colaborativo entre humano-máquina que es y será más aún en el futuro la base de nuestro desarrollo hacia la modernización.

1.6 PROPUESTA

Se propone en este proyecto ahondar y explorar la integración del brazo robótico KUKA en conjunto con visión artificial, primeramente, programando un sistema de análisis de datos obtenidos a partir de cámaras, para después poder programar eficazmente y con un nivel de complejidad adecuado instrucciones de operación al Robot KUKA, por último, poder buscar soluciones de integración entre ambos sistemas y crear un trabajo colaborativo.

1.7 REQUERIMIENTOS

Para cualquiera de las actividades mencionadas con anterioridad se debe tener en cuenta los recursos necesarios para esta aplicación, considerando que las actividades son de carácter

didáctico-educativo, es que los recursos a utilizar no requieren cumplir con los mayores estándares industriales y se utilizarán recursos comunes de fácil acceso para los estudiantes.

Considerando que esto cumple cómo base para posteriores desafíos industriales donde los estándares de calidad, sofisticación y robustez de los componentes involucrados si deben ser adecuados para el proceso que se estime conveniente.

A continuación, se presenta en la tabla 1-1 los requerimientos para este proyecto.

Tabla 1-1. Requerimientos del proyecto

Equipos y elementos	Descripción	Criterio de selección
Robot KUKA Agilus	Modelo KR6 R900, corresponde al modelo de brazo robótico que posee la universidad, se caracteriza por poseer 6 ejes de rotación, ser rápido y versátil.	Es el componente base del proyecto y ya se encuentra en las instalaciones de la universidad, por ende, no se considera un costo asociado al proyecto.
Laptop u ordenador con MATLAB	Puede ser un Laptop personal del alumno o un computador de escritorio proporcionado por la universidad para poder realizar la actividad, debe poseer la capacidad de trabajar con MATLAB.	Se considera cómo necesidad básica para el proyecto debido a su potencial de integración entre diversos dispositivos, actualmente la universidad dispone de ordenadores y de licencia educativa de MATLAB por lo que si llegase a generar un costo asociado no sería elevado.

Cámara de visión artificial	Este apartado es especial, ya que se puede integrar desde una simple cámara web, hasta una cámara industrial especializada, aunque dentro de las mejores opciones se puede encontrar la de dispositivos móviles (smartphones), las cuales tienen características bastante adecuadas para el trabajo con procesamiento de imagen.	Actualmente casi todas las personas poseen un smartphone y se pueden aprovechar las capacidades de estos para obtener información debido a su alta cantidad de sensores y sus cámaras de alta calidad, de igual manera la universidad puede proveer cámaras industriales de ser posible para mejorar el rendimiento de las aplicaciones y el conocimiento de operarlas.
Red de comunicación	Puede ser una red física con acceso a internet o una red Wi-Fi, esto es para proporcionar acceso a paquetes de actualizaciones de toolbox de MATLAB y para la transmisión de imágenes inalámbricamente de ser necesario.	Este segmento es un de carácter complementario, pero de igual manera muy importante para poder dar a vasto con las actividades a realizar, actualmente la universidad posee red Wi-Fi.
Objetivo por detectar	Corresponde al uso de elementos cotidianos o específicos para ubicar frente a la cámara, puede ser una pelota, un globo, el alumno,	El elemento lo puede traer el mismo alumno o el docente siendo muy amplio el espectro a utilizar según la actividad a realizar, para este

	una pieza con forma específica, etc...	caso en particular se utilizará una pelota de tenis de color amarillo.
--	--	--

Fuente: Elaboración propia para desarrollo del proyecto

1.8 **ALTERNATIVAS DE SOLUCIÓN**

Para dar solución a la problemática, se proponen tres alternativas de solución, las cuales representan distintos escenarios de programación, a nivel básico y experto, considerando elementos de mayor sofisticación y por ende diferenciándose unas de las otras.

1.8.1 **Alternativa N°1 “Implementar secuencias a través de KUKA SmartPAD”**

Esta alternativa representa un trabajo inicial con el dispositivo, refiere a una introducción con el brazo robótico, pero a nivel de usuario, cumple con la expectativa de dar utilidad al equipamiento de laboratorio, pero carece en cierto grado de innovación y utilización de conocimientos para sacar el mayor provecho a la integración de sistemas y a la automatización.

Se caracteriza por designar rutas específicas que el robot puede cursar al generar puntos con el manipulador manual del SmartPAD, sin variabilidad creadas cómo un script en lenguaje KRL para que el robot cumpla con ciertas ordenes e instrucciones, que van desde dibujar en un plano 2D hasta realizar un movimiento tipo saludo, recoger y depositar objetos que se encuentren siempre en la misma posición, etc.

Esta alternativa no es capaz de proporcionar un movimiento a través de intercomunicación entre sistemas, ya que solo cumple la función de otorgar movimientos y puntos en base a su propio sistema de coordenadas, es limitada en comparación con la programación a nivel “experto” proporcionada vía computador con el software “WorkVisual 6.0”.

1.8.2 Alternativa N°2 “Búsqueda de integración del robot KUKA Agilus con visión artificial utilizando los recursos existentes”

La segunda alternativa corresponde a la búsqueda de la integración de visión artificial con el robot KUKA Agilus de una manera discreta utilizando los recursos actuales, para ello se utilizará el software MATLAB con la licencia estudiantil, en conjunto con IMAGE ACQUISITION TOOLBOX para obtener y procesar imágenes en tiempo real, este software se instalará en un laptop personal del alumno, se considera utilizar cámaras de celular o webcams conectadas al laptop, de esta manera se obtienen y procesan las imágenes para crear máscaras de filtros de color y seguimiento de objetivo. Posteriormente una vez se interpreten las imágenes y se considere una respuesta adecuada, se procede a generar y probar la comunicación con KUKA Agilus a través del controlador KR C4, para ello es necesario exportar la coordenada desde MATLAB al controlador KR C4 por protocolo TCP/IP dando uso a la librería propia de KUKA “KUKA Ethernet KRL” que se encuentra ya instalada en el Robot, de esta manera se busca establecer una intercomunicación entre dos cámaras y un brazo robótico KUKA, considerando la programación en lenguaje propio de MATLAB y lenguaje KRL propio de los Robots KUKA.

1.8.3 Alternativa N°3 “Integración del robot KUKA Agilus con visión artificial utilizando los mejores recursos”

La tercera alternativa corresponde a la integración de visión artificial con el robot KUKA Agilus bajo estándares industriales de alta calidad utilizando nuevos recursos, para ello se considera el software MATLAB con una licencia profesional, en conjunto con IMAGE ACQUISITION TOOLBOX para obtener y procesar imágenes en tiempo real, el software se instalara en un ordenador de sobremesa independiente y especializado para entregar el mejor rendimiento para la tarea, para obtener las imágenes se utilizara una cámara industrial Datalogic P20C capaz de obtener una alta cantidad de imágenes por segundo, de esta manera poder crear un mapa de bits a alta velocidad para crear máscaras de filtros de color o filtros morfológicos para procesamiento de imágenes con una alta tasa de frames, una vez se interpreten las imágenes y se considere una

respuesta adecuada, se procede a la comunicación con KUKA Agilus a través de la conexión del controlador KR C4 ya antes mencionada, la comunicación se realiza físicamente a través de un controlador EtherCAT para potenciar la velocidad de transmisión de datos a alta velocidad utilizando la mayor cantidad de ancho de banda posible, de esta manera se establece una intercomunicación entre una cámara virtual y el brazo robótico KUKA de alta fidelidad y de alto rendimiento, considerando la transformación del lenguaje propio de MATLAB a un lenguaje KRL que es el que maneja el controlador KR C4, además evaluar la opción de trabajar con PLC Siemens S7 1500, en conjunto con la librería de paga “mxAutomation” que no se encuentra disponible en los recursos estudiantiles.

1.9 EVALUACIÓN DE LAS ALTERNATIVAS PROPUESTAS

Cómo alternativa escogida se utiliza la Alternativa N°2 “Integración del robot KUKA Agilus con visión artificial con los recursos actuales”, esto ya que es la alternativa que proporciona mayor innovación en sus procesos además de ser la más atractiva para implementar en el área de la instrumentación, a un bajo costo es capaz de proporcionar un sin número de desafíos y habilidades, además de ser la que aporta en mayor medida al desarrollo de competencias con respecto al enfrentarse al mundo de la robótica, sin la necesidad de utilizar altos recursos económicos para obtener un resultado y un propósito similar al de la opción N°3, que sin lugar a dudas entregaría la mejor calidad de operación y una tasa de respuesta muy eficaz, pero involucra un gasto que la alternativa N°2 no necesita.

Tabla 1-2. Tabla de puntuación

Tabla de Puntuación de Alternativas de Solución				
Muy deficiente	Deficiente	Aceptable	Bueno	Óptimo
1	2	3	4	5

Fuente: Elaboración propia para desarrollo de evaluación

Dada la escala de evaluación es que se procede a considerar los siguientes criterios de evaluación, asignando a cada uno de ellos una ponderación según su impacto global al momento

de ser implementadas, los criterios corresponden a, Innovación (25%), Recursos económicos (30%), Factibilidad (10%), Tiempo de implementación (20%), Desarrollo de competencias (25%).

Tabla 1-3. Tabla de evaluación de alternativas

Alternativa / Característica	Innovación (25%)	Recursos económicos (30%)	Factibilidad (10%)	Tiempo de implementación (10%)	Desarrollo de competencias (25%)	Nota Final
"Implementar secuencias a través de smartPAD"	1	5	5	5	1	3
"Integración del robot KUKA Agilus con visión artificial con los recursos actuales"	4	5	5	5	4	4.5
"Integración del robot KUKA Agilus con visión artificial utilizando los mejores recursos"	4	1	4	2	5	3.2

Fuente: Elaboración propia para desarrollo de evaluación

Cómo resultado de la evaluación, la Alternativa N°2 "Búsqueda de integración del robot KUKA Agilus con visión artificial utilizando los recursos actuales" cumple con los requisitos de evaluación para ser escogida cómo la solución apta a desarrollar.

1.10 OBJETIVOS

Este apartado pretende entregar los objetivos claves del proyecto a realizar, la finalidad que persigue y por qué se debe realizar, estipulando un marco a seguir para la ejecución del trabajo paso a paso.

1.10.1 Objetivo general

El objetivo general de este proyecto es “Lograr controlar los movimientos del Robot Kuka Agilus R6 900 interpretando coordenadas de un objeto en movimiento a través de visión artificial”

1.10.2 Objetivos específicos

Para cumplir con las metas indicadas en el proyecto se deben considerar los siguientes objetivos claves:

- Identificar y reconocer el funcionamiento y programación del Robot KUKA Agilus R6 900.
- Desarrollar una alternativa de integración con los elementos existentes.
- Crear un sistema de interpretación y exportación de datos basado en visión artificial.
- Integrar los datos obtenidos en la rutina de operación del Robot KUKA, y analizar sus límites operacionales.
- Evaluar alternativas de integración y costos asociados.

CAPITULO 2: PRUEBAS DE INTEGRACIÓN ENTRE SISTEMAS

2 PRUEBAS DE INTEGRACIÓN ENTRE SISTEMAS

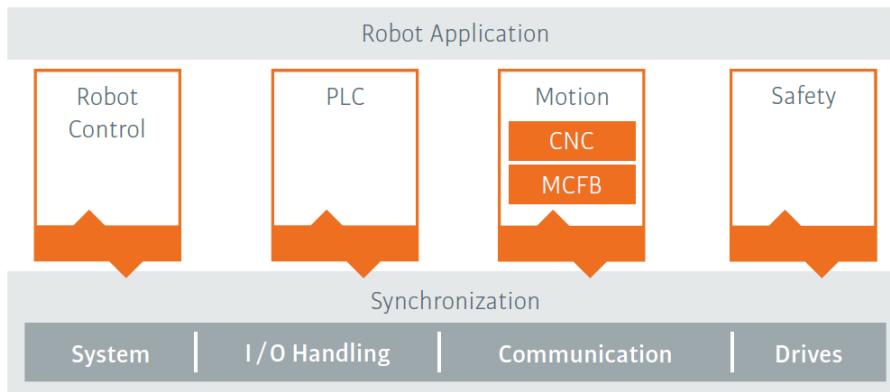
Este capítulo detalla la investigación, avances y programación requerida para buscar una alternativa efectiva en el desarrollo de una integración de sistemas entre visión artificial, utilizando la herramienta IMAGE ACQUISITION Toolbox de Matlab y las funciones de adquisición y proceso de instrucciones por parte de KUKA ETHERNET KRL (EKI), identificando primeramente cada uno de los elementos y componentes asociados al proyecto.

2.1 IDENTIFICACIÓN KUKA AGILUS KR6 R900

Para conocer y analizar el complejo actuador que se utilizará en este proyecto correspondiente al brazo robótico KUKA Agilus KR6 900, es necesario dividir su estructura en cuatro partes, iniciando por su controlador KR C4, el manipulador de 6 ejes de la línea Agilus, la interfaz de programación básica para el usuario correspondiente al KUKA SmartPad y finalmente su lenguaje propio de programación denominado KRL en conjunto con el software WorkVisual 6.0.

2.1.1 Controlador KR C4

El controlador KR C4 de KUKA (One For All) se caracteriza principal por ser sencillo y potente además de pionero en su estilo, en referencia a su nombre este controlador se compone de 4 interfaces que conforman una estructura de software robusta que es capaz de integrar las tecnologías Robot Control, PLC Control, Motion Control y Safety Control. Todos los controladores comparten una misma base de datos y una infraestructura, en consecuencia, se logra una programación más sencilla y potente, ya que permite una flexibilidad de compatibilidad al momento de realizar integraciones en diversos ambientes, tanto industriales cómo educativos.



Fuente: KUKA_PB_CONTROLLERS_EN es.pdf (Manual en biblioteca www.kuka.com)

Figura 2-1. Esquema controlador KR C4

En este proyecto se utilizará la versión KR C4 compact, ya que es la unidad que con la que actualmente se encuentra equipado el brazo robótico de la universidad, este controlador de 19" es ideal para ahorrar espacio en armarios y en pequeñas carcasas de protección, está diseñado para un manipulador de hasta seis ejes de baja carga útil, opcionalmente ampliable hasta dos ejes externos con una potencia de motor de hasta aproximadamente 2[kW].

Tabla 2-1. Descripción del controlador KR C4

Tipo de controlador	Global communicator
Compatibilidad global	KR C4 para compatibilidad en todo el mundo
Idiomas	25 idiomas, incluyendo los principales asiáticos
Cumplimiento de normas	Cumple con normas ISO y estándares / CSA relevantes en todo el mundo, incluyendo UL1740 / CSA434-3 / ANSI RIA 15.06 en EE.UU. y Canadá
Dimensiones (H x A x L)	271 x 483 x 460 [mm]
Procesador	Multi-core technology (Tecnología multinúcleo)

Disco duro	SSD
Interfaces	USB3.0, GbE, DVI-D, Display Port
Cantidad máxima de ejes	6 ejes + 2 anexando ejes externos
Frecuencias de suministro	50 / 60 [Hz] $\pm$ 1[Hz]
Voltaje de suministro	200 a 230 [Vac]
Grado de protección IP	IP20
Temperatura ambiente	+5 [°C] a +45 [°C]
Peso	33 [kg]

Fuente: KUKA_PB_CONTROLLERS_EN es.pdf (Manual en biblioteca www.kuka.com)



Fuente: KUKA_PB_CONTROLLERS_EN es.pdf (Manual en biblioteca www.kuka.com)

Figura 2-2. Controlador KR C4

El módulo de soporte y transporte del robot KUKA posee un compartimiento interior, en el cual se encuentra el controlador KR C4, este módulo se compone por una estructura interna rígida, una cubierta de material ligero y cuatro ruedas que soportan el peso completo del equipamiento.

2.1.2 Manipulador de 6 ejes

El manipulador es la cara visible del robot KUKA, es considerado muchas veces de manera errónea cómo el “robot” en sí mismo dejando de lado a su controlador, cabe destacar que se compone por 6 ejes de rotación: A1, A2, A3, A4, A5 y A6.

Tabla 2-2. Datos de los ejes del manipulador KR6 R900

A1	$\pm 170^\circ$
A2	$-190^\circ / 45^\circ$
A3	$-120^\circ / 156^\circ$
A4	$\pm 185^\circ$
A5	$\pm 120^\circ$
A6	$\pm 350^\circ$

Fuente: https://www.kuka.com/-/media/kuka-downloads/imported/8350ff3ca11642998dbdc81dcc2ed44c/0000205456_en.pdf

Sus ejes de rotación permiten una amplia gama de posiciones de trabajo y de desplazamiento, cuenta con una velocidad de operación máxima de 3[m/s] siendo uno de los más rápidos dentro de su gama de competencia, la línea KUKA Agilus es reconocida por el fabricante cómo la más rápida y precisa dentro de sus productos, considerada idónea para los trabajos industriales en procesos cómo paletizado de elementos a través de herramientas cómo grippers (pinzas) y ventosas para recoger objetos, además de herramientas más industriales tales cómo taladros y máquinas soldadoras en líneas de armado de vehículos, los robots KUKA son ampliamente conocidos por su estilo y ejemplos de trabajo automatizado a lo largo del mundo.



Fuente: Fotografía personal tomada en “Laboratorio Electrotecnia USM-JMC”

Figura 2-4. KUKA Agilus R6 900 Laboratorio

A continuación, se puede apreciar en la Tabla 2-3 las características del manipulador.

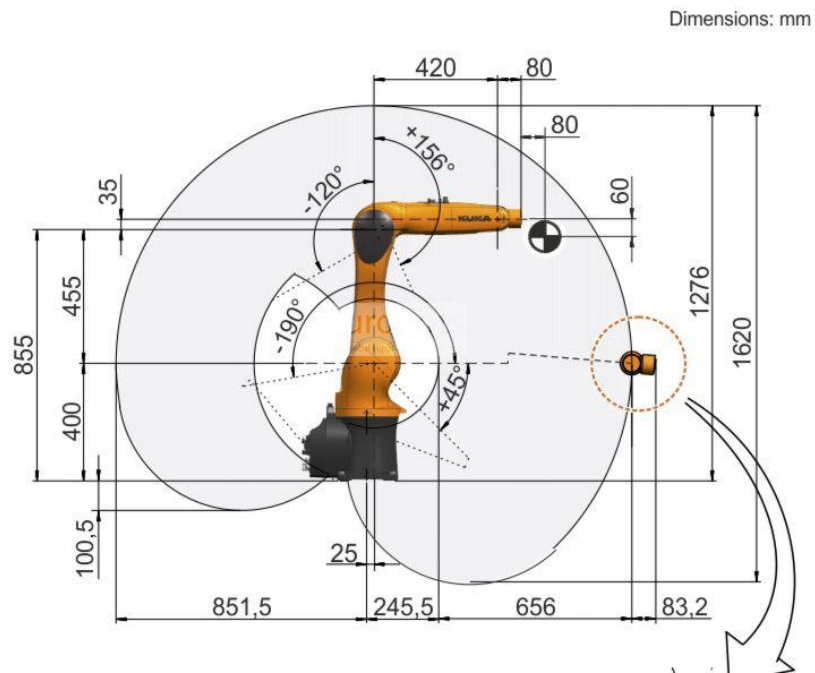
Tabla 2-3. Datos técnicos manipulador KUKA KR6 R900

Alcance máximo	901,5 [mm]
Carga máxima	6 [kg]
Repetibilidad de posición (ISO 9283)	$\pm 0,03$ [mm]
Número de ejes	6 ejes
Posición de montaje	Suelo; Techo; Pared en cualquier ángulo
Superficie de colocación	320 [mm] x 320 [mm]
Peso	Aprox. 52 [kg]
Condiciones de servicio	Temperatura ambiente entre 5 hasta 45 [°C]
Tipo de protección	IP general (IEC 60529) = IP54 // IP muñeca (IEC 60529) = IP54
Unidades de control compatibles	KR C5 micro; KR C4 smallsize-2; KR C4s micro

Tiempo de ciclo	150 Ciclos por minuto (25 [mm] / 305 [mm] / 25 [mm], 1 [kg]).
-----------------	---

Fuente: https://www.kuka.com/-/media/kuka-downloads/imported/8350ff3ca11642998dbdc81dcc2ed44c/0000205456_en.pdf

Los ángulos y distancias de operación se pueden apreciar de mejor manera en la siguiente figura 2-3, detallando el desplazamiento y el ángulo de inclinación.



Fuente: https://www.kuka.com/-/media/kuka-downloads/imported/8350ff3ca11642998dbdc81dcc2ed44c/0000205456_en.pdf

Figura 2-3. Ejes de operación KUKA Agilus R6 900

2.1.3 Kuka SmartPad

El KUKA SmartPad es una interfaz de control para robots industriales fabricada por KUKA, este dispositivo es esencialmente la interfaz de entrada predefinida para el usuario, corresponde a una tableta que se utiliza para programar, operar y controlar los robots industriales de la marca KUKA, su programación es intuitiva a través de una interfaz gráfica incorporada y también es un dispositivo seguro al poseer botoneras de seguridad y de interacción, estableciendo distintos niveles de operación, formatos de programación y uso.

Una de sus principales funciones es facilitar la operación del equipo y la enseñanza a nuevos operadores, ya que posee interacciones táctiles de movimiento manual y también a través de botoneras, posee además un control manual interactivo físico muy parecido a un “Trackpoint” que permite de manera eficiente simular el movimiento del brazo robótico escogiendo previamente un punto de origen según corresponda su orden de coordenadas, las coordenadas de operación son las siguientes: “World”, “Tool” o “Base”.

El KUKA SmartPad se encuentra enfocado para la operación inicial, tal como la de trabajos pequeños, con instrucciones básicas, que no requieran de un análisis de variables mayores, ideal para programadores que comienzan a trabajar con el equipo, a través de la tableta se pueden escoger las opciones de control y las velocidades de operación, también se pueden crear rutinas de operaciones simples para que se ejecuten de manera automática, para ello se debe crear un programa con sus respectivas rutinas y subrutinas que se ejecutan de manera muy similar al lenguaje C++, esto se ejecuta con el lenguaje de programación KRL propio de KUKA, otra de las funciones del KUKA SmartPad es atribuir el poder o la jerarquía a otros dispositivos de control o de programación cómo lo son el software de programación WorkVisual o ceder el control del controlador KR C4 cómo esclavo de un controlador superior cómo puede ser un PLC.

En la siguiente figura se puede apreciar el KUKA SmartPad y la descripción de su interfaz al operador.



Fuente: Manual KUKA_smartPAD_2_EN (Biblioteca www.kuka.com)

Figura 2-5. KUKA SmartPad

Tabla 2-4. Característica de KUKA SmartPad

Pantalla	Pantalla táctil, capacitiva, resistente a arañazos y apta para uso industrial
Tamaño de la pantalla	8,4"
Dimensiones	292 x 247 x 63 [mm] (altura x anchura x profundidad)
Peso	1,1 [kg]

Fuente: Manual KUKA_smartPAD_2_EN (Biblioteca www.kuka.com)

Cabe destacar que la programación a nivel “experto” del Robot no requiere en su totalidad el uso de esta herramienta, pero si es muy útil para reconocer la posición actual de los ejes, la posición actual sobre el plano y la creación de bases con su sistema “Touch up” que permite generar un sistema de coordenadas basado en la toma de puntos (muestras), este sistema de coordenadas se llama \$Base y se utilizará para establecer un perímetro de operación segura en este proyecto.

2.1.4 Lenguaje de programación KRL

El lenguaje de programación KRL (KUKA Robot Language) corresponde al lenguaje específico para programar los robots industriales fabricados por KUKA, el lenguaje de programación está diseñado para ser trabajado con controladores propios de KUKA, la programación con este lenguaje se realiza sobre las plataformas KUKA SmartPad y WorkVisual , creando archivos “SRC” que poseen una vista simplificada de los comandos y un archivo “DAT” de interpretación para el controlador que posee la programación compilada para el funcionamiento correcto del actuador, ambas plataformas de programación poseen un asistente que proporciona las instrucciones más utilizadas en el lenguaje KRL, además de permitir configurar variables como velocidad, aceleración, coordenadas de operación y tipo de desplazamiento al inicio de la programación declaradas como variables, además permite al programador designar tareas de movimiento sencillas como dirigirse de un punto a otro según el

origen de la coordenada, ya sea “World”, “Tool” o “Base”, la programación de este lenguaje corresponde a seguir sentencias de manera vertical descendente al igual que el lenguaje C++, además permite la interacción con elementos externos como entradas y salidas digitales (IO) del controlador asignando marcas llamadas “flags”, por ejemplo sensores y límites de carrera, cuenta además con la particularidad de poder guardar variables a utilizar dentro de sus mismas rutinas del programación, gracias a eso se permite utilizar y almacenar datos en las variables antes mencionadas y realizar operaciones matemáticas y lógicas con dichos valores, por otro lado cuenta con controles de flujo, programación en tipo bucle y condiciones de operación como las condiciones “IF” entre otras.

```

DEF FR_VERS ( )

;----- Sección de declaraciones -----
EXT BAS (BAS_COMMAND :IN,REAL :IN )
DECL AXIS HOME      ;Variable HOME del tipo AXIS
DECL FRAME BASE1    ;Variable BASE1 del tipo FRAME

;----- Inicialización -----
BAS (#INITMOV,0 )    ;inicialización de las velocidades,
                    ;aceleraciones, $BASE, $TOOL, etc.
HOME={AXIS: A1 0,A2 -90,A3 90,A4 0,A5 0,A6 0}
BASE1={FRAME: X 300,Y -100,Z 0,A 90,B 0,C 0}

;----- Sección principal -----
PTP HOME ; desplazamiento COI
; Movimiento respecto al sistema de coordenadas $BASE
PTP {POS: X 540,Y 630,Z 1500,A 0,B 90,C 0,S 2,T 35}
; Movimiento respecto al $BASE-SC corrido en BASIS1
PTP BASE1:{POS: X 540,Y 630,Z 1500,A 0,B 90,C 0,S 2,T 35}
PTP HOME
END

```

Fuente: <https://www.infoplc.net/descargas/73-kuka/2263-programacion-experta-robot-kuka-kr-c>

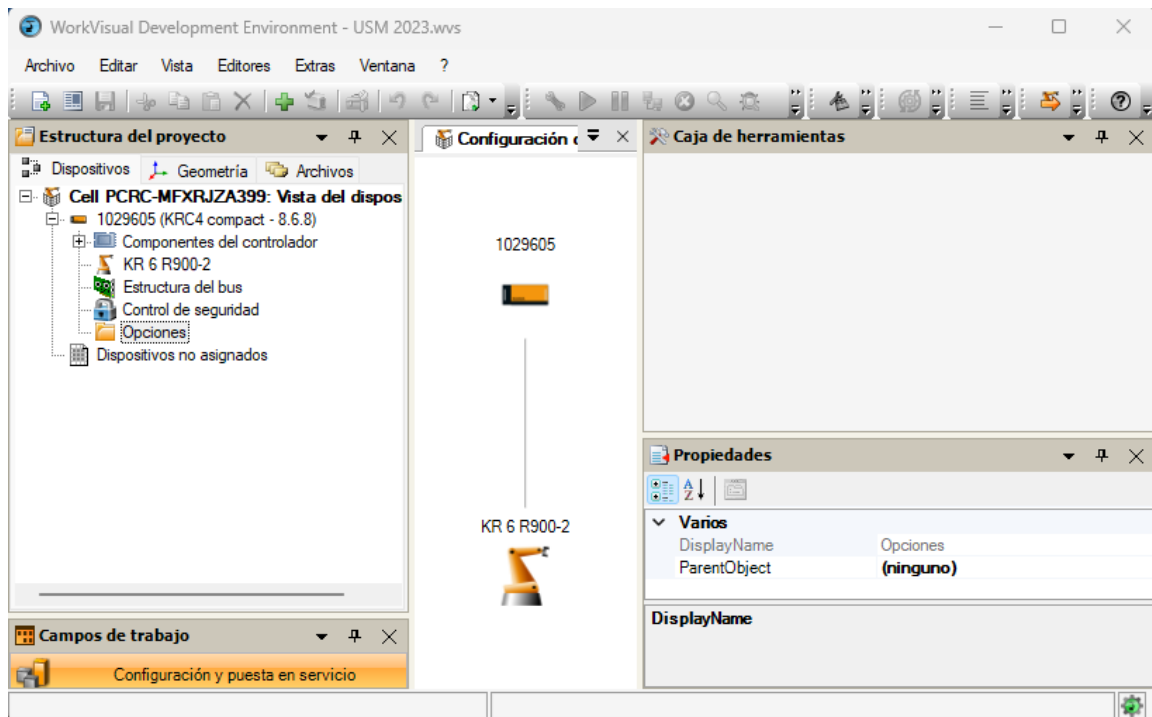
Figura 2-6. Instrucción de ejemplo KRL

2.1.5 Software de programación WorkVisual 6.0

El software de programación WorkVisual 6.0 de KUKA Robotics es una interfaz de programación avanzada para el programador “Experto” que se ejecuta en un ordenador con base Windows, este software es capaz de administrar múltiples configuraciones de programas,

controladores y manipuladores, además de poder cargar librerías de comunicación hacia el controlador KRC4 cómo es en este caso.

Se debe considerar que para cumplir con objetivo principal WorkVisual 6.0 es un software imprescindible, esto debido a que la programación dentro de este software puede ser offline y online lo que permite programar y revisar una configuración de trabajo con mayor flexibilidad, al crear una celda de proyecto se pueden incorporar diversos paquetes de librerías que aportan una gama más amplia de posibilidades de programación e integración con otros dispositivos, tales como comunicación PROFINET y TCP/IP a través de la librería KUKA ETHERNET KRL (EKI), acceder a la base de datos de los proyectos ya creados dentro del Robot KUKA y cargar nuevos proyectos.

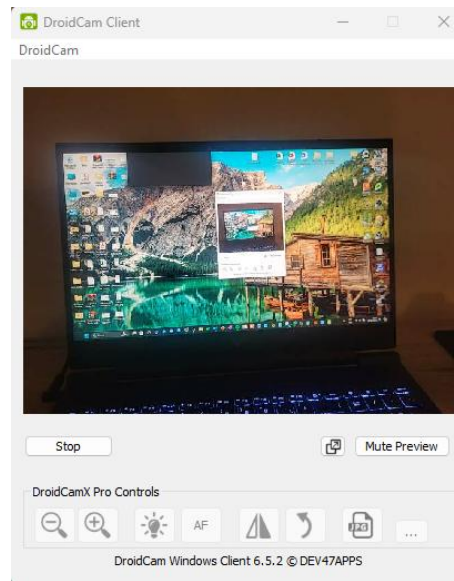


Fuente: Recorte de pantalla WorkVisual 6.0 ordenador propio

Figura 2-7. Entorno WorkVisual 6.0

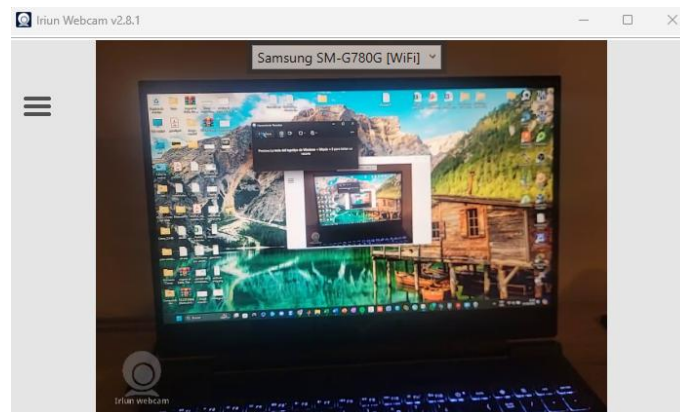
2.2 COMUNICACIÓN ENTRE CÁMARAS Y MATLAB

De acuerdo con los constantes avances tecnológicos de los últimos tiempos es posible hoy en día dentro del bolsillo de la mayoría de las personas encontrar un dispositivo multifuncional que posee una amplia gama de sensores y componentes, estos dispositivos son los smartphones, en esta sección del proyecto se utilizarán las capacidades de estos dispositivos para capturar imágenes digitales, además de tener una arquitectura de programación bastante avanzada que nos permite compartir nuestros datos con múltiples dispositivos y en múltiples plataformas, es por ello que acompañado de softwares como “DroidCam” e “Iriun Webcam” se compartirá el acceso de las cámaras de ambos dispositivos inteligentes al ordenador portátil (laptop) donde se trabajará con MATLAB, estos softwares se deben instalar en los smartphones que en este caso corresponden a un dispositivo de gama alta de hace 5 años, un Samsung S20 FE, este dispositivo trabaja con un sistema operativo Android, que es bastante compatible con funciones asociadas a Windows, estos softwares crean un canal de transmisión de datos a través de una red Wi-Fi o de conexión USB utilizando la depuración USB del dispositivo, en este caso se utilizará la comunicación vía Wi-Fi con una red estable para facilitar el posicionamiento de las cámaras en un espacio amplio sobre trípodes para facilitar su nivelación y distanciamiento entre ellas, en este punto es donde entra en juego el software MATLAB al recuperar la información enviada al ordenador a través de los softwares mencionados anteriormente utilizando las herramientas de la librería “IMAGE ACQUISITION toolbox”.



Fuente: Recorte de pantalla DroidCam Client ordenador propio

Figura 2-8. Enlace DroidCam Client



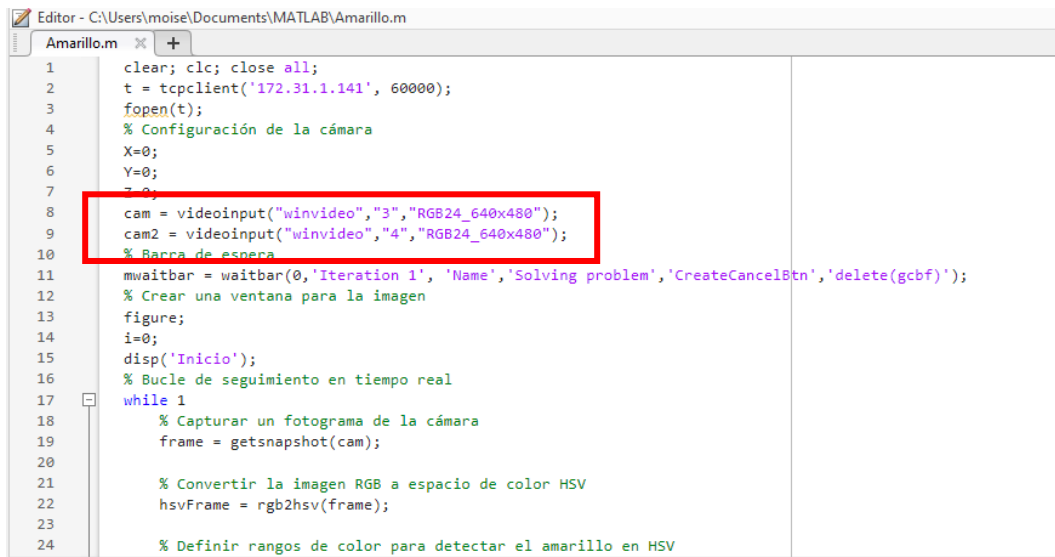
Fuente: Recorte de pantalla Iriun Webcam v2.8.1 ordenador propio

Figura 2-9. Enlace Iriun Webcam v2.8.1

2.2.1 IMAGE ACQUISITION toolbox

IMAGE ACQUISITION toolbox es una librería descargable de MATLAB utilizada para el reconocimiento de dispositivos que funcionan como entradas de imágenes y video, en este caso es capaz de reconocer los softwares “DroidCam” e “Iriun Webcam” como dispositivos de video

independientes y les asigna una ruta de comunicación dentro de MATLAB cómo se puede ver en las figuras siguientes:



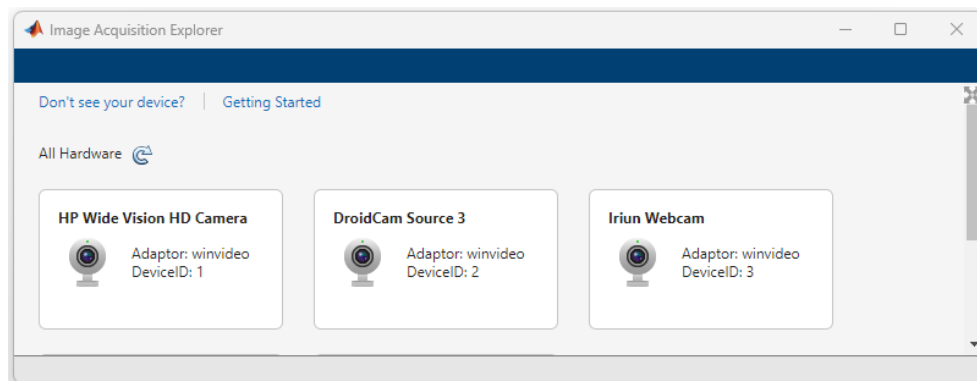
```

1 clear; clc; close all;
2 t = tcpclient('172.31.1.141', 60000);
3 fopen(t);
4 % Configuración de la cámara
5 X=0;
6 Y=0;
7 Z=0;
8 cam = videoinput("winvideo","3","RGB24_640x480");
9 cam2 = videoinput("winvideo","4","RGB24_640x480");
10 % Barra de espera
11 mwaitbar(0,'Iteration 1', 'Name','Solving problem','CreateCancelBtn','delete(gcf)');
12 % Crear una ventana para la imagen
13 figure;
14 i=0;
15 disp('Inicio');
16 % Bucle de seguimiento en tiempo real
17 while 1
18     % Capturar un fotograma de la cámara
19     frame = getsnapshot(cam);
20
21     % Convertir la imagen RGB a espacio de color HSV
22     hsvFrame = rgb2hsv(frame);
23
24     % Definir rangos de color para detectar el amarillo en HSV

```

Fuente: Recorte de pantalla Script propio MATLAB ordenador propio

Figura 2-10. Declaración de variables a dispositivos de video MATLAB



Fuente: Recorte de pantalla Script propio MATLAB ordenador propio

Figura 2-11. Selección de dispositivo IMAGE ACQUISITION toolbox

De acuerdo con la figura anterior se puede apreciar que la función “VideoInput” posee una estructura del tipo “(clase de dispositivo, canal asociado, formato de imagen)”, por lo que a cada cámara se le asigna una ruta diferente, respetando la igualdad en la resolución de cada dispositivo para poder analizar los datos de ambas cámaras en simetría, en primera instancia se

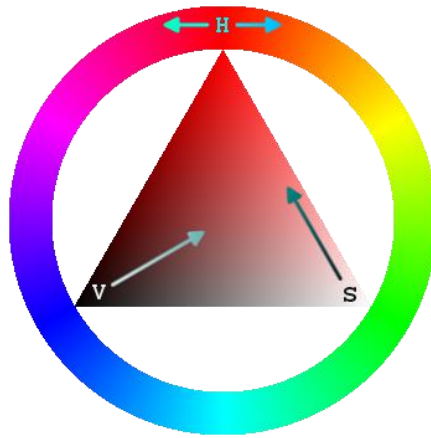
utilizará una resolución de 640x480 píxeles para cada muestra, de esta manera acelerar el procesamiento de datos, ya que no se requiere una mayor resolución para el seguimiento de objetos de mediano y gran tamaño.

2.3 SCRIPT DE PROCESADO DE IMÁGENES EN TIEMPO REAL

En este apartado se evaluará, analizará y ejecutará el desarrollo de un Script en MATLAB en conjunto con la librería antes mencionada IMAGE ACQUISITION toolbox la integración por medio de ambos softwares antes mencionados DroidCam e Irium Webcam previamente instalados en el ordenador y en los smartphones para el desarrollo de la integración del entorno real con el seguimiento del objetivo y la exportación actualizada de sus coordenadas en movimiento.

2.3.1 Captura y conversión

Una vez adquiridos los datos a través de la función “VideoInput” se crea un ciclo “while” en donde inicia por tomar los datos de la primera cámara con la función “getsnapshot”, al momento de adquirir la muestra procesa inmediatamente la imagen con un mapa de bits RGB, es por ello que descompone cada pixel en 3 matrices de saturación de color, ósea rojo, verde y azul, cada matriz de color posee en cada pixel ordenado un valor de 0 a 255 bits de saturación, para proporcionar el color adecuado en la superposición de las 3 matrices, al reconocer estos datos es posible convertir el arreglo matrices de RGB a HSV, correspondiente a sus siglas en inglés “Hue, Saturation, Value”, que corresponden a matiz, saturación y valor, esta conversión se realiza para identificar colores compuestos, en este caso el color a buscar es “Amarillo”, para contextualizar se escoge el color amarillo, ya que es un color que generalmente no se utiliza en las prendas de vestir cotidianas de las personas y además dentro del laboratorio existen muy pocos dispositivos que tengan este color, por ende se reduce la interferencia de otros objetos dentro la zona a analizar, de acuerdo a la siguiente figura se puede entender la función de colorimetría HSV.



Fuente: https://es.wikipedia.org/wiki/Modelo_de_color_HSV

Figura 2-12. Triángulo de conversión HSV

Para la conversión de colores se ajustan los valores HSV hasta encontrar el color amarillo.

Fuente: <https://www.peko-step.com/es/tool/hsvrgb.html>

Figura 2-13. Conversión RGB a HSV

Cabe destacar que por la presencia de la luz y del movimiento es que se considerarán un rango de valores HSV para poder captar diferentes tonos de color amarillo, cómo se puede apreciar en la siguiente tabla expresados en porcentaje del valor total.

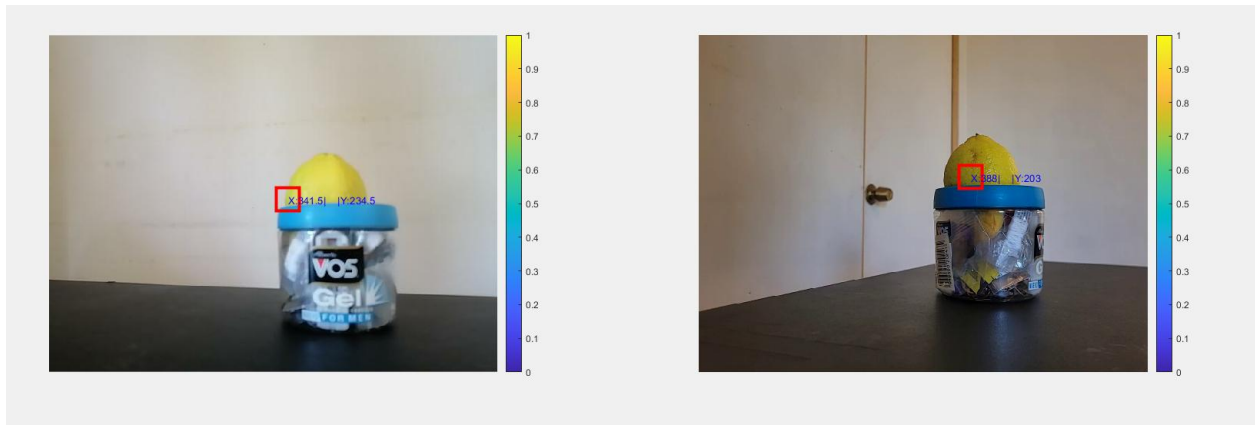
Tabla 2-5. Valores HSV a analizar en el Script

	HUE %	SATURATION %	VALUE %
Min	0.15	0.6	0.6
Máx	0.30	1	1

Fuente: Elaboración propia por rangos apropiados de color para el objetivo

2.3.2 Procesado de imágenes

El procesado de imágenes se realiza dentro del mismo ciclo “while” de la obtención y conversión de las capturas de imágenes, en este punto se utiliza la función “regionprops” junto a la asignación “centroide” para obtener el pixel central del objeto descubierto dentro de los tonos de color amarillo antes mencionados, es por ello que posterior a encontrar el centroide del objetivo y a modo de análisis se crea en MATLAB una interfaz gráfica de corroboración con la función “figure” para poder comprender el seguimiento del objeto, creando un marco cuadrado alrededor del centroide de color rojo, además de obtener las coordenadas en pixeles en un arreglo de matrices donde solo se presenta el eje X y el eje Y de la captura en 2 dimensiones frente a la cámara, estas también se ven reflejadas en la interfaz gráfica y se actualizan con cada muestreo de las cámaras, cabe destacar que el ciclo “while” toma las muestras de la cámara principal, las convierte, las procesa y entrega los datos de los ejes del centroide, para seguir inmediatamente con la segunda cámara, a pesar de que esto representa un retraso del orden de los [ms] es suficientemente rápido cómo para actualizar desde 2 a 5 fotogramas por segundo dependiendo de las condiciones de la red, de la iluminación y de la distancia con el objetivo, para comprobar esto se crea una barra de operación con conteo de iteraciones y con un botón de detención para dejar de procesar datos.



Fuente: Fotografía propia “ejemplo de funcionamiento Script”

Figura 2-14. Ejemplo de funcionamiento Script MATLAB

2.3.3 Exportar datos desde MATLAB

Para poder exportar los datos adecuadamente, se crea al inicio del script anterior una interfaz de comunicación TCP/IP con la función “tcpipclient”, se otorga la dirección de IP del previamente configurado y en la red de trabajo del sistema, se escoge el puerto de comunicación requerido, en este caso se utilizará el puerto “60000” y se aplica la configuración de comunicación por defecto para poder enviar información desde MATLAB, una vez establecida la comunicación en red con la función “fopen” se debe utilizar la función “fwrite” indicando la variable y además que tipo de variable es la que se enviará, para ello es necesario primeramente crear una variable del tipo “Struct” y asignar los valores dentro de esta con la función “sprintf”, de esta manera cómo se puede ver en el ejemplo se declara “Poskuka” cómo una variable del tipo “Struct” integrando las variables de coordenadas de la siguiente manera:

```
Poskuka = sprintf('{POS: < X > %g </X >, < Y > %g </Y >, < Z > %g </Z >}', X, Y, Z);
```

La utilización del formato “%g” es para asegurar el traspaso de un valor real con la cantidad de decimales que este posea sin agregar los ceros posteriores, ejemplo el valor 381.15 se enviará cómo tal, pero el valor 55.600 solo se enviará cómo 55.6, de esta manera es más efectiva la

Figura 2-15. Prueba de comunicación entre MATLAB y Hercules SETUP

En la anterior imagen de ejemplo se puede apreciar que al cambiar el valor de las variables de entrada también se ven afectadas las variables de salidas por el puerto de comunicación, dando a entender que éstas pueden variar en el tiempo cómo se puede ver en la siguiente figura de un ejemplo real.

2.4 CREACIÓN DE LA ZONA DE OPERACIÓN

Contemplar un espacio de trabajo definido es sumamente importante al momento de transferir los datos de manera regulada y ordenada, es por ello que esta parte del proyecto se centra en poder definir los parámetros de trabajo y crear un espacio simétrico de lectura y procesado de datos, para evitar errores y acumulación de datos irrelevantes, es por ello que se ajusta la zona de visión de las cámaras definiendo una distancia “L” y una altura “H” específica e igual para ambas cámaras, esto crea una especie de cuadrícula “invisible” pero definida dentro del campo de visión de las cámaras, considerando que ambas cámaras observarían la misma cantidad de pixeles por X [cm] de área de visión abarcado, también se considera un objetivo “central” para calibrar la posición de origen de las cámaras, esto evita que existan irregularidades a la hora de mapear los pixeles y transformarlos al área de trabajo del robot KUKA, no es relevante la distancia a utilizar para el centro del objeto, pero si es necesario que ambas cámaras estén a una distancia muy parecida del objeto, es decir que la distancia X[cm] desde el lente al objetivo sea cercana entre ambas cámaras, ya que no influye directamente la relación [cm] / pixel en el movimiento final del actuador, más bien solo se considera el valor de pixel del centroide del objeto.

2.4.1 Disposición física del área de trabajo

Para poder crear un área de trabajo se considerarán las siguientes variables:

D = distancia horizontal en X [cm] desde el lente de las cámaras hasta el centro del área de trabajo

T = altura del trípode hasta lente de la cámara

L = ancho del área de visión (Eje X)

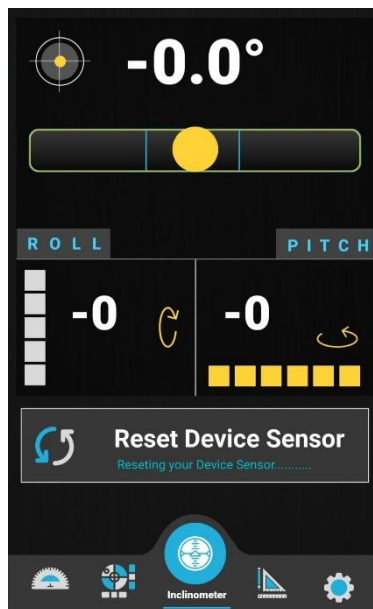
H = alto del área de visión (Eje Z)

X = corresponde al ancho del recuadro de imagen en [píxeles] de la cámara principal

Z = corresponde al alto del recuadro de imagen en [píxeles] de la cámara principal

Y = corresponde al ancho del recuadro de imagen en [píxeles] de la cámara secundaria

Para alinear la correcta inclinación en ambos dispositivos se utilizará una aplicación de inclinómetro, de esa manera los valores de Tilt, Pitch y Roll se encontrarán dentro del mismo rango, y eso reduce el error de la proyección visual de los píxeles.



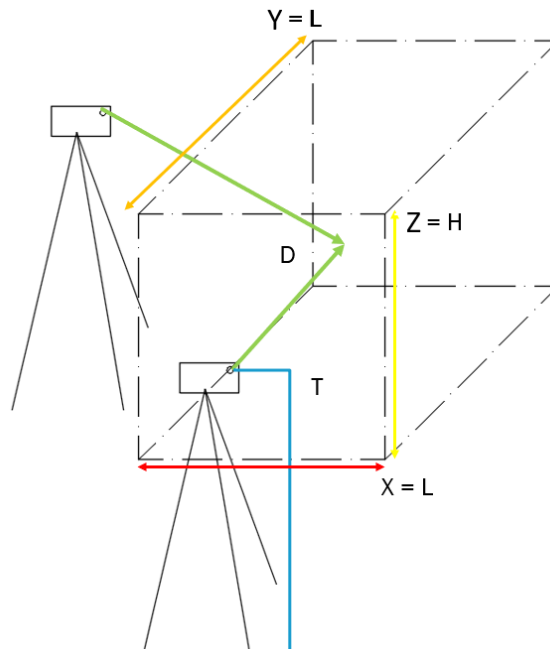
Fuente: Captura de pantalla Smartphone propio con "APP" Inclinometer

Figura 2-16. Calibración de orientación de Smartphone con APP Inclinometer

La lógica de virtualización consiste en alinear la misma cantidad de píxeles a un espacio definido en centímetros alrededor del objetivo central, para ello se configuran las distancias equivalentes entre ambas cámaras, así como su inclinación, de esta manera ambas cámaras abarcaran la misma cantidad de [cm] por [píxel] entre ellas, lo que proporciona una simetría al momento de crear el eje "Z" con la cámara secundaria, es por ello que se deben realizar ajustes

antes del inicio del Script, evaluando las distancias entre las esquinas de visualización de cada cámara, para poder crear un espacio 3D virtual según la cantidad de píxeles que abarca la imagen.

De esta manera, la resolución 640x480 [píxeles] abarcará una región de $X*Z$ [cm], al encontrarse simétrica la visual de ambas cámaras los 640 [píxeles] del eje X corresponderán al eje Y, es por ello que se conseguirá una región tridimensional de 640x480x640 [píxeles tridimensionales], lo que representa una región de trabajo de $X*Z*Y$ [cm³], esta proporción se utilizará para ajustar la zona de trabajo del manipulador creando una base de trabajo cómo se apreciará en el siguiente apartado, para visualizar de mejor manera se expresa la lógica de trabajo en el siguiente diagrama.



Fuente: CAD de Elaboración propia para exponer la zona de virtualización

Figura 2-17. Virtualización de la zona de trabajo

2.4.2 Creación del espacio 3D de trabajo en SmartPad

Cómo ya antes se ha mencionado, el brazo robótico KUKA Agilus KR6 R900 posee una interfaz de trabajo manual llamada SmartPad, gracias a este dispositivo es posible crear un

espacio de trabajo llamado \$BASE, para ello se debe ingresar al menú de opciones con la configuración manual y crear un área de trabajo tomando como referencia 4 puntos de base en el sistema de coordenadas \$WORLD, para ello se deben seguir los pasos del asistente de creaciones de bases, este permite a través del movimiento manual posicionarse en una coordenada específica del manipulador y con la herramienta “Touch up” se crea un punto que se asocia a uno de los vértices del área de trabajo, ya que no se requiere una base de trabajo inclinada se mantiene el eje Z a una altura prudente que no toque el mesón del equipo, y posteriormente solo se desplazan los ejes X e Y para generar los otros puntos de la base creando un área basal del tipo rectangular donde se podrá trabajar posteriormente. En las siguientes figuras se puede apreciar el desplazamiento del manipulador para la creación del área de trabajo.



Fuente: Fotografía propia de Robot KUKA en Laboratorio al momento de crear \$BASE de trabajo

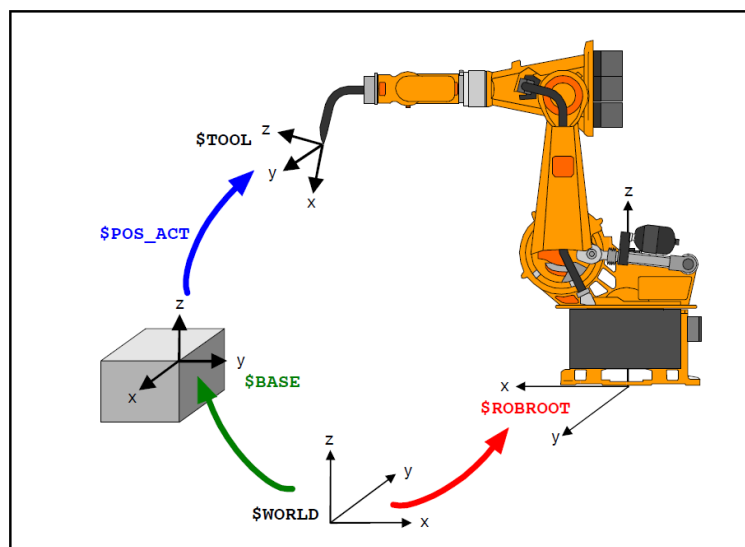
Figura 2-18. Creación de \$BASE de trabajo con asistente en SmartPad

Se considera el primer recuadro cómo OFFSET, siendo este el primer punto a tomar en cuenta de la creación de la base, este punto obtiene las coordenadas base de $\{X\ 0.0, Y\ 0.0, Z\ 0.0\}$, gracias a esto se toma como referencia para posteriormente trabajar con los movimientos dentro la base ya creada.

La base anteriormente creada se guarda en la memoria del robot dentro de los parámetros geométricos, enumerándola según el orden en el que estas sean creadas, actualmente el

dispositivo es capaz de almacenar más de 20 bases diferentes que se pueden utilizar para distintas subrutinas de trabajo, en este proyecto se utilizará la base N°2 llamada “ejemplo”, para seleccionarla se debe escribir #BASE 2.

Para las finalidades del proyecto no es necesario crear o utilizar una herramienta, por lo tanto, la opción \$TOOL no se utilizará, para ello es necesario seleccionar por software o a través del SmartPad la opción \$NULLFRAME para \$TOOL dejándola inhabilitada con las coordenadas en {0.0} para todos sus ejes de movimiento y espacio.



Fuente:

Figura 2-19. Sistemas de coordenadas Robot KUKA Agilus R6 900

2.5 PROGRAMACIÓN DE ROBOT KUKA EN WORKVISUAL 6.0

Programar un Robot Kuka puede ser una tarea compleja, y para ello se ha implementado un software de programación para programadores “expertos”, esto asegura un rendimiento mucho más eficiente y funciones mucho más completas para ejecutar a través del manipulador, la interfaz de este software permite crear un proyecto de trabajo desde cero, indicando el tipo de controlador, el tipo de manipulador y las librerías que se utilizaran para trabajar con estos, además de poder incorporar al proyecto más manipuladores en red y otros accesorios requeridos,

entregando una gama de posibilidades mucho mayores que al programar solo con el SmartPad.

Es por ello, que para este proyecto se utilizará la programación en lenguaje KRL dentro del software WorkVisual 6.0 que viene junto al Robot o descargarla gratuitamente desde la página oficial del fabricante en el apartado de Kuka Xpert para obtener su versión más actualizada.

2.5.1 Configuración del software y enlace con controlador KRC4

Lo primero es poder instalar el software WorkVisual 6.0 en nuestro ordenador o portátil (laptop, notebook), este software se puede conseguir entrando a la raíz de datos del controlador y descargándolo en una unidad de almacenamiento USB (Pendrive), los requisitos de operación para este software son mínimos y se pueden apreciar en la siguiente figura extraída de su página web.

Requisitos del sistema	Hardware <i>Robot controller:</i> • KR C4 or VKR C4 • Or KR C5 <i>Laptop/PC:</i> Minimum requirements: • PC with Pentium IV processor, min. 1500 MHz • 2 GB RAM • 200 MB free hard disk space • DirectX-compatible graphics card with a resolution of 1024x768 pixels Recommended specifications: • PC with Pentium IV processor or higher and 2500 MHz • 4 GB RAM • 1 GB free hard disk space • DirectX-compatible graphics card with a resolution of 1280x1024 pixels	Software <i>KR C4 robot controller:</i> • KUKA System Software 8.2, 8.3, 8.5 or 8.6 <i>VKR C4 robot controller:</i> • VW System Software 8.2, 8.3 or 8.6 <i>KR C5 robot controller:</i> • KUKA System Software 8.7 • Or VW System Software 8.7 <i>Laptop/PC:</i> • Windows 7 • Or Windows 10 If Multiprog is to be interfaced with WorkVisual: • KUKA.PLC Multiprog5-51 6.0 must be installed. • Multiprog must be licensed.
-------------------------------	---	---

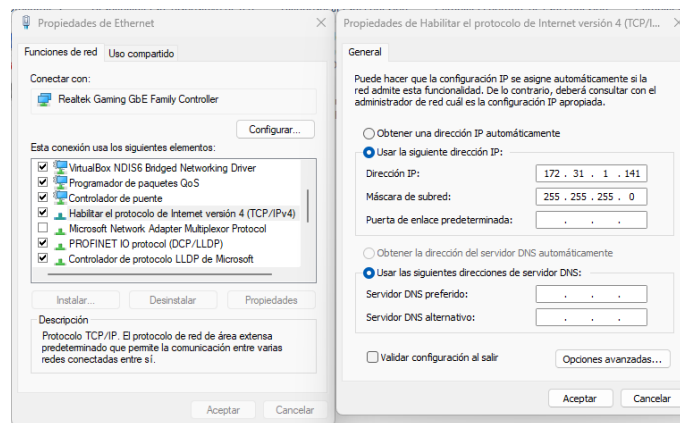
Fuente: https://www.kuka.com/es-es/productos-servicios/sistemas-de-robot/software/software-de-sistema/kuka_systemsoftware/kuka_work-visual

Figura 2-20. Requisitos del sistema de WorkVisual 6.0

La particularidad de este software es que permite trabajar de manera offline con un proyecto determinado o importar un proyecto existente conectándose al controlador vía ethernet, para ello solo se debe configurar previamente las características de la red y asignar una

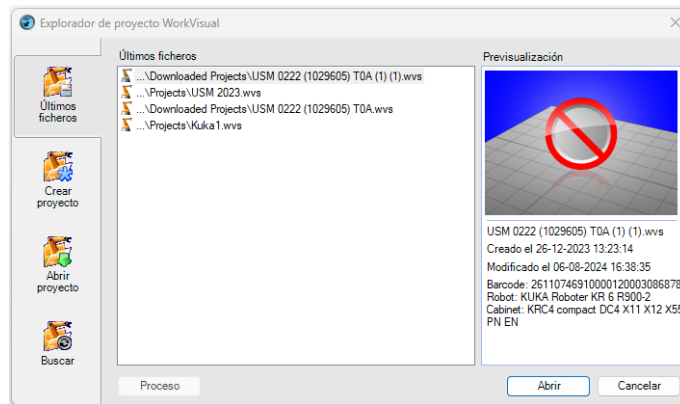
dirección IP dentro del segmento de red correspondiente para que ambos hosts puedan establecer una comunicación adecuada.

En este punto es imperativo configurar la KLI del equipo de control a través del SmartPad, ingresando a las configuraciones con los permisos de usuario “administrador” y posteriormente entrando al menú “Puesta en Marcha”, para configurar la KLI se ingresar al submenú “Configuraciones de Red” y se desplegará una ventana donde se puede ver la dirección IP del controlador y las direcciones IP internas de intercomunicación del sistema, solo se debe configurar la dirección externa del equipo, pero para los fines de este proyecto solo se reconocieron los datos de red del controlador y se configuró la red en función del controlador del Robot, cambiando las opciones del Red del ordenador, como se puede apreciar en la siguiente figura.



Fuente: Captura de pantalla de ordenador propio

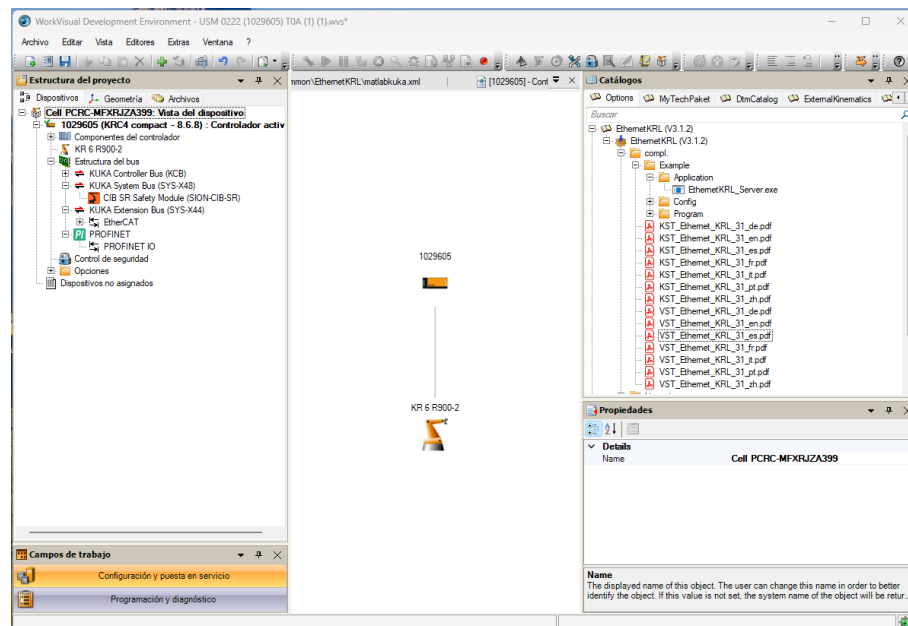
Figura 2-21. Configuración de red del ordenador



Fuente: Captura de pantalla de ordenador propio

Figura 2-22. Exploración de Proyectos WorkVisual 6.0

Una vez realizadas las configuraciones de red se puede buscar el proyecto en el ícono “buscar” cómo se puede apreciar en la figura anterior correspondiente al explorador de proyectos de WorkVisual, una vez abierto el proyecto se podrá ver una interfaz en el típico entorno de softwares de ingeniería cómo MATLAB o TIA PORTAL, con un menú de objetos al costado izquierdo un menú de trabajo al centro y un menú de catálogos y especificaciones al costado derecho.

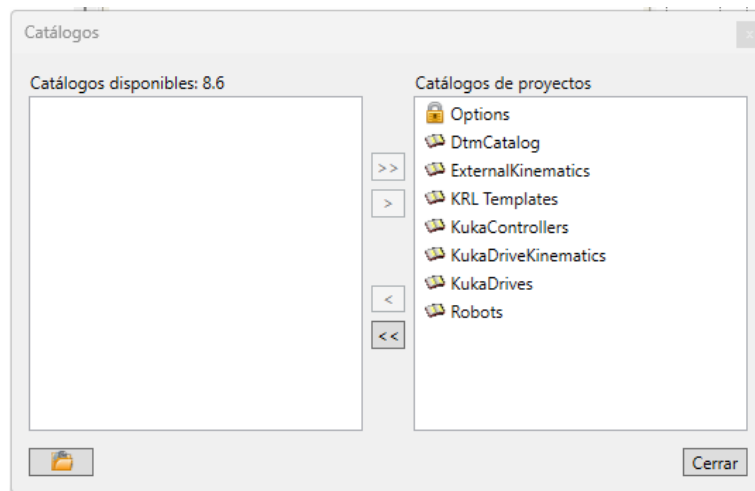


Fuente: Captura de pantalla de ordenador propio

Figura 2-23. Interfaz de trabajo WorkVisual 6.0

Para disponer de las funciones principales del software estas deben ser agregadas al catálogo del proyecto con las siguientes instrucciones:

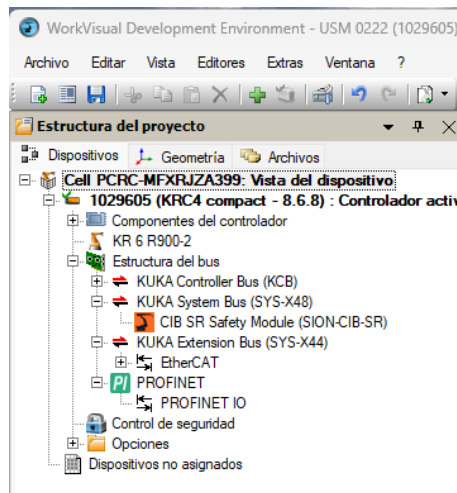
- Desplegar el menú superior dando clic en la pestaña “Archivo”
- Seleccionar la opción “Administración del catálogo”



Fuente: Captura de pantalla de ordenador propio

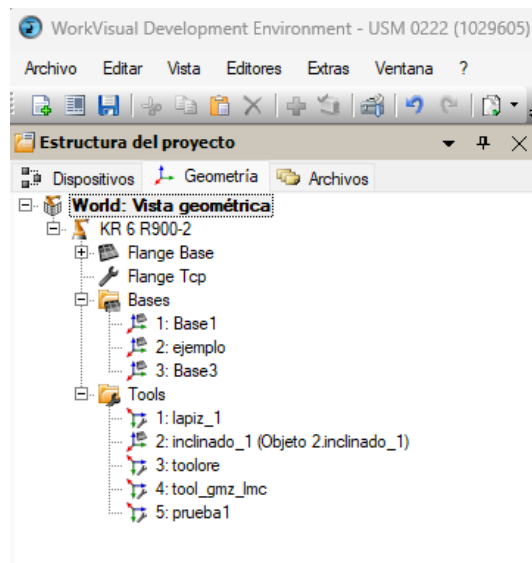
Figura 2-24. Administrador de catálogos de WorkVisual 6.0

En la figura anterior se pueden apreciar los catálogos incorporados al proyecto actual, además es importante destacar que las configuraciones del controlador y del actuador se verán reflejadas de manera inmediata cuando se importe el proyecto existente, no así cuando se cree uno desde cero, por ende, lo más recomendable es trabajar con el proyecto importado directamente desde la célula del robot para modificarlo y no incurrir en algún error por falta de elementos o una elección errónea de componentes y tecnologías de comunicación.



Fuente: Captura de pantalla de ordenador propio

Figura 2-25. Árbol de dispositivos del proyecto en WorkVisual 6.0



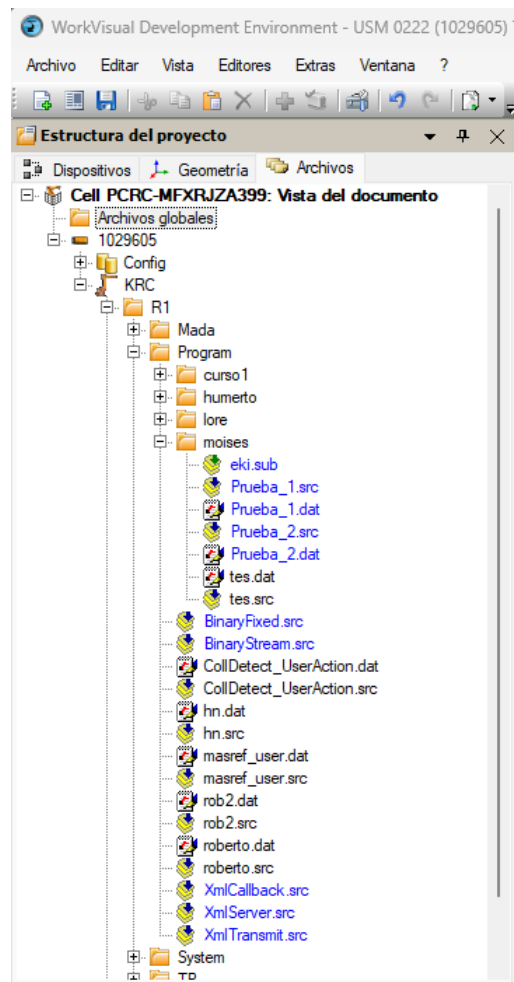
Fuente: Captura de pantalla de ordenador propio

Figura 2-26. Árbol de elementos geométricos en WorkVisual 6.0

En la figura anterior se puede apreciar el diagrama de dispositivos existente en el proyecto y en el menú desplegable se pueden realizar configuraciones de cada ítem desplegado.

En la siguiente figura se muestran los elementos geométricos del proyecto, tales como las herramientas (\$TOOLS) conectadas en el extremo del manipulador y las bases (\$BASE) correspondientes a los planos con coordenadas cartesianas de trabajo.

Por último, en el menú de archivos se puede encontrar todos los directorios, archivos y documentos que se encuentran dentro del controlador, para acceder a los programas se debe ingresar a la siguiente ruta: ***“Célula del robot/serie del controlador/KRC/R1/Program”*** cómo se puede apreciar en la siguiente figura.



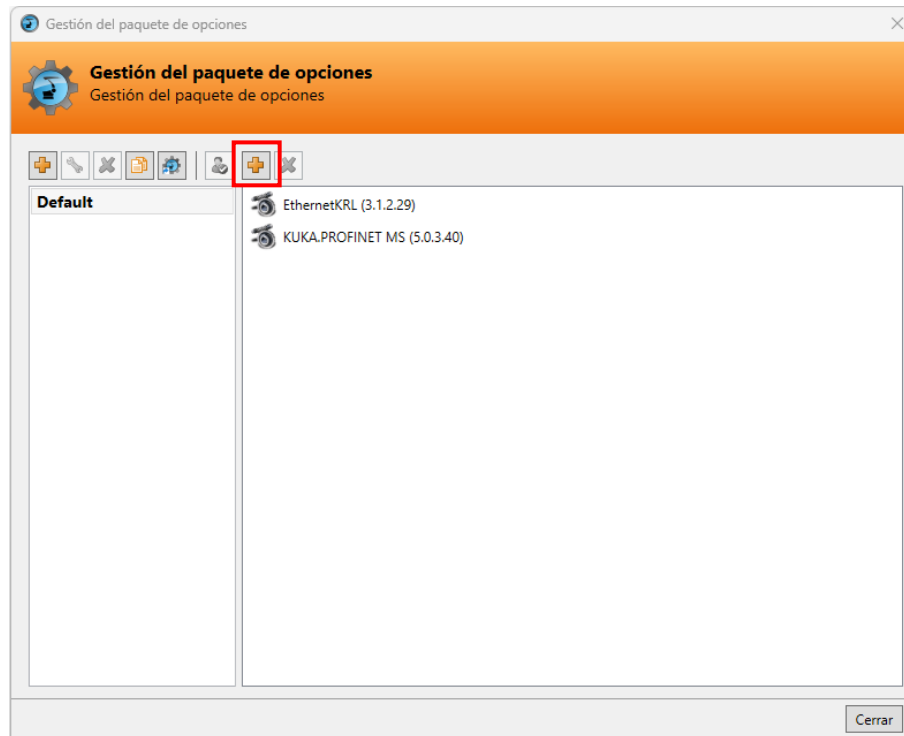
Fuente: Captura de pantalla de ordenador propio

Figura 2-27. Explorador de archivos del proyecto en WorkVisual 6.0

2.5.2 Incorporación de librerías

En este apartado se explicará la herramienta de catálogos y librerías que pueden añadirse a la base del software de programación WorkVisual dentro del mismo con las siguientes instrucciones:

- Para añadir las librerías se debe desplegar el menú de la pestaña superior “Extra”
- Posteriormente se debe dar clic en “Gestión del paquete de opciones”



Fuente: Captura de pantalla de ordenador propio

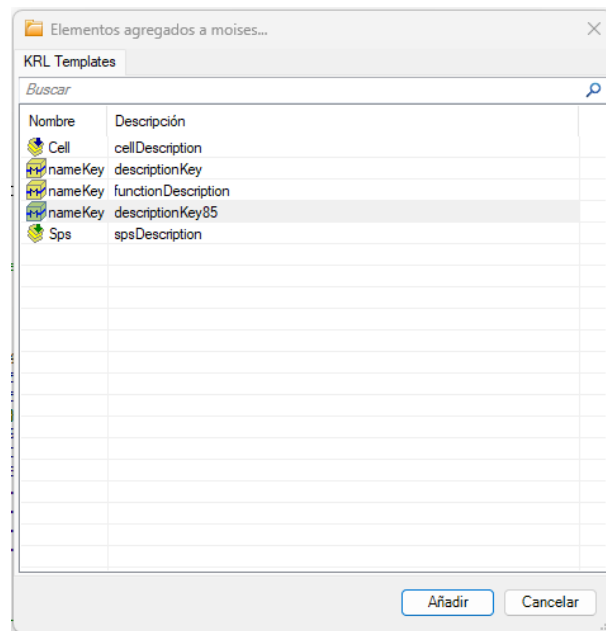
Figura 2-28. Gestor de paquetes en WorkVisual 6.0

En la figura anterior se puede ver resaltado dentro de un recuadro de color rojo el icono con el signo “+”, este icono despliega un menú de directorios para encontrar el archivo de las librerías con extensión “.kop” esta extensión solo sirve para ser incorporada a través de

WorkVisual al controlador KRC4 y KRC5, las librerías son archivos de pago a través de la web oficial de KUKA, pero en este caso se ha obtenido vía extracción de datos desde el mismo Robot, ya que venía como archivo dentro de este, de otra manera las funciones EthernetKRL y KUKA.PROFINET MS no hubieran estado disponibles, imposibilitando la opción de comunicar el controlador KRC4 con MATLAB en el ordenador.

2.5.3 Creación de ciclo de trabajo en coordenadas \$BASE

Una vez dentro de la pestaña archivos se da clic derecho en la carpeta “Programs” y se añade un directorio, posteriormente dando clic derecho al nuevo directorio y presionando “+ Añadir” se puede crear un archivo de programa cómo se puede ver en la siguiente figura.



Fuente: Captura de pantalla de ordenador propio

Figura 2-29. Creación del archivo de programación en WorkVisual 6.0

Inicialmente se creará un archivo del tipo “.src” al dar clic en añadir sobre la composición “nameKey / descriptionKey85”, al añadir este nuevo archivo de programación se creará en

conjunto su respectivo archivo “.dat” que corresponde a la programación interna del controlador, esta reescribe los datos cuando se utilizan comandos asistidos por el software para asegurar un correcto compilado de los procesos, pero cuando se escriben instrucciones directas no se genera una interpretación de datos, es decir que si utilizamos la función “PTP” asistida y aplicamos la herramienta “Touch up” se generará una instrucción sencilla, pero en el archivo “.dat” se reescribirá para poder ser interpretada correctamente por el controlador, en cambio cuando se declaran las variables sobre una base o se crea una base nueva utilizando las instrucciones escritas, no se requerirá una interpretación en el archivo “.dat” cómo se puede ver en la siguiente imagen.

```

39 POSITION10={POS: X 78.8, Y 112.2, Z 35.9}
40
41 --ENDFOLD
42 ;FOLD MOVIMIENTO
43 ;----- Sección principal -----
44 LOOP
45
46 ;PTP P02 CONT VEL=50 & PDAT02 TOOL[0] BASE[2];EXTNPL0 COLDETECT[1];
47
48 PTP HOME ;desplazamiento COI
49
50 LIN POSITION1
51 LIN POSITION2
52 LIN POSITION3
53 LIN POSITION4
54 LIN POSITION5
55 LIN POSITION6
56 LIN POSITION7
57 LIN POSITION8
58 LIN POSITION9
59 LIN POSITION10
60
61 PTP HOME
62 --ENDFOLD
63 --ENDLOOP
64 --END

```

```

1 DEFDAT Prueba_2
2 ;FOLD EXTERNAL DECLARATIONS;%(PE)%MKUKATPBASIS,%CEXT,%VCOMMON,%P
3 ;FOLD BASISTECH EXT;%(PE)%MKUKATPBASIS,%CEXT,%VEXT,%P
4 EXT BAS (BAS_COMMAND :IN,REAL :IN )
5
6 --ENDFOLD (BASISTECH EXT)
7 ;FOLD USER EXT;%(E)%MKUKATPUSER,%CEXT,%VEXT,%P
8 ;Make your modifications here
9
10 --ENDFOLD (USER EXT)
11
12 --ENDFOLD (EXTERNAL DECLARATIONS)
13 DECL ESPOS XP02 = ( X 666666.666, Y 66.6, Z 666.66, A 6.6, B 66.66, C 666.6, S 0, T 0, E1 0, E2 0, E3 0, E4 0, E5 0, E6 0, E7 0, E8 0, E9 0, E10 0 )
14 DECL ESPOS XP02 = ( X 666666.666, Y 66.6, Z 666.66, A 6.6, B 66.66, C 666.6, S 0, T 0, E1 0, E2 0, E3 0, E4 0, E5 0, E6 0, E7 0, E8 0, E9 0, E10 0 )
15 DECL ESPOS XP02 = ( X 666666.666, Y 66.6, Z 666.66, A 6.6, B 66.66, C 666.6, S 0, T 0, E1 0, E2 0, E3 0, E4 0, E5 0, E6 0, E7 0, E8 0, E9 0, E10 0 )
16 DECL PDAT PPDAT02 = (APO_MODE #CDIS,APO_DIST 500,VEL 50,ACC 100,GEAR_JERK 100.0,EXAX_IGN 0)
17 DECL PDAT PPDAT02 = (BASE_NO 2,TOOL_NO 0,IFO_FRAME #BASE,POINT2[ ] " " )
18 ENDDAT
19

```

Fuente: Captura de pantalla de ordenador propio

Figura 2-30. Demostración del interpretador interno del Robot

Para este proyecto se utilizarán las coordenadas previamente creadas en la base, por lo que es indispensable asignar la base adecuada al momento de programar los movimientos de desplazamiento en los distintos ejes, para ello se debe anotar en la programación posterior a la zona de declaración de variables la asignación de velocidades, la asignación de base y de la herramienta a trabajar con los siguientes enunciados:

BAS (#INITMOVE, 0)

BAS (#BASE, 2)

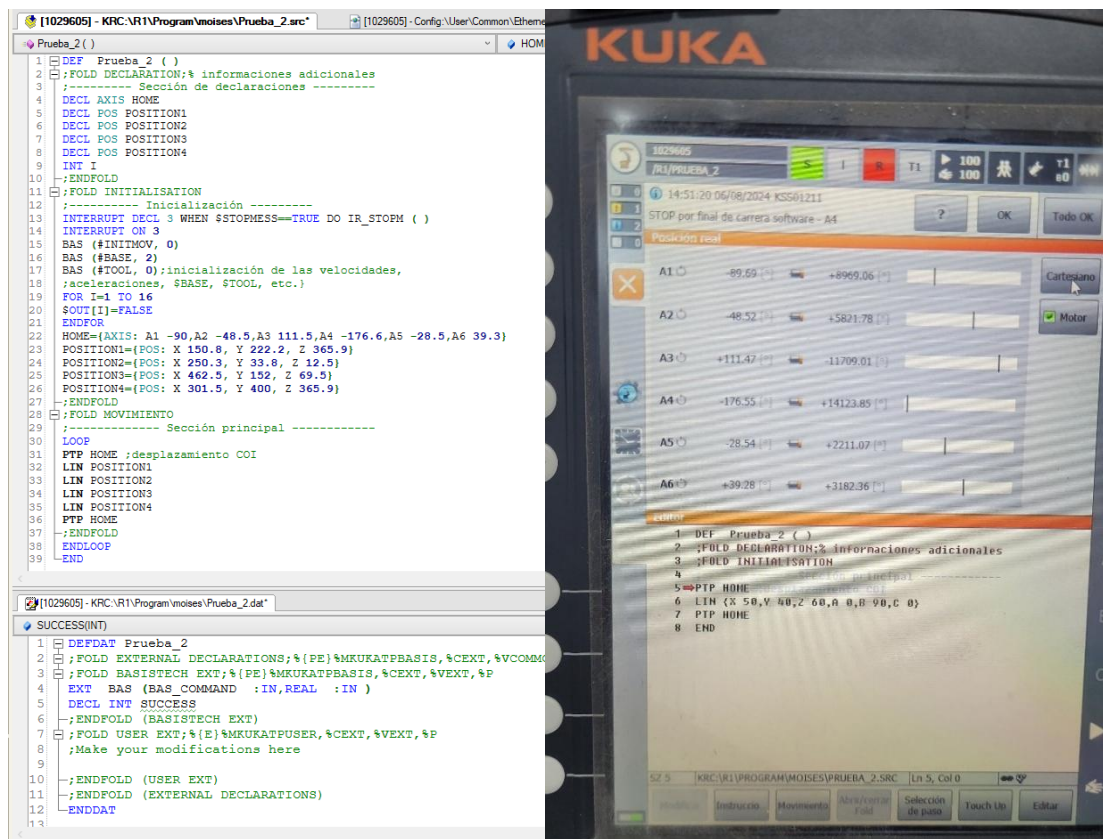
BAS (#TOOL, 0) o \$TOOL = \$NULLFRAME

BAS (\$VEL, 30) = velocidad constante a 30%

BAS (\$ACC, 50) = aceleración constante al 50%

Por lo tanto, el archivo “.dat” se encontrará casi vacío de instrucciones interpretadas, ya que las instrucciones serán directas en el archivo “.src”, y esto corresponde a una peculiaridad de la programación en modo “experto” ya que representa el libre uso de variables para generar el movimiento del manipulador, ya sea en ejes cómo en direcciones.

En la siguiente figura se puede apreciar lo comentado en el párrafo anterior además de poder analizar la declaración del punto inicial “HOME”, esta estructura de coordenadas se establece bajo la disposición del movimiento de los ejes con la función “AXIS”, para ello se evalúa a través del SmartPad la posición actual de los ejes del manipulador, posteriormente se realiza manualmente un movimiento para reconocer el ángulo de cada uno de los ejes de movimiento, una vez capturados o anotados se pueden establecer posiciones del manipulador en base al recorrido angular de los ejes.



Fuente: Captura de pantalla de ordenador propio y fotografía de KUKA SmartPad

Figura 2-31. Demostración del programa KRL y posición de ejes para punto HOME en SmartPad

Con los datos anteriores y utilizando la guía del programador experto de KUKA se logra programar una secuencia de movimientos con declaraciones de variables modificables en lenguaje KRL a nivel “experto” sin la necesidad de utilizar el SmartPad para la implementación del movimiento. Cabe destacar que el uso del SmartPad en esta prueba corresponde solo a dar partida al programa y revisar los alcances del movimiento avanzando la secuencia manualmente para evitar colisiones, ya que al programar en modo experto las colisiones se deben declarar de manera aparte cómo si fueran una restricción de movimiento en el plano.

2.5.4 Relación de transformación Pixel a [mm] de movimiento

En el siguiente apartado se explica la relación entre los datos obtenidos a través de las cámaras al movimiento ejecutado por el manipulador, para ello se debe considerar lo siguiente:

La relación de aspecto de la toma de muestras digitales corresponde a 4:3 con una resolución de 640x480 pixeles por cámara, considerando que la cámara principal posee los ejes X-Z y la segunda cámara con una resolución de 640x480 pixeles toma el valor del eje Y asignado a la distribución horizontal de pixeles da una resolución tridimensional de 640x480x640 pixeles para X, Z e Y. Por lo que en primera instancia se recopilan datos manualmente con el SmartPAD operando el manipulador para conocer las distancias de trabajo sobre el mesón, siendo estos los resultados bajo las coordenadas \$WORLD.

Tabla 2-6. Muestreo de coordenadas con SmarPad

<i>Muestra inicial \$WORLD</i>						
				X [mm]	Y [mm]	Z [mm]
Punto 1	Inferior	Izquierda	Frontal	-48,23	40,72	0,08
Punto 2	Inferior	Derecha	Frontal	-466,59	-592,99	0,05
Punto 3	Superior	Izquierda	Frontal	-51,93	27,16	500,02
Punto 4	Superior	Derecha	Frontal	-486,52	-521,60	484,77
Punto 5	Inferior	Izquierda	Trasero	-336,00	328,93	-0,48
Punto 6	Inferior	Derecha	Trasero	-853,23	-243,28	0,03
Punto 7	Superior	Izquierda	Trasero	-385,42	328,67	499,76
Punto 8	Superior	Derecha	Trasero	-853,67	-189,40	482,35

Fuente: Elaboración propia para el desarrollo y comprensión del movimiento del Robot

Posteriormente se analiza el comportamiento utilizando las coordenadas \$BASE 2 (ejemplo), base de trabajo creada con anterioridad en el proyecto y se logra encontrar un punto OFFSET adecuado con las siguientes características {X 80.0, Y 0.0, Z 0.0} valores numéricos en [mm] con respecto a la distancia de la base creada, siendo este punto nuestro OFFSET para establecer una referencia de movimiento. Además, se establece una relación de movimiento a partir de la coordenada en píxel obtenida para promover un movimiento homogéneo dentro del

espacio tridimensional, es por ello que los valores en coordenadas de pixel se escalan según la siguiente tabla.

Tabla 2-7. Factor de escalado para desplazamiento en \$BASE

Ejes	X	Z	Y
Píxeles cámaras	640	480	640
Factor distancia [mm] KUKA	0,9375	1	0,75

Fuente: Elaboración propia para el desarrollo y comprensión del movimiento del Robot

Esto genera una correlación de movimiento en milímetros del manipulador que se puede apreciar en la siguiente tabla cómo movimientos máximos en cada eje, produciendo así un espacio tridimensional rectangular ajustado a las dimensiones del mesón y con una relación de movimiento cercana a la escala 1:1.

Tabla 2-8. Límites de movimiento en \$BASE del Robot

Arista	Eje	Posición 1	Posición 2	[mm]
Línea 1	X	Inferior	Trasera	600,00
Línea 2	X	Superior	Trasera	600,00
Línea 3	X	Inferior	Delantera	600,00
Línea 4	X	Superior	Delantera	600,00
Línea 5	Y	Izquierda	Trasera	480,00
Línea 6	Y	Derecha	Trasera	480,00
Línea 7	Y	Izquierda	Delantera	480,00
Línea 8	Y	Derecha	Delantera	480,00
Línea 9	Z	Inferior	Izquierda	480,00
Línea 10	Z	Superior	Izquierda	480,00
Línea 11	Z	Inferior	Derecha	480,00
Línea 12	Z	Inferior	Izquierda	480,00

Fuente: Elaboración propia para el desarrollo y comprensión del movimiento del Robot

Cómo resultado final se obtiene la siguiente tabla de desplazamiento en los ejes cómo valores de tolerancia máxima dentro de las coordenadas “\$BASE 2 (ejemplo)”.

Tabla 2-9. Ajuste de OFFSET y posiciones límites para el manipulador en \$BASE

<i>Disposición final \$BASE 2 (ejemplo)</i>						
				X [mm]	Y [mm]	Z [mm]
Punto 1	Inferior	Izquierda	Frontal	80,00	0,00	0,00
Punto 2	Inferior	Derecha	Frontal	680,00	0,00	0,00
Punto 3	Superior	Izquierda	Frontal	80,00	0,00	480,00
Punto 4	Superior	Derecha	Frontal	680,00	0,00	480,00
Punto 5	Inferior	Izquierda	Trasero	80,00	480,00	0,00
Punto 6	Inferior	Derecha	Trasero	680,00	480,00	0,00
Punto 7	Superior	Izquierda	Trasero	80,00	480,00	480,00
Punto 8	Superior	Derecha	Trasero	680,00	480,00	480,00

Fuente: Elaboración propia para el desarrollo y comprensión del movimiento del Robot

Obteniendo estos datos se genera una corrección en el Script de MATLAB para proporcionar una coordenada adecuada al movimiento dentro del mesón para la base de trabajo ya creada cómo se puede ver en la siguiente figura.

```
% Correcciones de coordenada
% OFFSET KUKA == {X 80.0, Y 0.0, Z 0.0}
% Factores de corrección (X*0.9375) para 600[mm] de movimiento máximo
% Factores de corrección (Z*1) para 480[mm] de movimiento máximo
% Factores de corrección (Y*0.75) para 480[mm] de movimiento máximo
Xcorr = (X*0.9375)+80;
Ycorr = (Y*0.75);
Zcorr = Z;

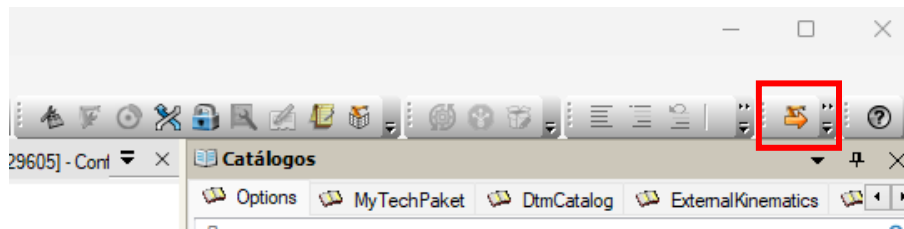
% Exportar coordenadas corregidas
Poskuka = sprintf('POS:<X>%g</X>,<Y>%g</Y>,<Z>%g</Z>}', Xcorr, Ycorr, Zcorr);
fwrite(t, Poskuka);
```

Fuente: Captura de pantalla del escalado y correcciones de la coordenada para exportar al Robot

Figura 2-32. Escalado y corrección de OFFSET en MATLAB para exportar coordenadas al Robot

De esta manera se establece una sincronía entre los datos a enviar desde Matlab una vez procesados y en los datos requeridos para trabajar en el programa del Robot KUKA.

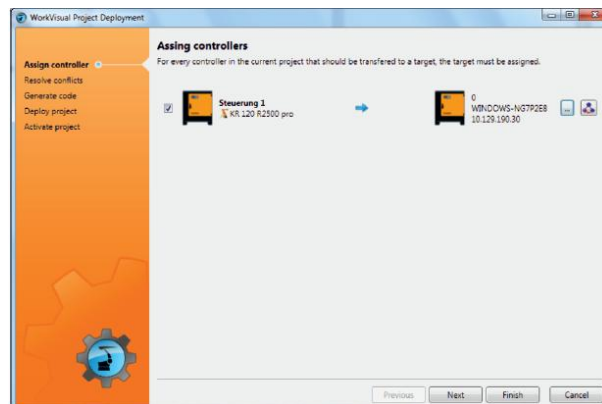
Una vez completada la programación del archivo “.src” en el proyecto se debe cargar al controlador del robot dando clic al icono de instalar en la barra de herramientas superiores, como se puede apreciar en la siguiente figura.



Fuente: Captura de pantalla de ordenador propio

Figura 2-33. Ícono de instalación del proyecto

Una vez dando clic se desplegará el siguiente menú para cargar el proyecto en el controlador del robot, automáticamente reconocerá al equipo en red y se procede con la carga.



Fuente: Captura de pantalla de ordenador propio

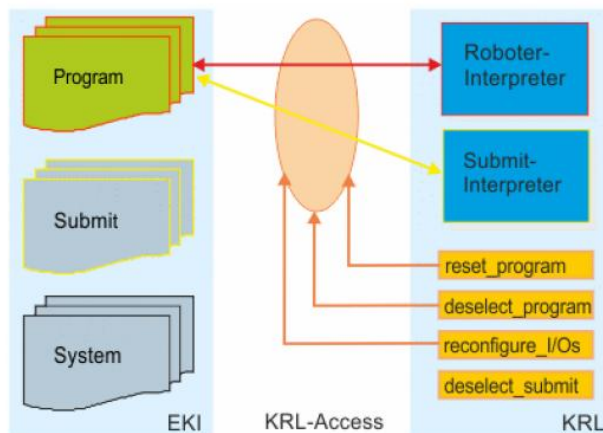
Figura 2-34. Carga del Proyecto a la célula Robot

Para que la carga se pueda ejecutar correctamente se deben establecer a través del SmartPad los permisos de acceso ingresando a un usuario de rango “Experto” o superior cómo “Administrador”, además el controlador debe encontrarse en los modos de trabajo “T1” o “T2”, no se puede iniciar la carga del proyecto si los modos de operación se encuentran en “AUT” o “EXT”, por ende, es necesario cumplir con ambas condiciones, por último se presentará un mensaje de confirmación en el SmartPad solicitando la validación de la carga del nuevo proyecto, al presionar “Si” se desplegará otra ventana indicando a grandes rasgos cuales son los cambios con respecto al proyecto actual, al aceptar nuevamente comenzará la carga del proyecto y se podrán ejecutar las nuevas programaciones.

2.6 KUKA ETHERNET KRL

En este apartado del proyecto se comentará el alcance e implementación correspondiente a la librería de comunicación KUKA ETHERNET KRL, que permite establecer un intercambio de datos a través de su interfaz propia Ethernet KRL Interface (EKI), esto con el fin de lograr una recepción y envío de datos XML a un sistema externo, así como también recepción y envío de datos binarios a un sistema externo.

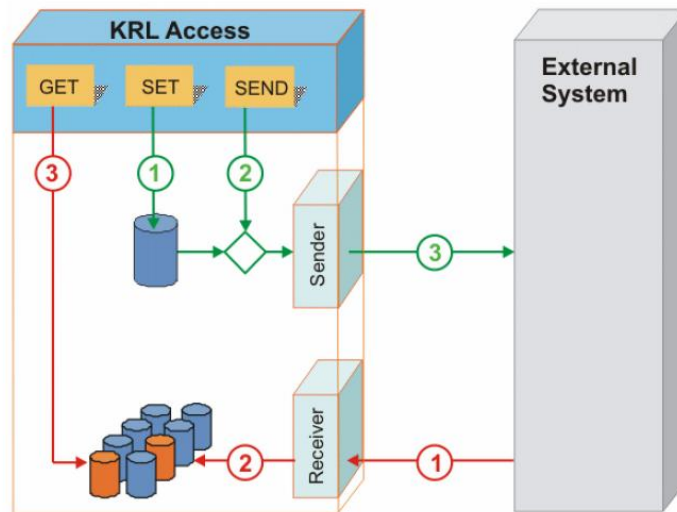
Para ello establece un sistema de comunicación TCP/IP o UDP/IP, en este proyecto se trabajará con TCP/IP, la unidad de control y el sistema externo pueden asumir ambos roles no simultáneamente, ya sea definir cómo servidor o cómo cliente al sistema externo o al controlador del robot.



Fuente: KST_Ethernet_KRL_31_es (Solo disponible al incorporar KUKA.EthernetKRL.kop a WorkVisual 6.0)

Figura 2-35. Diagrama del interpretador EKI a KRL

La configuración se realiza mediante un archivo basado en XML, se puede realizar de manera constante una revisión del sistema externo ejecutando comandos de Ping en la rutina de programación y esta se puede realizar dentro del interpretador "Submit", es decir en paralelo al trabajo del programa principal, además de poder ejecutarse en el programa principal, pero se recomienda utilizar el interpretador "Submit" para establecer una comunicación ininterrumpida durante el trabajo del programa principal.

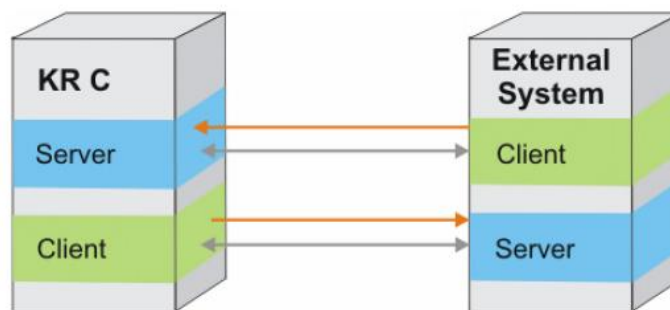


Fuente: KST_Ethernet_KRL_31_es (Solo disponible al incorporar KUKA.EthernetKRL.kop a WorkVisual 6.0)

Figura 2-36. Esquema de envío, recepción e interpretación de datos con EKI en KRL

2.6.1 Comunicación TCP/IP con MATLAB

Para definir la comunicación es necesario crear dos archivos de programación, el primero corresponde a un archivo de estructura XML que defina las características de la red y el rol del sistema externo cómo también el rol del controlador del robot, posteriormente se definen las características de la comunicación y los elementos que se envían y los que se reciben, esto se puede apreciar en la siguiente imagen.



Fuente: KST_Ethernet_KRL_31_es (Solo disponible al incorporar KUKA.EthernetKRL.kop a WorkVisual 6.0)

Figura 2-37. Diagrama de roles de comunicación TCP/IP

```

<ETHERNETKRL>
  <CONFIGURATION>
    <EXTERNAL></EXTERNAL>
    <INTERNAL></INTERNAL>
  </CONFIGURATION>
  <RECEIVE>
    <ELEMENTS></ELEMENTS>
  </RECEIVE>
  <SEND>
    <ELEMENTS></ELEMENTS>
  </SEND>
</ETHERNETKRL>

```

Fuente: KST_Ethernet_KRL_31_es (Solo disponible al incorporar KUKA.EthernetKRL.kop a WorkVisual 6.0)

Figura 2-38. Configuración estándar de archivo XML

Este archivo debe agregarse a una ruta específica del proyecto, de lo contrario no funciona, la ruta específica corresponde a la siguiente:

- “Cell/controlador/Config/User/Common/EthernetKRL

2.6.2 Programar XML para obtención y entrega de datos

La escritura del archivo XML para este proyecto se prueban distintos archivos con el fin de encontrar las mejores propiedades de envío y recepción de datos, una de estas configuraciones será llamada “EXT” y debe ser inicializada a través del interpretador “Submit” con las siguientes declaraciones, donde “EXT” dentro del paréntesis corresponde al nombre del archivo “.xml”.

$$EKI_STATUS = EKI_Init("EXT")$$

$$EKI_STATUS = EKI_OPEN("EXT")$$

Posterior a la iniciación de EKI se deben declarar las variables obtenidas con las funciones $EKI_STATUS = EKI_Get("EXT", "Tag\ Elemento", "Var")$, la función Get, tiene distintos sufijos según el tipo de variable que se requiere rescatar del archivo XML, es por ello que se pueden utilizar GetBool, GetFrame, GetReal, GetInt, según sea necesario.

```

<ETHERNETKRL>
  <CONFIGURATION>
    <EXTERNAL>
      <IP>172.31.1.140</IP>
      <PORT>60000</PORT>
      <TYPE>Server</TYPE>
    </EXTERNAL>
    <INTERNAL>
      <ENVIRONMENT>Program</ENVIRONMENT>
      <BUFFERING Mode="FIFO" Limit="10"/>
      <BUFFSIZE Limit="16384"/>
      <TIMEOUT Connect="60000"/>
      <ALIVE Set_Out="666" Ping="200"/>
      <PROTOCOL>TCP</PROTOCOL>
      <MESSAGES Logging="error" Display="disabled"/>
    </INTERNAL>
  </CONFIGURATION>
  <RECEIVE>
    <XML>
      <ELEMENTS>
        <ELEMENT Tag="Ext/Pos/XPos" Type="REAL"/>
        <ELEMENT Tag="Ext/Pos/YPos" Type="REAL"/>
        <ELEMENT Tag="Ext/Pos/ZPos" Type="REAL"/>
      </ELEMENTS>
    </XML>
  </RECEIVE>
  <SEND>
    <XML>
      <ELEMENTS>
        <ELEMENT Tag="Robot/Data/ActPos/@X"/>
        <ELEMENT Tag="Robot/Data/ActPos/@Y"/>
        <ELEMENT Tag="Robot/Data/ActPos/@Z"/>
      </ELEMENTS>
    </XML>
  </SEND>
</ETHERNETKRL>

```

Fuente: Captura de pantalla de ordenador propio del archivo XML de EKI

Figura 2-39. XML “EXT” para el traspaso de información entre MATLAB y EKI RKL

2.6.3 Integrar variables obtenidas a programación KRL

En esta parte de la programación se requiere haber asignado a una variable del sistema o llamar a una función del interpretador “Submit” para asignar las coordenadas a la variable del tipo “FRAME” del programa principal.

La integración de variables externas aún no se ha podido programar eficazmente, ya que aún se producen errores de comunicación y asignación de datos a las variables internas, por ende, la exploración de la intercomunicación entre un sistema externo como MATLAB y los procesos de

ejecución del Robot KUKA quedarán pendientes para la realización de un proyecto específico a desarrollar con posteridad.

Ya que en la actualidad la aplicación de estas funciones se encuentra sustentada por terceras aplicaciones desarrolladas principalmente para el este uso específico, cómo lo son los EKI_bridges o el desarrollo de API's en Python.

CAPITULO 3: EVALUACIÓN DE RECURSOS Y OPCIONES DE MEJORA

3 EVALUACIÓN DE RECURSOS Y OPCIONES DE MEJORA

En este capítulo se evaluarán los recursos económicos y humanos invertidos en la realización y finalización de este proyecto, tales como son los recursos existentes en la universidad, los recursos incorporados posteriormente, el tiempo invertido en el desarrollo de la actividad, además de las habilidades obtenidas y el aprendizaje relevante durante la realización del proyecto.

Cabe destacar que durante el desarrollo del proyecto se evidenciaron distintas maneras de lograr el objetivo, generando más opciones de trabajo y análisis de los factores relevantes que lograron incidir en la toma de decisiones sobre uso de tecnologías y programación de este mismo.

Por ello se presentarán algunas ideas de mejora con sus respectivos valores asociados dentro del marco estudiantil para llevar a cabo otro desarrollo de habilidades y competencias para los futuros alumnos que se enfrenten a este tipo de desafíos.

3.1 RECURSOS MATERIALES

En este primer apartado se enlistarán y evaluarán económicamente los recursos utilizados para concretar el proyecto, de esta manera se considerarán los recursos existentes en la universidad como una inversión previa que no influye en los costos asociados para el proyecto ni para sus mejoras.

En cambio, los recursos utilizados anexos o externos a la propiedad de la universidad serán detallados y evaluados con alternativas que pudiesen cumplir un mismo propósito con características similares o con mejoras en el proceso.

3.1.1 Recursos materiales de la universidad

En primera instancia se consideran los recursos existentes en la Universidad como una inversión previa con fines académicos y pedagógicos disponibles para el alumnado y para el

desarrollo de actividades a fines a la carrera dentro de un entorno Universitario, estos elementos se detallan en la siguiente tabla.

Tabla 3-1. Recursos de la universidad

Recursos Previos	
Robot KUKA Agilus R6 900 c/ mesón de trabajo	1 [u]
Cable de red CAT6	1[u]
Extensión monofásica 220[v]	1[u]
Adaptador "Chuco" 220[v]	1[u]
Cámaras de visión artificial Datalogic P20C	2 [u]
Software Matlab 2023a	1[u]
Software WorkVisual 6.0	1[u]
Librería Ethernet KRL 3.1	1[u]
Laboratorio espacioso con energía	1[u]

Fuente: Elaboración propia para el desarrollo del proyecto

Gracias a estos elementos presentes en las instalaciones de la Universidad y al cuerpo docente y paradocente que siempre se encuentra disponible para el apoyo de actividades de este estilo es que este proyecto pudo ser llevado a cabo.

3.1.2 Recursos materiales anexos y/o externos

Dentro de los elementos físicos y virtuales detallados con anterioridad se puede apreciar la demarcación diferenciada de las cámaras de visión artificial Datalogic P20C diseñadas para trabajos y entornos industriales, compatibles con trabajos relacionados al paletizado con brazos robóticos y diseñadas para la lectura de códigos a alta velocidad, porque lamentablemente se encontraban fuera de servicio, esto se descubrió debido a una inspección inicial al momento de la puesta en marcha y evaluación de funcionamiento, ya que no contaban con el lente óptico necesario para la captura de la imagen, por ende quedaron fuera del proyecto debido a la tardía respuesta por parte del proveedor y se consideró el uso de otras alternativas tales como el uso de cámaras web USB, aunque a pesar de rápida conexión con entornos informáticos su rendimiento distaba de las necesidades con respecto a la tasa de muestreo requerida por el Script

de MATLAB, por ende se consideró utilizar un dispositivo más robusto y con mejor desarrollo tecnológico, el Smartphone, y ya que actualmente todos tenemos uno, se utilizó para cumplir con los requerimientos del proyecto, es por ello que además se consideró el uso de dos trípodes para cumplir con la estabilidad de la captura de imágenes a través de las cámaras de los Smartphones.

Tabla 3-2. Recursos anexos y externos

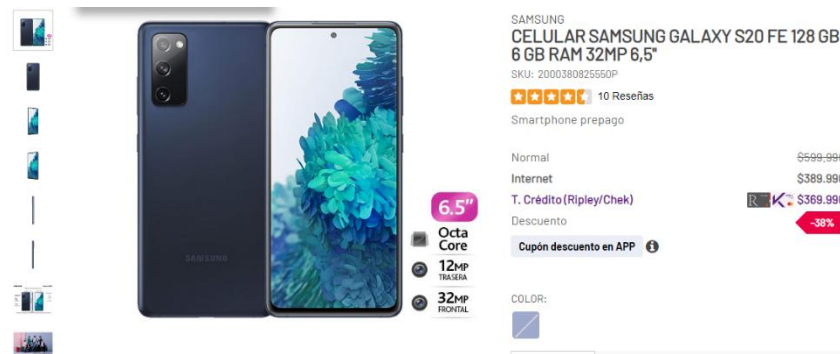
Recursos Anexos y Externos		Costo asociado p/u	Subtotal
Smartphones Samsung S20 FE	2 [u]	\$ 369.990	\$ 739.980
Trípodes	2 [u]	\$ 17.960	\$ 35.920
Software Droid Cam	1[u]	\$ -	\$ -
Software Iriun Webcam	1[u]	\$ -	\$ -
Notebook	1 [u]	\$ 599.990	\$ 599.990
		Total	\$ 1.375.890

Fuente: Elaboración propia para el desarrollo del proyecto

Analizando los resultados de la tabla anterior, se puede apreciar el costo total correspondiente a la adquisición de elementos nuevos, pero este caso no representa la realidad, ya que actualmente cada alumno posee un smartphone con el cual puede sacar provecho de las tecnologías y los softwares de desarrollo para experimentar y obtener conocimientos enriquecedores que servirán para el resto de su desarrollo académico y laboral, si se considera un trabajo colaborativo entre dos alumnos el costo aplicado a los Smartphones se reduce a \$0 pesos, y solo se debe considerar el gasto de los trípodes para la sujeción de estos.

Además, se puede implementar un ordenador de la universidad en un espacio adecuado para el trabajo directo con el Robot KUKA, pero en este caso se prescindió de ello ya que se utilizó un ordenador portátil propio.

Los softwares enunciados son gratuitos y compatibles con sistemas operativos Android e iOS, además de poder ser instalados en ordenadores con sistema operativo Windows sin ningún problema.



Fuente: <https://simple.ripley.cl/celular-samsung-galaxy-s20-fe-128-gb-6-gb-ram-32mp-65-2000380825550p?s=mdco>

Figura 3-1. Cotización Samsung S20 Fe



Fuente: https://articulo.mercadolibre.cl/MLC-2224039060-tripode-celular-camara-portatil-soportes-carga-5kg-140-cm-JM#position%3D34%26search_layout%3Dstack%26type%3Ditem%26tracking_id%3D23286111-2234-4b92-862b-b8ad16771eb0

Figura 3-2. Cotización Trípodes



Fuente: https://simple.ripley.cl/notebook-gamer-hp-victus-15-fa1010la-intel-core-i5-8-gb-ram-512-gb-ssd-nvidia-rtx-2050-156-2000399428124p?gad_source=1&gclid=CiwKCAjw_Na1BhAlEiwAM-dm7Jb_quPXwgz0o2sKPaxLnVlvTfE0a9fEqS0i0tVCRG-t9ziFF4A4jhoCPI0QAvD_BwE&color_80=azul-marino&s=mdco

Figura 3-3. Cotización Notebook

3.2 RECURSOS HUMANOS

El análisis de este apartado es algo especial, ya que gran parte de este proyecto se dedicó al estudio de las nuevas estructuras de programación, la utilización del Robot KUKA Agilus R6 900, aplicaciones nuevas formas de trabajo dentro del software MATLAB, estudio sobre la comunicación TCP/IP, análisis de distintas posibilidades y alternativas para desarrollar el trabajo, aprendizaje sobre el uso de nuevos softwares como WorkVisual 6.0 y Hercules para establecer un ecosistema de trabajo adecuado a las necesidades del proyecto ejecutando pruebas de comunicación y pruebas de programación, intentando cumplir con todos los objetivos planteados al inicio de la actividad, a continuación se muestra en la siguiente tabla en cantidad de horas de estudio correspondiente a cada área para la comprensión de todos los elementos incorporados en este proyecto.

Tabla 3-3. Recursos humanos utilizados en horas de estudio

Recursos humanos en estudio [horas]	
Robot KUKA Agilus R6 900	20
Visión Artificial	10
KUKA Sunrise toolbox (MATLAB)	5
IMAGE ACQUISITION toolbox (MATLAB)	10
Comunicación TCP/IP	10
Lenguaje de programación XML	10
KUKA.PLC.mxAutomation (PLC)	20
Lenguaje de programación KRL	20
KUKA Ethernet KRL	20
KUKA WorkVisual 6.0	15
Total [horas]	140

Fuente: Elaboración propia para el desarrollo del proyecto

Los datos reflejados en la tabla cómo se comentó inicialmente corresponde al tiempo invertido en estudio para llevar a cabo el proyecto, ya que las tecnologías son nuevas y no se contemplan dentro de la malla curricular la mayoría del trabajo fue de investigación y desarrollo, dentro del detalle anterior se pueden apreciar coloreadas de color amarillo dos tecnologías que fueron estudiadas cómo posibilidades de desarrollo del proyecto, pero que por viabilidad y por

factibilidad de obtención de las mismas fueron descartadas para esta ocasión, pero si serán evaluadas cómo futuras oportunidades de mejora e implementación dentro del campo de la robótica, la primera correspondiente a la librería “KUKA Sunrise toolbox” corresponde a una librería de MATLAB desarrollada por KUKA para la comunicación directa con los controladores Sunrise del Robot LBR iiwa, pero para este caso el Robot KUKA Agilus R6 900 posee un controlador KRC4 que no es compatible con las instrucciones de este paquete.

El segundo elemento destacado de la lista corresponde a la librería para el software WorkVisual, correspondiente a “KUKA.PLC.mxAutomation” siendo esta la opción más atractiva para el desarrollo del proyecto y además desde el punto de vista académico para la utilización del Robot cómo recurso educativo, ya que contempla el control total del Robot a través de un PLC Siemens S7-1500 con su respectiva interfaz de programación dentro de TIA PORTAL con una librería propia y acceso a un amplio catálogo de manipuladores, pero por condiciones económicas se descartó por su elevado valor comercial.

Aparte es necesario cuantificar la cantidad de horas trabajadas en el desarrollo del proyecto, lo que contempla diseño, programación pruebas y ejecución, detallados en cantidad de horas en la siguiente tabla.

Tabla 3-4. Recursos humanos utilizados en el desarrollo del proyecto

Recurso humano desarrollo del proyecto [horas]	
Script MATLAB	50
TIA Portal PLC S7-1500	10
Revisión cámaras industriales	4
Virtualización 3D	20
Pruebas TCP/IP Hercules	5
Creación \$BASE KUKA	4
Proyecto en WorkVisual 6.0	10
Programación KRL	8
Programación XML + Pruebas	15
Total [horas]	126

Fuente: Elaboración propia para el desarrollo del proyecto

De igual manera se añadió el trabajo realizado en TIA Portal, aunque este no se utilizara concretamente en el desarrollo del proyecto, pero sí sugirió una importante cantidad de horas destinadas a la integración de librerías y pruebas realizadas para una posible programación y ejecución del proyecto.

3.3 OPCIONES DE MEJORA

Cómo opciones de mejora al proyecto para darle un enfoque más académico se puede considerar la obtención de cámaras de visión artificial adecuadas para los procesos industriales, además de poder incorporar la librería de programación “KUKA.PLC.mxAutomation” ya antes mencionada para trabajar en un entorno más familiar, ya que la programación de movimientos y rutinas de operación se ejecutarían dentro de TIA Portal, y las configuraciones realizadas con WorkVisual serían solo correspondientes a la estructura del proyecto, mas no a la programación de rutinas de movimiento, obtención de datos, ni procesamiento de variables, trabajando con una interfaz más amigable en el desarrollo de programación en lenguaje Ladder con bloques de procesos ya automatizados, donde se podrían crear un sinnúmero de actividades relacionadas ligadas al área de la robótica y la visión artificial.

Para ello se propone el siguiente listado de elementos con sus valores respectivos en el mercado para una posible obtención de licencias y componentes para un futuro desarrollo de actividades con el Robot KUKA Agilus R6 900 que se encuentra en la universidad.

Tabla 3-5. Cotización de elementos para opciones de mejora del proyecto

Opciones de mejora		Proveedor	Valor p/u	Subtotal
Cámara Datalogic P20C	2	Delind Electronic	\$ 3.215.116	\$ 6.430.232
KUKA.PLC.mxAutomation	1	KUKA	\$ 2.024.812	\$ 2.024.812
ThinkCentre Neo 50s 4ta Gen	1	Lenovo	\$ 1.199.990	\$ 1.199.990
Monitor ThinkVision S27i-30 27"	1	Lenovo	\$ 199.990	\$ 199.990
PLC S7-1500	1	wiautomation	\$ 514.588	\$ 514.588
			Total	\$ 10.369.612

Fuente: Elaboración propia para el desarrollo del proyecto

Estos elementos pueden representar una mejora sustentable para el desarrollo de actividades relacionadas a la programación del Robot KUKA Agilus R6 900 disponible en el laboratorio de electrotecnia de la Universidad.



Fuente: <https://www.delind.cl/product/smart-camara-p20c-000-000-ml-sku-da40-12601>

Figura 3-4. Cotización cámara industrial Datalogic P20C



Fuente: <https://my.kuka.com/s/product/kukaplc-mxautomation-33/01t1i000002lqQdAAI?recordName=kukaplc-mxautomation-33>

Figura 3-5. Cotización paquete KUKA.PLC.mxAutomation 3.3

ThinkCentre Neo 50s 4ta Gen - Black

★★★★★ 4.5 (39)

Número de parte: 12JE000TCS

Ahorra \$200.000 14% de descuento

Precio Web ~~\$1,399,990~~

\$1,199.990 Promociones formas de pago \$99.999/mes en 12 cuotas sin interés. [Ver cuotas](#)

IVA Inc.

Agregar al Carrito



Fuente: [https://www.lenovo.com/cl/es/p/computadoras-de-escritorio/thinkcentre/thinkcentre-serie-neo/thinkcentre-neo-50s-gen-4-sff-\(intel\)/12je000tcs?cid=cl:sem:pmax|se|google|pmax+top+roas|||es_CL12JE000TCS|18133386624|||pmax|mixed|all&gad_source=1&gclid=CjwKCAjw2dG1BhB4EiwA998cqGWzcVonbfKY0IQ7QP2MAMRPLG4Glcml7iIMolcl7qtYy9To0EnhoCI9oQAyD_BwE](https://www.lenovo.com/cl/es/p/computadoras-de-escritorio/thinkcentre/thinkcentre-serie-neo/thinkcentre-neo-50s-gen-4-sff-(intel)/12je000tcs?cid=cl:sem:pmax|se|google|pmax+top+roas|||es_CL12JE000TCS|18133386624|||pmax|mixed|all&gad_source=1&gclid=CjwKCAjw2dG1BhB4EiwA998cqGWzcVonbfKY0IQ7QP2MAMRPLG4Glcml7iIMolcl7qtYy9To0EnhoCI9oQAyD_BwE)

Figura 3-6. Cotización WorkStation

Monitor ThinkVision S27i-30 27" - Storm Grey

★★★★★ 4.2 (7)

Número de parte: 63DFKAR4CL

Ahorra \$100.000 **33% de descuento**

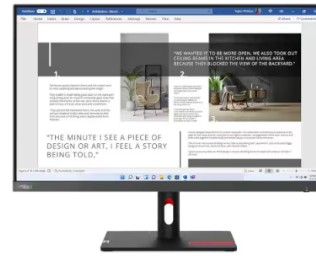
Precio Web ~~\$299.990~~

\$199.990

IVA Inc.

Promociones formas de pago
\$16.666/mes en 12 cuotas sin interés. [Ver cuotas](#)

Agregar al Carrito



Fuente: <https://www.lenovo.com/cl/es/p/accessories-and-software/monitors/office/63dfkar4cl>

Figura 3-7. Cotización Monitor 27" para WorkStation



S7-1500 Siemens SKU 6ES75111AK020AB0

6ES7511-1AK02-0AB0 SIEMENS

13 EN ESTOCK Entrega urgente

Nuevo - 12 Meses de Garantía

CLP 514,558.00

— 1 +



Agregar

[Calcular costo de envío](#)

[Solicitud de cotización](#)

Fuente: [https://cl.wiautomation.com/siemens/plc-sistemas/simatic-s7/s7-](https://cl.wiautomation.com/siemens/plc-sistemas/simatic-s7/s7-1500/6ES75111AK020AB0?utm_source=google&utm_medium=cpc&utm_campaign=CL_pmax_full&gad_source=1&gclid=CjwKCAjw2dG1BhB4EiwA998cqIPOC-eCC4RS3y40P-TaEzI ndEZuk2Z060GtXxoTgQNXQ-bIC8D1BoChToQAvD BwE)

[1500/6ES75111AK020AB0?utm_source=google&utm_medium=cpc&utm_campaign=CL_pmax_full&gad_source=1&gclid=CjwKCAjw2dG1BhB4EiwA998cqIPOC-eCC4RS3y40P-TaEzI ndEZuk2Z060GtXxoTgQNXQ-bIC8D1BoChToQAvD BwE](https://cl.wiautomation.com/siemens/plc-sistemas/simatic-s7/s7-1500/6ES75111AK020AB0?utm_source=google&utm_medium=cpc&utm_campaign=CL_pmax_full&gad_source=1&gclid=CjwKCAjw2dG1BhB4EiwA998cqIPOC-eCC4RS3y40P-TaEzI ndEZuk2Z060GtXxoTgQNXQ-bIC8D1BoChToQAvD BwE)

Figura 3-8. Cotización PLC Siemens S7-1500

CONCLUSIONES

El proyecto de integración de sistemas entre cámaras de visión artificial y el Robot KUKA Agilus R6 900, fue un desafío provechoso que permitió exprimir en su totalidad los conocimientos y capacidades aprendidas durante la carrera, de tal manera que instó a la obtención de información de diferentes fuentes fidedignas y confiables para establecer un ritmo de crecimiento en cuanto a conocimientos que permitió enfrentar cada obstáculo presentado a lo largo del proyecto, fue una valiosa experiencia en donde se pudo evidenciar el arduo trabajo de un ingeniero en cuanto debe enfrentarse al diseño de nuevas plataformas de trabajo, a la innovación constante y el desarrollo de nuevas tecnologías, la adaptación a los cambios y la resiliencia con respecto a los errores y los problemas.

BIBLIOGRAFÍA

Seminarios de instrucción KUKA por Carolina Fuster en YouTube:

https://www.youtube.com/watch?v=8WiOL1F6_iE&t=1173s

Modelo de color HSV:

https://es.wikipedia.org/wiki/Modelo_de_color_HSV

Manual del programador experto KUKA Robots:

<https://www.infopl.net/descargas/73-kuka/2263-programacion-experta-robot-kuka-kr-c>

Seminario Web KUKA.Ethernet.KRL:

<https://www.youtube.com/watch?v=Ne13sBHPGv4&t=2018s>

Seminario Web mxAutomation in TIA Portal

https://youtu.be/eI10cOjD1JU?si=bl_jF3zY6oiu_d7o

Manual KUKA_smartPAD_2_EN.pdf

Manual KST_Ethernet_KRL_31_es.pdf

Manual KUKA_PB_CONTROLLERS_EN.pdf

Enlaces de ayuda de MATLAB:

IMAGE ACQUISITION toolbox

TCP/IP server & client