UNIVERSIDAD DEL SINÚ
Elías Bechara Zainúm
Seccional Cartagena

SISTEMA DE CAPTURA DE MOVIMIENTO EN MIEMBRO SUPERIOR MEDIANTE CAPTURA DE VIDEO Y SENSORES INERCIALES

PRESENTADO POR:

CRISTIAN GUANCHA CERÓN

JORGE HENRIQUE ZARUR AYOLA

UNIVERSIDAD DEL SINÚ ELIAS BECHARA ZAINUM

FACULTAD DE CIENCIAS EXACTAS E INGENIERIA

ESCUELA DE INGENIERIA DE SISTEMAS

CARTAGENA, COLOMBIA

NOVIEMBRE 2019

SISTEMA DE CAPTURA DE MOVIMIENTO EN MIEMBRO SUPERIOR MEDIANTE CAPTURA DE VIDEO Y SENSORES INERCIALES

Trabajo de grado presentado como requisito para optar al título de

INGENIERO DE SISTEMAS

Asesor disciplinar

LUIS FERNANDO MURILLO FERNANDEZ

Asesor metodológico

EUGENIA ARRIETA RODRIGUEZ

UNIVERSIDAD DEL SINÚ ELÍAS BECHARA ZAINÚM

FACULTAD DE CIENCIAS EXACTAS E INGENIERÍA

ESCUELA DE INGENIERÍA DE SISTEMAS

CARTAGENA-COLOMBIA

NOVIEMBRE DE 2019

TABLA DE CONTENIDO

TABLA DE ILUSTRACIÓNES

TABLAS

RESUMEN

El presente proyecto tiene como propósito desarrollar dos sistemas de captura de movimiento del miembro superior, mediante tecnologías de bajo costo, utilizando, por una parte, grabación de movimiento MOCAP (Motion capture) y por otra, la captura de movimiento mediante sensores inerciales. Estos dos sistemas desarrollados podrán servir posteriormente como dos fuentes de datos que puedan ser fusionadas en trabajos futuros para obtener mejores resultados del movimiento del brazo.

Este proyecto fue llevado a cabo para facilitar la adquisición de sistemas de captura de movimiento con una precisión superior y a bajo costo, y de esta manera se logren realizar más aportes en investigación para facilitar el acceso de estas tecnologías a cualquier industria que requiera o necesite de un sistema similar. Además, con este proyecto se dará continuidad a un proyecto macro que se encuentra en desarrollo en la Universidad del Sinú seccional Cartagena relacionado al problema de seguridad que se dio con el avance de la tercera revolución industrial al momento en que máquina y humano empezaron a trabajar juntos en un entorno colaborativo. Estos sistemas de captura de movimiento se desarrollaron utilizando el lenguaje de programación Python y las extensas librerías disponibles que facilitan el desarrollo de este tipo de proyectos. Una vez desarrollados los sistemas, se realizaron pruebas de validación contrastando los resultados con herramientas de medida angular.

INTRODUCCIÓN

Actualmente existen sofisticados métodos para la captura del movimiento, estos son utilizados en diversas industrias, como la industria cinematográfica para realizar animaciones y colocar a los actores en los lugares poco convencionales o en la medicina, para estudiar la biomecánica del cuerpo. Estos sistemas, aunque ofrecen relativamente un alto grado de precisión de las características del movimiento de un ser humano, también poseen un alto costo. Un campo importante donde estos sistemas son utilizados es la robótica colaborativa e industria 4.0, que a su vez están ligadas al campo de la visión por computadora. Esta ciencia facilita obtener información de los objetos en un determinado entorno, pero no solo se utiliza visión por computadora para dotar máquinas de inteligencia y que sean capaces de tomar decisiones, también son utilizados sensores inerciales para obtener información del entorno que posteriormente puede ser procesada. Este proyecto nace como la continuidad de una investigación de captura de movimiento mediante video que se basa en la medición de pixeles, que ha sido ajustado para obtener resultados en un sistema métrico real (milímetros) y que junto al algoritmo de sensores inerciales contribuirán posteriormente a la creación de un algoritmo de fusión para dar una solución más confiable y precisa. La gran variedad de algoritmos y técnicas de inteligencia artificial que existen hoy en día para lograr la integración de diferentes fuentes de datos, brindan la posibilidad de obtener resultados más precisos.

1. DISEÑO METODOLÓGICO

1.1. PLANTEAMIENTO DEL PROBLEMA

1.1.1. Descripción del problema

La captura de movimiento y la animación computada hoy en día se ha convertido en una herramienta indispensable en múltiples campos e industrias, para producciones cinematográficas, la industria de video juegos, estudios de anatomía humana, etc. Pero el uso de esta tecnología no solo se limita al entretenimiento, también pueden usarse para realizar simulaciones con fines de investigación como, por ejemplo, en medicina, un equipamiento de captura permite a un kinesiólogo estudiar el movimiento de una persona, reproducirlo múltiples veces y desde distintos ángulos, medir velocidad, fuerza, etc. Puede ser usado también para fines deportivos, como aprender movimientos en distintas disciplinas como lo son las artes marciales.

A lo largo de los años, se han logrado avances importantes en las técnicas de captura de movimiento y existen diferentes técnicas para realizarlo, como lo es MOCAP y la captura de movimiento a través de sensores inerciales. La problemática radica en que obtener acceso a este tipo de sistemas solo está al alcance de grandes compañías de producción de video o personas con un gran alcance adquisitivo, una prueba de esto es que solo obtener un sensor inercial de calidad profesional con una empresa de la industria como lo es XSENS, a la fecha (marzo de 2019) cuesta alrededor de tres millones de pesos colombianos [1], Lo cual es un precio muy elevado y que no propicia la adquisición de estos sistemas. Las personas del común o estudiantes de ingeniería, producción de video, muy difícilmente pueden realizar proyectos de calidad superior al no poder contar con los recursos necesarios para la adquisición de estos productos. Todavía existen muchos campos en los que la captura de movimiento no ha sido utilizada y para los que significaría un gran diferencial, ya que se podrían constituir grandes

innovaciones con el desarrollo de proyectos de investigación en universidades que permitiera a estudiantes y graduados utilizar estas tecnologías de última generación en la búsqueda de nuevas ideas que contribuyan a mejorar nuestra calidad de vida. Mas específicamente actualmente existe una problemática entorno a la robótica colaborativa, lo cual hace referencia a robot y humano trabajando en un mismo entorno, la idea de personas laborando junto a máquinas cuya fuerza en ocasiones suele ser muy superior, trae consigo la consecuencia de riesgos en el espacio de trabajo, debido a la probabilidad de accidentes que puedan causar lesiones, muertes, pérdida de producción y tiempo. Hasta la liberación de la recomendación ISO/TS-15066 en 2016, esta interacción estaba muy limitada, debido a que la norma ISO 10218 restringe la presencia de personas en una celda de operación de robots, instalados en una ubicación industrial, esto debido a los daños que puede causar un robot a una persona si llegara a impactar contra ella, daños que van desde golpes y aplastamientos, pasando por cortes, perdida de miembros y en algunos casos hasta causar la muerte.

1.1.2. Justificación

Una de las razones principales para el desarrollo de este proyecto consiste en utilizar sensores inerciales y captura de movimiento MOCAP, es posible aumentar la precisión de los datos obtenidos y lograr resultados similares a los que podría generar un sistema de captura de movimiento profesional, pero a un costo mucho menor. En efecto, uno de los inconvenientes más grandes en la masificación del uso de estas tecnologías, es el costo de los sistemas, debido al uso de sensores, cámaras, y en general, hardware de muy altas especificaciones y alto precio. Siendo esta la motivación principal para el desarrollo este proyecto, con el propósito de que estas tecnologías puedan ser utilizadas y adquiridas por más personas con intereses investigativos.

Además, este proyecto de captura de movimiento de miembro superior sería una base para la realización de nuevos proyectos que abarquen temas como la problemática de la robótica colaborativa, el cual tendría como solución el lograr que las máquinas identifiquen la posición en el espacio de trabajo de las personas, con el fin de poder anticiparse a los movimientos del individuo, y de esta manera el dispositivo tome acciones que eviten una colisión o impacto, cuando exista la aproximación de una parte del cuerpo de la persona a la máquina.

Por otra parte, la generación de estos dos sistemas de captura de movimiento servirá de insumo para un algoritmo que realice la fusión de los datos a partir de modelos biomecánicos y el movimiento motor del cuerpo humano (Human Motor Control), con el fin de obtener un único flujo de datos que permita construir un sistema de captura con garantía de seguridad verificable o estandarizada, con estimación del retraso y caracterización de las incertidumbres asociadas a la detección del movimiento.

1.1.3. Formulación del problema

¿Cómo capturar el movimiento del miembro superior, mediante la captura de video y sensores inerciales, con tecnologías de bajo costo y con un error relativo porcentual absoluto bajo?

1.1.4. Alcance

El presente proyecto se enmarca en un macroproyecto que pretende conseguir el desarrollo de un algoritmo de fusión de datos para un prototipo de sistema de captura de movimiento de miembro superior mediante vídeo y sensores inerciales en sistemas de seguridad para Robótica Colaborativa. Este proyecto es una parte fundamental del anterior concepto global. En este caso, el proyecto se centra en desarrollar dos sistemas de captura de movimiento que arrojen las posiciones y ángulos generados por las articulaciones del brazo, captar el movimiento, procesar los datos y enviar los datos serializados a un algoritmo de fusión; de esa manera, se tendrá el sistema con captura de vídeo y sensores, los datos serán preprocesados, pasarán por el algoritmo donde serán fusionados y se obtendrá un paquete de datos con las posiciones del brazo y los ángulos generados. El sistema de captura de movimiento mediante video ya ha sido desarrollado, y es la base de partida para uno de los algoritmos que se desarrollan en este proyecto donde lo que se busca es ajustar los datos generados a un formato adecuado para la fusión.

Para la toma de los datos se tendrá a una persona realizando movimientos y se irán recibiendo en tiempo real, estos datos podrán ser visualizados a través de las aplicaciones. Los datos que entregue cada sistema van a ser almacenados, para posteriormente ser validados contra instrumentos de medida angular y de esta manera determinar su precisión; pero para tener pruebas válidas, y siguiendo lo establecido por los investigadores del tema a nivel internacional, se recomienda

contrastar los resultados obtenidos con sistemas profesionales de captura de video con marcadores tales como Vicon, disponibles en laboratorio biomecánicos especializados ubicados en la universidad de Oviedo, España.

No es parte de este proyecto desarrollar una interfaz para la ejecución de los diferentes módulos, ni desarrollar un algoritmo de fusión, sino preparar los dos sistemas para su posterior integración. Tampoco hace parte del alcance de este proyecto contrastar los datos obtenidos con sistemas profesionales.

1.1.5. Objetivos

Objetivo General.

Desarrollar un sistema de captura de movimiento del miembro superior mediante captura de video y sensores inerciales, utilizando tecnologías y recursos de hardware de bajo costo.

Objetivos Específicos.

- Analizar las técnicas de captura de movimiento mediante sensores inerciales y sistema de captura de video para determinar las principales variables de interés y su funcionamiento
- Desarrollar un módulo en Python de preprocesamiento para cada sistema que facilite el ajuste de los datos generados por los sistemas de captura de movimiento a un formato adecuado para un algoritmo de fusión.
- Determinar la precisión de los sistemas de captura de movimiento contrastando los datos con instrumentos de medida angular.

1.2. Estado del arte

A lo largo del tiempo se han desarrollado diferentes proyectos de investigación con relación a la captura de movimiento y fusión de datos a nivel local, nacional e internacional. En la siguiente tabla, se nombran los trabajos que guardan mayor relación con los objetivos del presente proyecto:

Tabla 1 *Estado del arte*

IDENTIFICACIÓN	OBJETIVO GENERAL	CATEGORIAS / VARIABLES	INSTRUMENTOS RECOLECCIÓN DE LA INFORMACIÓN	RESULTADOS
Alessandro Filippeschi, Norbert Schmitz, Markus Miezal, Gabriele Bleser, Emanuele Ruffaldi, Didier Stricker. **"Survey of Motion Tracking Methods Based on Inertial Sensors: A**	Investigar los métodos de seguimiento del movimiento basados en sensores inerciales en la extremidad superior.	Cinemática, Fusión de sensores, rastreo de movimiento y unidades de medidas inerciales.	Encuestas, revisión del estado del arte y consulta de medios electrónicos.	Después de una encuesta sobre rastreo humano basado en IMU, se seleccionaron cinco técnicas para la reconstrucción del movimiento y se compararon para reconstruir un movimiento del brazo humano. La estimación basada en IMU se comparó con

Focus on Upper Limb Human Motion". Kaiserslautern, Germany, 2017				el seguimiento de movimiento basado en el sistema de seguimiento de movimiento basado en marcador Vicon considerado como verdad en el terreno. Los resultados muestran que todos menos uno de los modelos seleccionados tiene un rendimiento similar (error de estimación de posición promedio de aproximadamente 35 mm).
Frey Giovanni Zambrano	Desarrollar, simular e	Robótica colaborativa,	Estudio bibliográfico	En este trabajo de investigación

Pinilla. **“Simulación, integración e implementación de un algoritmo de robótica colaborativa para dos robots móviles de arquitectura homogénea**” Universidad Militar Nueva Granada, Bogotá D.C, 2018	implementar una estrategia de trabajo en robots móviles de arquitectura homogénea para el desarrollo de una tarea colaborativa, implementándola en un programa de simulación de robots (Webots) y luego de forma física utilizando robots reales.	agentes, robótica de servicio, sistemas homogéneos, inteligencia artificial.	, Pruebas de concepto y entrevistas.	se diseñó, implementó y desarrolló un sistema multi-agente que utiliza un algoritmo como estrategia, para el trabajo de dos robots móviles homogéneos, en el desarrollo de una tarea de forma colaborativa.
Lesly Lisbeth Gómez Echeverry, Anyi Melissa Jaramillo Henao, Madeleine **“Sistemas de captura y**	se realizó una revisión sistemática de literatura científica a nivel global, siguiendo los parámetros de las metodologías	Cinemática del movimiento humano; Captura del movimiento; Animación 3D; Biomecánica	Revisión literaria; entrevistas	Los sistemas de captura de movimiento se deben elegir según su aplicación. En el caso de los estudios que requieran gran precisión e

análisis de movimiento cinemático humano: una revisión sistemática” Centro de Servicios y Gestión Empresarial, Tecnoparque Nodo Medellín, Servicio Nacional de Aprendizaje SENA, Medellín, Colombia	PRISMA y PRISMA P-2015 sobre las numerosas investigaciones que involucran el movimiento humano principalmente en las ciencias biomédicas, del deporte y animación 3D. Determinando las ventajas, limitaciones y aplicaciones de cada una de ellas.	.		indagar sobre mecanismos de movimiento complejos, lo mejor es utilizar las tecnologías ópticas con marcadores como Vicon o BTS. El sistema Vicon lidera también las investigaciones en animación 3D, realidad virtual y realidad aumentada. Si son aplicaciones en biomecánica, telerehabilitación y rehabilitación de menor complejidad, se puede recurrir a la tecnología Kinect sin marcadores. Finalmente, para investigaciones

				que se requieran realizar al aire libre y sin restricciones de movimiento, la mejor opción son los sistemas inerciales y magnéticos
Néstor Manuel Alemán Soler. "**Solución de bajo coste de captura de movimiento basada en Kinect**". Universidad de las Palmas de Gran Canaria, Julio 2014.	Implementar un sistema de captura de movimiento de bajo coste, usando, como dispositivo de captura, el Kinect de Microsoft. Implementar la solución MOCAP con el software de animación Blender, así como elaborar un tutorial del uso de Blender	Captura de movimiento; modelos 3D; captura de movimiento de bajo costo.	Revisión del estado de arte de la captura de movimiento ; Pruebas de concepto.	se encontró la manera de realizar captura de movimiento sin tener que disponer de una tecnología muy cara. Como se ha podido comprobar en este trabajo, el dispositivo Kinect es un dispositivo que no sólo es válido para juegos

	para este propósito.			

1.3. MARCOS DE REFERENCIA

1.3.1. Marco teórico

A continuación, se mencionarán los diferentes fundamentos teóricos que guardan relación con el presente proyecto y también serán mencionadas las herramientas a utilizar para la implementación.

Caracterización del miembro superior y sus movimientos asociados.

El cuerpo humano es un gran sistema complejo en todos los sentidos. Como el objetivo del presente proyecto es el desarrollo de dos sistemas de captura de movimiento (video y sensores inerciales) del miembro superior, es necesario conocer la estructura y el comportamiento del miembro superior

Miembro superior

La estructura del miembro superior humano está constituida en tres segmentos principales: el hombro, el codo y la mano como se muestra en la ilustración 1.

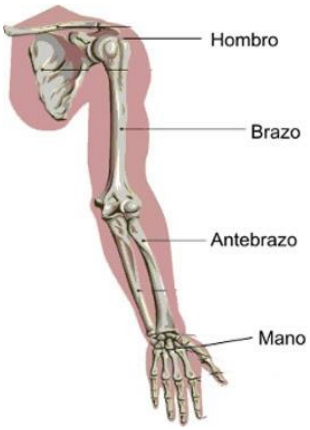


Ilustración 1 *Estructura del Miembro Superior. (Tomado de: http://principaleshuesos2colmanare2010.blogspot.com/2010/08/brazo.html)*

Tabla 1. Grupos musculares del miembro superior.

Musculo	Función
Músculos del hombro Deltoides (grupo anterior y grupo posterior)	Abducción, aducción, flexión, extensión y rotación del brazo.
Músculos del brazo Bíceps braquial	Elevador y aductor del brazo; flexor y supinador del codo.
Músculos del antebrazo	Pronador y flexor del codo
Músculos de la mano	Abducen y flexionan los dedos

Articulaciones del brazo

Las articulaciones son zonas de unión entre los huesos o entre cartílagos del esqueleto y permiten la movilidad de las diferentes extremidades del cuerpo humano. La extremidad de estudio para este proyecto es el brazo, el cual está compuesto por hombro, codo y muñeca.

Es necesario conocer el rango de movimiento de las articulaciones del brazo para el desarrollo de este sistema de fusión. El hombro tiene una flexión de 180 grados, una extensión de 45 grados, rotación externa e interna de 90 grados y una abducción y aducción de 180 grados.

La articulación del codo tiene una flexión de 145 grados, extensión de 0 grados y el antebrazo una supinación de 90 grados al igual que la pronación. La articulación de muñeca tiene una inclinación radial de 20 grados, inclinación cubica de 45 grados, flexión dorsal de 70 grados y flexión palmar de 80 grados [2].

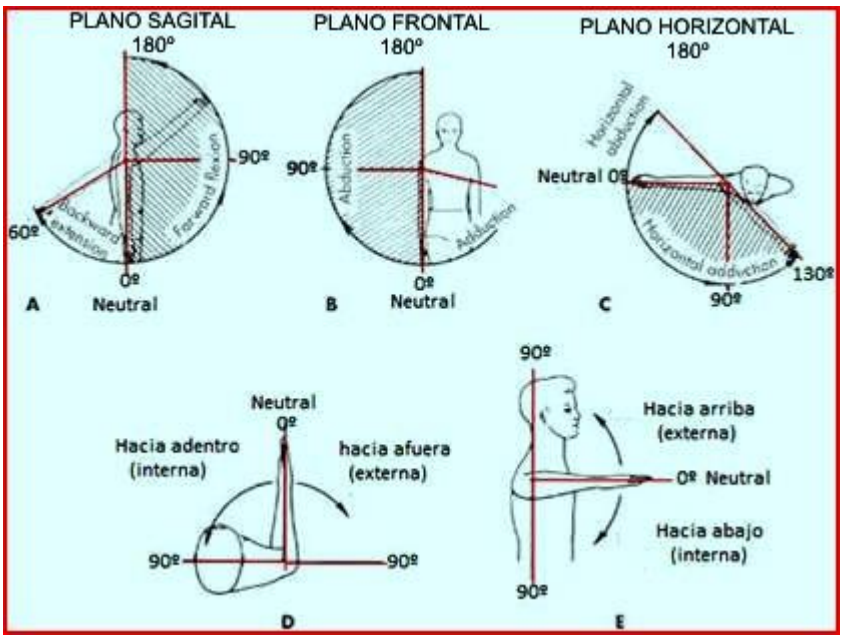


Ilustración 2 *Amplitud de la flexión / extensión del brazo (Tomado de: http://www.cto-am.com/hombroexploracion.htm).*

Algoritmo: En computación, un algoritmo es un conjunto de pasos establecidos que permiten resolver un problema a través de una computadora. Este algoritmo debe ser fácil de transcribir a un lenguaje de programación, para esto se utilizan herramientas que permiten realizar algoritmos escritos en un lenguaje entendible para el ser humano [3].

Un algoritmo debe contar con las siguientes características:

1. Precisión. Debe ser específico en cuanto al orden en que se realizan cada uno de los pasos a seguir para la solución de determinado problema.
2. Definición. El resultado no debe cambiar sobre las mismas condiciones del problema, este siempre debe ser el mismo.
3. Finito. No debe ser repetitivo en procesos de manera innecesaria; deberá terminar en algún momento [3].

Teniendo en cuenta las características anteriores [3] define un algoritmo como "una serie de operaciones detalladas y no ambiguas para ejecutar paso a paso que conducen a la resolución de un problema, y se representan mediante una herramienta o técnica" [3].

Según Delgado [3] se debe tener presente que el algoritmo luego, en cuestión, será convertido en un programa de computadora, por lo cual se deben tener en cuenta las siguientes partes:

- Descripción de los datos que serán tratados
- Una serie de acciones que deben ser ejecutadas para la manipulación de los datos
- Los resultados a obtener en consecuencia de la manipulación de los datos.

Visión artificial: La visión artificial o también llamada visión por computadora es una rama de la inteligencia artificial comprendida por un conjunto de técnicas y tecnologías que permiten a una maquina obtener información desde imágenes digitales, resolver alguna tarea o comprender diferentes aspectos de la escena que están visionando [4].

En la actualidad la visión artificial tiene una multitud de aplicaciones, desde el campo de la medicina, como por ejemplo para recuento y búsqueda de células, el campo de la industria (contar botellas, comprobar defectos en una cadena de montaje y hasta los sistemas más complejos que especifican a los robots su orientación en un entorno [4].

Para el análisis de imágenes la visión artificial utiliza un conjunto de técnicas, estas técnicas son: Captación de imágenes, memorización de la información recibida y por último un procesado que permite realizar una interpretación de los resultados.

En [5] se mencionan Algunos de los métodos de captación de imágenes:

Píxel. En este método una imagen digital es propuesta como una cuadricula, donde elemento de esa cuadricula es llamada píxel.

Nivel de grises. Al digitalizar una imagen, es cuantificada la intensidad del brillo en la escena original que corresponde a cada punto, de esta manera se obtiene un numero llamado “nivel de gris”.

Imagen binaria. Una imagen binaria solo contiene dos niveles de gris: blanco y negro. Se convierte cada píxel en blanco o negro teniendo en cuanta el llamado nivel binario o UMBRAL, tal como se muestra en la figura 1.

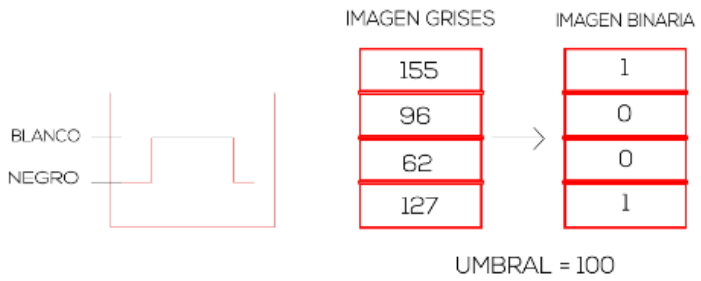


Ilustración 3 *Método de captación de imagen: imagen binaria (imagen propia).*

Captura de movimiento con sensores inerciales: Los sistemas inerciales utilizan unos pequeños sensores (normalmente acelerómetros y giroscopios) que recogen información sobre la aceleración y la velocidad angular del sensor. Conociendo la posición y la velocidad angular inicial e integrando las informaciones que recogen los sensores, es posible determinar la posición, eje de giro y velocidad angular de cualquier sensor. Los datos recogidos por los sensores inerciales (inertial guidance system) se transmiten a un ordenador, donde se puede observar sobre una figura animada el movimiento completo registrado. En los sistemas inerciales puros puede producirse el problema de la deriva de integración (integration drift: los errores numéricos se acumulan sobre la velocidad u orientación del sensor), por lo cual es común combinar esta técnica con otros métodos de captura.

Este tipo de sistemas de captura de movimiento no utiliza mecanismos externos como cámaras; y como en el caso de los sistemas ópticos, cuantos más sensores se utilicen, más real es el movimiento reproducido. Son fáciles de transportar y tienen grandes rangos de captura. Uno de los sistemas más conocidos de este tipo son los mandos inalámbricos de la Wii de Nintendo (Wiimote o Wii remote), si bien para captura de movimiento se emplean otros sensores mucho más precisos y con mayor frecuencia de captura, generalmente acoplados a unos trajes

especiales con varios sensores y en donde se ubica también una unidad transmisora. Existen trajes de prestaciones muy variadas cuyos precios básicos varían entre 25000 y 80000 dólares [6].

Fusión de datos:

La fusión de datos de diferentes fuentes comprende un conjunto de técnicas para integrar datos que se originan a través de diferentes sensores con el propósito de determinar aspectos del mundo exterior. El objetivo es obtener resultados más precisos, a partir de múltiples sensores, generalmente de diferente clase, realizando deducciones que no podrían ser conseguidas a partir de uno solo [7].

Una de las definiciones más citadas sobre fusión de sensores corresponde a la establecida por la Junta de Directores de Laboratorios del grupo de fusión de datos Joint Directors of Laboratories (JDL) quienes definen este proceso como “un proceso de múltiples niveles y fases de detección automática, que tratan con la asociación, correlación, estimación, y combinación de datos e información de una o múltiples fuentes para lograr posiciones de refinado, estimaciones de identidad, evaluaciones completas, situaciones de menazas oportunas y su significado” [1].

Niveles de fusión de datos: Según la junta de directores de laboratorios (JDL: Joint Directors of Laboratories) definen la fusión como un proceso integral de tratamiento de datos [8], según el autor [9] éste proceso está conformado por 6 niveles que empiezan desde la captura de datos hasta el resultado final. A continuación, serán presentados cada uno de estos niveles:

Nivel 0: Pre-procesado de las fuentes: Normalmente los datos que son pre-procesados filtrándolos o alineándolos temporalmente y se hace a través de un software integrado a cada uno de los sensores comerciales, de manera que cada uno de los sensores entrega sus datos de manera independiente. Este proceso es llamado pre-fusión.

Nivel 1: Evaluación del objeto: Se combinan los sensores para obtener diferentes características de la entidad como velocidad, posición, atributos, etc.

Los algoritmos de este nivel son:

1. Alineación de datos: Se realizan ajustes espacio-tiempo y de unidades para permitir su procesamiento.
2. Correlación dato/objeto: asociación y correlación de datos para cada individuo. Con el fin de permitir una integración correcta de los datos.
3. Estimación de posición, cinemática y demás atributos del objeto: A través de la combinación de modelos físicos y estadísticos se realiza la obtención del vector de estado.

Nivel 3: Evaluación del futuro: Esta evaluación es realizada mediante modelos de inteligencia artificial como los predictivos, razonamiento automático y la estimación estadística. Con los ya mencionados modelos se proyecta a futuro la entidad que fue analizada desde la situación actual.

Los algoritmos de este nivel son:

1. Estimación/agregación de capacidad de fuerza: buscando establecer interrelaciones y hacer un sistema robusto se realiza una agregación de la información de diferentes subsistemas.
2. Estimación de implicaciones: Son los resultados de una hipotética acción aplicada al sistema.

3. Evaluación de multiperspectiva: Análoga al del nivel 2, observación del sistema desde el exterior e interior.

Nivel 4: Proceso de refinamiento: Es una meta proceso que supervisa todo el proceso de DF con el fin de obtener mejor rendimiento del proceso en tiempo real, a través del uso de recursos como la modelización sensorial y la computación de medidas de perfeccionamiento.

Se puede decir que este nivel no está incluido del todo en el proceso de fusión, sino más bien de una manera parcial, puesto que este proceso de refinamiento es necesario para asegurar el funcionamiento de la fusión y las operaciones a realizar en el DF.

Los algoritmos de este nivel son:

1. Gestión de la misión: Para conseguir los resultados esperados se hace una dirección de los recursos existentes.
2. Para reconocer el proceso se definen las entidades específicas.
3. Requerimientos de las fuentes: para identificar las entidades se establece la infraestructura que necesitan las fuentes.
4. Modelización del rendimiento del sistema: Se define la estructura del sistema de fusión de datos, la relación que tienen las fuentes, etc.
5. Control del sistema: como el control de optimización y multiobjetivo.

Nivel 5: Refinamiento cognitivo: Tomando la relación sensor-procesamiento-persona se optimiza el sistema de fusión. Proyectando modelos en los que la representación de los resultados posibilita al operador encontrar errores en el procesamiento, y que este previamente pueda detectar los que error que suministren los sensores.

Robótica colaborativa: “Son procesos e intercambio de información de máquinas, sistemas IT y equipos humanos con el cual trabajen conjuntamente en red”.

Sensor inercial o Sensor IMU: “Es un componente capaz de obtener la posición, orientación y velocidad de cualquier dispositivo donde sea utilizado, es un sensor que mide aceleración y velocidad angular y se utiliza en aplicaciones de captura y análisis de movimiento”.

Internet of Things o Internet de las cosas: “Es un sistema de dispositivos de computación interrelacionados, máquinas mecánicas y digitales, objetos, animales o personas que tienen identificadores únicos y la capacidad de transferir datos a través de una red, sin requerir de interacciones humano a humano o humano a computadora”.

Motion capture o Captura de movimiento: “Es una técnica de grabación de movimiento, en general de actores u animales vivos, y el traslado de dicho movimiento a un modelo digital, realizado en imágenes de computadora”.

1.3.2. Marco legal

Para la ejecución del presente proyecto se considera la siguiente reglamentación:

Por ser un proyecto de ingeniería, se toma como marco de referencia el Código de ética de la ingeniería en Colombia mediante la **ley 842 de 2003**, la cual es la norma que legaliza la conducta profesional del Ingeniero en general y su violación será sancionada mediante el procedimiento establecido en la misma.

La **resolución 8430 de 1993** en el cual se establece las normas científicas, técnicas y administrativas para investigación en salud. **EL ARTICULO 11** nos dice que las investigaciones de este tipo se clasifican en 3 categorías: investigación sin riesgo, investigación con riesgo mínimo e investigaciones con riesgos mayor que

el mínimo, teniendo lugar la presente investigación en la primera categoría “investigaciones sin riesgo” puesto que se emplean técnicas y métodos de investigación documental y no se realiza ninguna intervención o modificación intencionada de las variables biológicas, fisiológicas o psicológicos en los individuos sometidos a las pruebas.

Aspectos éticos.

Se prevé el uso de un banco de imágenes, por lo cual los registros que serán tomados y de los cuales se hará uso posteriormente, serán obtenidos con el consentimiento previo de cada individuo de acuerdo con el **ARTICULO 15 Y 16** de la **resolución 8430 de 1993**. Este consentimiento podrá ser definido verbalmente y en caso de requerirse, por consentimiento escrito. Los datos tomados serán anonimizados según el criterio de las personas que sean sometidas a las pruebas con los sensores.

El **código ético del IEEE** donde se establecen diez compromisos para guiarnos de la manera más ética y profesional en el desarrollo del proyecto de investigación:

1. Aceptar la responsabilidad de tomar decisiones consistentes con la seguridad, salud y bienestar de las personas, y exponer prontamente los factores que pueden dañar a la gente o al entorno;
2. Evitar, siempre que sea posible, conflictos de interés reales o percibidos, y exponerlos a las partes afectadas cuando aquellos existan;
3. Ser honestos y realistas en las afirmaciones y estimaciones basadas en los datos disponibles;
4. Rechazar sobornos en todas sus formas;
5. Mejorar la comprensión de la tecnología, su aplicación adecuada y sus posibles consecuencias;

6. Mantener y mejorar nuestra competencia técnica, y aceptar tareas para otros sólo si estamos cualificados por adiestramiento o experiencia, o después de exponer completamente las limitaciones pertinentes;
7. Buscar, aceptar y ofrecer críticas honestas sobre el trabajo técnico, aceptar y corregir errores y reconocer adecuadamente las contribuciones de otros;
8. Tratar equitativamente a todas las personas independientemente de su raza, religión, sexo, capacidades, edad o nación;
9. Evitar dañar a otros, sus propiedades, reputación o puesto de trabajo mediante acción falsa o maliciosa;
10. Ayudar a los/las compañeros/as en su desarrollo profesional y apoyarles en el seguimiento de este código de ética.

1.4. METODOLOGÍA

1.4.1. Línea de investigación

El presente proyecto de ingeniería pertenece a la línea de investigación de la inteligencia artificial y la visión por computadora.

1.4.2. Tipo de investigación

Según los objetivos planteados ésta es una investigación de tipo aplicada, ya que se centra en encontrar una forma o estrategia para desarrollar dos sistemas captura de movimiento que puedan ser fusionados para obtener mejores resultados del movimiento del brazo. El enfoque metodológico es de tipo cuantitativo puesto que se manipulan datos numéricos como las posiciones de las articulaciones del miembro superior y determinación de la precisión que tendrá la solución desarrollada.

1.4.3. Definición de la metodología

Técnicas para la recolección de información.

Los datos de captura de movimiento a través de video serán proporcionados por los autores del proyecto "**PROTOTIPO DE CAPTURA DE MOVIMIENTO DEL CUERPO HUMANO EXTREMIDAD SUPERIOR UTILIZANDO OPEN CV**", el cual es otra parte del proyecto genérico que se encuentra en desarrollo por el docente Luis Murillo en la Universidad del Sinú. Donde se realiza la captura de movimiento a través de video utilizando una cámara web y la librería de software libre OpenCV La otra parte de los datos que se requieren para la fusión serán recolectados con sensores inerciales MPU-6050 conectados a un Arduino Uno.

Para desarrollar los sistemas de captura de movimiento (y prepáralos para el algoritmo de fusión de datos) se eligió la metodología utilizada por los autores del sistema de captura de movimiento mediante video () quienes se basaron en el modelo de Martin y McClure y del SEI [1], que consiste en tres grandes fases: 1. investigación y desarrollo inicial, 2. Investigación aplicada, y 3. Transferencia. Cada una de ellas involucrando el desarrollo de varios proyectos de investigación, posiblemente cada uno de ellos con métodos y técnicas diferentes.

Ilustración 4 *Esquema de la estrategia de investigación (Tomado de: http://web.imt-atlantique.fr/x-info/harbol07/MIFISIS2004.pdf)*

A. Investigación y desarrollo inicial

Es necesario conocer a fondo los métodos, funciones y técnicas con los que dispone OpenCV para la captura de movimiento, con el fin de comprender el funcionamiento de este sistema y establecer las mejoras a realizar. Por otra parte, se necesitaba identificar los requerimientos necesarios que permitiera la captura de las tres articulaciones de la extremidad superior utilizando sensores inerciales.

B. Investigación aplicada

Una vez interiorizados los conceptos fundamentales de la visión por computador y el funcionamiento de los sensores inerciales IMU-6050 se realizarán pruebas con

diferentes técnicas; dentro de los cuales se destacan el cálculo del ángulo inclinación con el acelerómetro del sensor y cálculo de ángulo de rotación con el giroscopio. Para el sistema de captura de video se habrán identificado las acciones de mejora a realizar.

Posteriormente se procederá al desarrollo de los sistemas finales, los cuales son capaces de marcar la extremidad superior con las posiciones y ángulos generados. Para ello es necesario un entorno controlado utilizando un chroma key, que no es más que un fondo de color verde y una buena iluminación.

C. Transferencia

Se realiza la sustentación del proyecto en la Universidad del Sinú seccional Cartagena por parte de los autores principales, con la finalidad de ser evaluado y aprobado como proyecto de grado para la obtención del título de ingeniero de sistemas.

Por otra parte, para mantener la consistencia de los datos obtenidas por los sistemas de captura de movimiento que van a ser emitidos al proceso de fusión, se tomará como referencia el diseño conceptual de fusión de datos de fuentes homogéneas planteado por (M.M. VAZQUEZ).[23]. Ver Ilustración 5.

Se realiza la sustentación del proyecto en la Universidad del Sinú seccional Cartagena por parte de los autores principales, con la finalidad de ser evaluado y aprobado como proyecto de grado para la obtención del título de ingeniero de sistemas.

Por otra parte, para mantener la consistencia de los datos obtenidas por los sistemas de captura de movimiento que van a ser emitidos al proceso de fusión, se tomará como referencia el diseño conceptual de fusión de datos de fuentes

homogéneas planteado por (M.M. VAZQUEZ).[23]. Ver Ilustración 5.

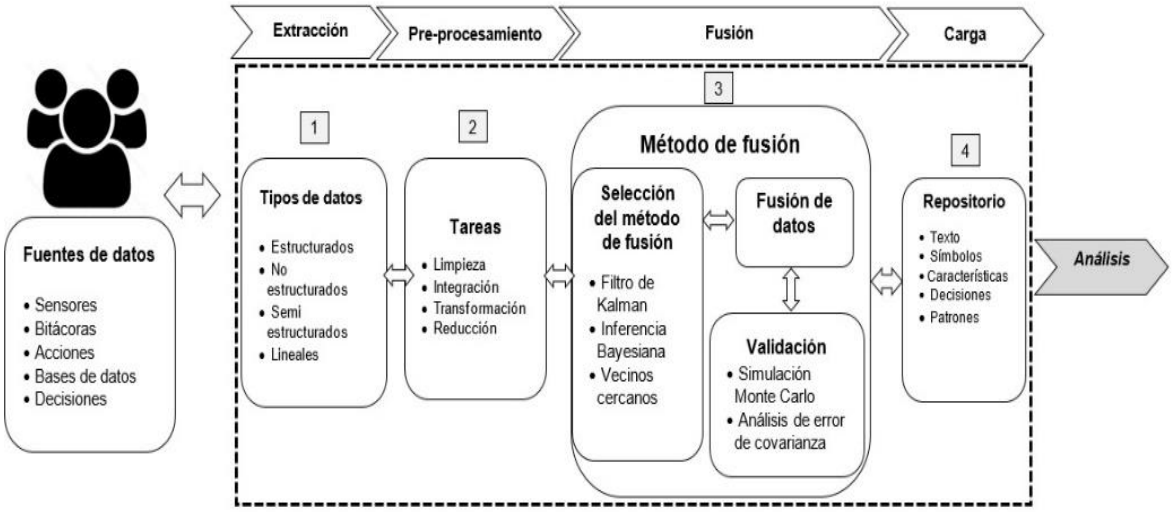


Ilustración 5 *Metodología para el desarrollo del algoritmo de fusión de datos heterogéneos propuesto por Julio Muñoz – Guillermo Molero*

Es importante tener en cuenta la anterior metodología puesto que los datos que generen los sistemas de captura de movimiento deben llegar en un formato adecuado al algoritmo fusión, por ello se tendrá en cuenta la fase de extracción y preprocesamiento propuesta en la Ilustración 5.

Finalmente, para determinar la precisión de los sistemas de captura de movimiento, los datos serán contrastados con instrumentos de medida angular como el gluniometro e inclinómetro.

2. DESARROLLO

En este capítulo se detalla el desarrollo de los sistemas de captura de movimiento con todas los métodos y técnicas seleccionadas teniendo en cuenta la metodología planteada en el capítulo anterior, iniciando por la construcción de los sistemas de captura, seguido del proceso de extracción de los datos, continuando con la etapa de preprocesamiento, y finalmente validando el algoritmo de fusión desarrollado.

2.1. Análisis de los sistemas de captura de movimiento.

Extracción de los datos

Para dar inicio con el desarrollo del presente proyecto fue necesario ejecutar y examinar los datos entregados por el sistema de captura de movimiento mediante video e iniciar la construcción del sistema de captura mediante sensores inerciales, con el fin de comprender su funcionamiento.

Sistema de captura de movimiento mediante video

Para la ejecución de este algoritmo fue necesario instalar las libreras de openCV en su versión 3.4.7 Y Numpy bajo el lenguaje de programación Python 3, y para facilitar la manipulación del código se usó “Spyder”, que es un entorno de desarrollo integrado multiplataforma de código abierto para la programación científica; marcadores de colores para las articulaciones del brazo; y una tela de color verde para hacer el contraste de color y facilitar la detección del brazo.

Se estudió a fondo la librería OpenCV y las diferentes técnicas que se utilizan para el reconocimiento de movimiento y objetos en el entorno. Se inició con el montaje del sistema de captura de video, siguiendo los pasos indicados en el manual técnico del algoritmo realizado en el trabajo anterior. Dado que OpenCV es muy sensible a los colores e iluminación del entorno, se utilizó Chroma key o mal

llamado fondo verde, que es una técnica comúnmente usada en la industria cinematográfica y los videojuegos para extraer el color de una imagen o video y reemplazarla en el área que ocupaba ese color por otra imagen o video. Se usa el color verde en particular, porque que no tiene incidencia en la piel, es decir, que es muy poco probable que la cámara relacione este color con alguna parte de nuestro cuerpo, controlando el fondo se realiza una mejor detección del objeto de estudio en la imagen. Este sistema utiliza funciones de reconocimiento de objetos por colores para lograr detectarlos en el espacio, en este caso se realizó una función para detectar el color azul, que actúa como la articulación del codo y la detección del color amarillo que a su vez actúa como la articulación del hombro.

Existen varios espacios de color, que son una manera de representar los canales de color presentes en una imagen, estos dan a la misma un matiz en particular. Algunos de los espacios de color más populares son: RGB (rojo, verde, azul), CMYK (cian, magenta, amarillo, negro), HSV (tono, saturación, valor), etc.

El espacio de color RGB es el predeterminado de OpenCV, pero hay algo muy particular con OpenCV, y es que almacena el color en formato BGR, intercambiando el canal azul (B) por el (Rojo). Las diferentes intensidades de color azul, verde y rojo dan diferentes tonos de color. RGB describe un color como una tupla de componentes como se puede apreciar en la ilustración 6, donde cada componente puede tomar un valor entre 0 y 255.

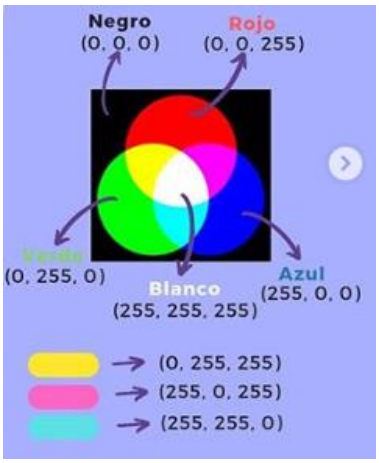


Ilustración 6 *Representación de algunos colores en formato BGR (tomado de: https://www.instagram.com/p/B2o1kw_BsV_/)*

Para lograr la detección de color en OpenCV es necesario hacer una conversión de espacio de colores, como el HSV, este formato es una matriz de tres dimensiones, con cada matriz guardando saturación, intensidad y color. Hay muchos métodos disponibles en OpenCV para convertir imágenes de un espacio de color a otro. En este caso se utilizó la función *cv.cvtColor(input_image, flat)* donde *flag* determina el tipo de conversión, se puede usar esto para extraer un objeto coloreado. En HSV es más fácil representar un color que en el espacio de color BGR. Esta aplicación extrae un objeto de color azul y uno amarillo, aplicando los métodos presentes en la ilustración 7.

Ilustración 7 *Metodología para extracción de un objeto por color en una imagen. (imagen propia)*

El siguiente segmento de código realiza la captura de video en vivo, luego hace la conversión de un espacio de color a otro y finalmente logra la detección del color azul en la imagen como se puede apreciar en la ilustración 8.

```
while (True):
    _, frame = cap.read()

hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

mascara_azul = cv2.inRange(hsv, azules_bajos, azules_altos)
ret,contoursAz,_ = cv2.findContours(mascara_azul,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)
```

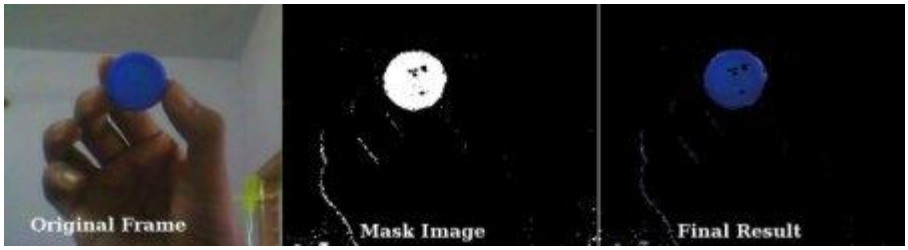


Ilustración 8 *Detección del color azul mediante openCV.(tomado de: https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_imgproc/py_colorspaces/py_colorspaces.html)*

La detección del rostro y el puño como punto de referencia para las demás articulaciones se logró con el método de *Haar cascade* para detección de objetos y rostros. Este es un método que se basa en la detección de objetos mediante clasificadores en cascada basados en funciones de *Haar*. Este método usa aprendizaje automático en función cascada y se entrena a partir de muchas imágenes positivas (imágenes de caras) y negativas (imágenes sin caras). Para hallar el punto o articulación principal se usa el método *Lucas Kanade* que sigue el flujo de movimiento de un objeto marcado, se inicia el ciclo de captura de *frame* por *frame* en un ciclo infinito para la detección de las articulaciones del brazo , y como ya se mencionó antes la articulación del hombro es marcada con un círculo de color amarillo y la del codo con uno de color azul, se realiza la detección de esos colores respectivamente y se traza una línea para simular el esqueleto.

En resumen este algoritmo empieza la realización de captura del video en tiempo real, procesando los fotogramas y convirtiendo la imagen de RGB a HSV, se

establece el rango de colores en el cual va a detectar la cámara, es decir, se dan los valores correspondientes al rango de colores que se desea calcular para detectar el objeto, se realiza el contorno al objeto de color establecido que aparezca en la imagen, se crea un condicional para ubicar el punto central del objeto que se está detectando y seguir su movimiento. se hace entrega de los datos que se están calculando, en este caso los pixeles en los que se está moviendo el objeto, arrojando por pantalla el video en tiempo real, la marcación de puntos en el objeto y el seguimiento de este [10].

Finalmente, lo datos que son entregados por este sistema representan las coordenadas de cada articulación detectada, basadas en pixeles y los ángulos que se forman entre los puntos codo-hombro y muñeca-hombro como se puede observar en la Ilustración 9. Estas posiciones entregadas en base a pixeles no pueden ser utilizadas para la fusión con los datos entregados por el sistema de sensores inerciales, debido a que no representan una métrica real. El primer reto de este proyecto fue hallar la relación existente entre pixeles/métrica real, donde la métrica elegida fueron los milímetros. Teniendo un objeto de referencia de tamaño y ubicación conocidos en la imagen.

En definitiva, se identificó que era necesario convertir los píxeles a la métrica del objeto de referencia para determinar la distancia de cualquier otro objeto, en este paso los puntos de las articulaciones desde el objeto de referencia y asimismo realizar una calibración de cámara para corregir la distorsión que pudiera ser generada por el lente de la cámara.

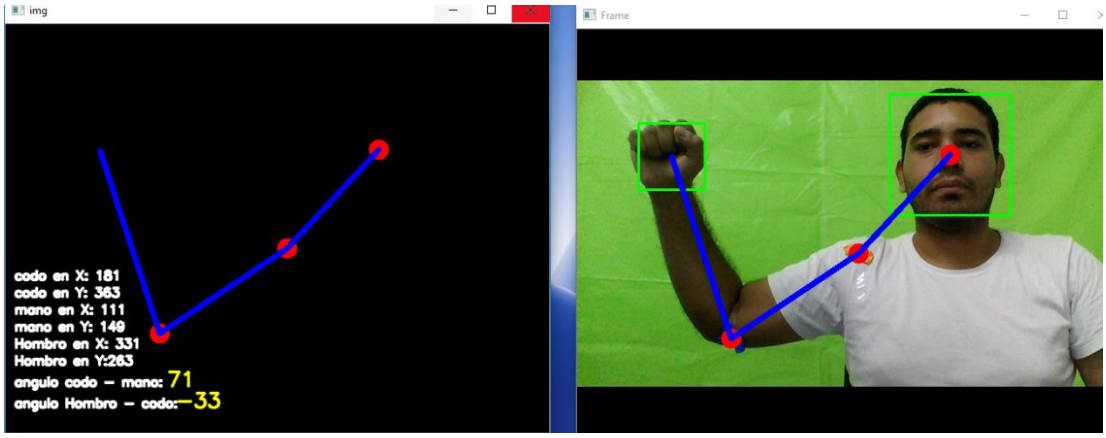


Ilustración 9 *Datos entregados por el sistema de captura de movimiento mediante video. (imagen propia).*

Análisis de los datos recibidos a través del sistema de captura de movimiento mediante sensores inerciales

Para este sistema se estableció la capturara de movimiento mediante sensores inerciales a través de 3 sensores de bajo costo (MPU-6050), conectados a una placa Arduino UNO (ver Ilustración 10). Estos sensores toman los ángulos del movimiento del movimiento del brazo, y generan un paquete de datos a través del puerto serial, estos datos se dividen en tres partes: aceleración, ángulo y giro.

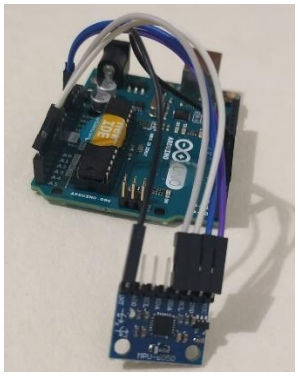


Ilustración 10 *Sensor inercial MPU-6050 conectado a placa Arduino Uno utilizado para la captura de movimiento.*

El sensor MPU-650 es un IMU (inertial Measurment Units) o en español unidad de medición inercial de 6 grados de libertad, éste cuenta con un acelerómetro y un giroscopio ambos de 3 ejes. Para tener claro el funcionamiento de estos sensores fue necesario tener claros dos conceptos fundamentales; como es medida la aceleración por el acelerómetro y como es medida la velocidad angular por el giroscopio.

Se sabe que la aceleración es la variación de la velocidad en un determinado tiempo y se obtiene con la ecuación $a = \frac{dV}{t}$, donde dV es la velocidad y t es el tiempo. De la misma manera la segunda ley de newton indica que en un cuerpo con masa constante, su aceleración será proporcional a la fuerza que actúa sobre él mismo $a = \frac{F}{m}$, donde F es la fuerza y m es la masa del cuerpo.

Los acelerómetros utilizan el mencionado concepto de Newton para medir la aceleración, en su interior tienen un sistema micro electromecánico que de una manera similar a un sistema masa-resorte permite medir la aceleración, pero hay que tener en cuenta que, aunque no haya movimiento, el acelerómetro siempre estará recogiendo la aceleración de la gravedad. Por otra parte, la velocidad es la tasa de cambio de desplazamiento angular en unidades de tiempo, en otras palabras, que tan rápido gira un cuerpo alrededor de su propio eje, en la ilustración 11 se puede apreciar este principio y la fórmula para calcularlo. Estos giroscopios utilizan al igual que el acelerómetro, un microsistema eléctrico mecánico para medir la velocidad angular empleando el efecto Coriolis, el cual hace que un objeto que se mueve sobre el radio de un disco en rotación tienda a acelerarse con relación a ese disco según si el movimiento es hacia el eje de giro o alejándose de éste.

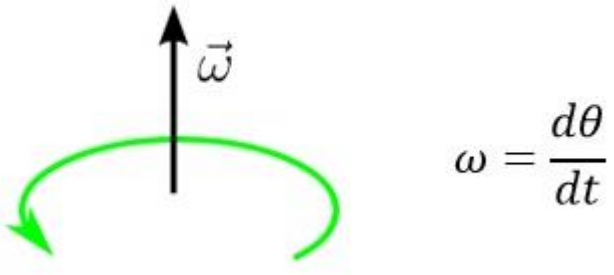


Ilustración 11 *Velocidad angular.*

Con el sensor utilizado se pudo medir la velocidad angular y los componentes X, Y, Z de la aceleración, la dirección de los ejes se indica en el sensor (ver Ilustración 12), lo cual es muy importante tener en cuenta para no cometer errores en el signo de las aceleraciones.

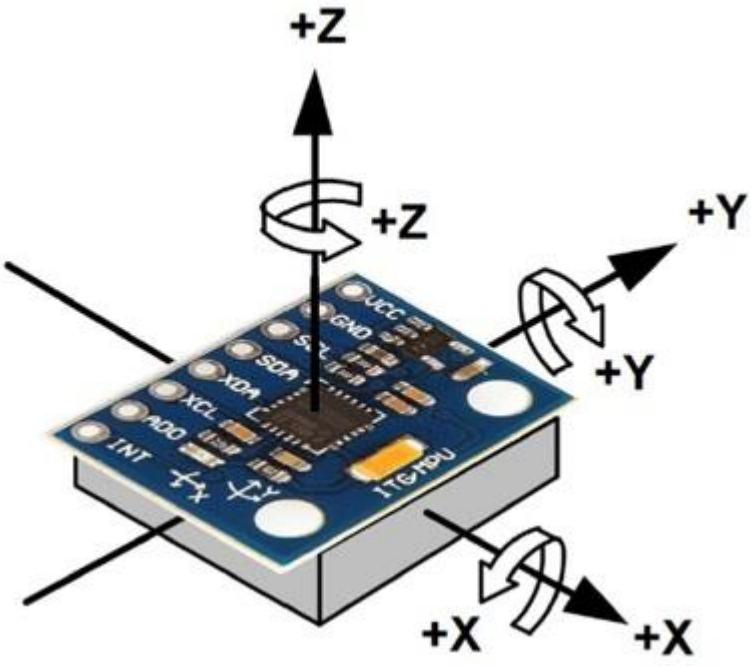


Ilustración 12 *Dirección de los ejes indicada en el sensor inercial.*

Una vez comprendido el funcionamiento del sensor inercial MPU-6050 se procedió a realizar las conexiones entre éste y la placa Arduino UNO, el esquema de conexiones utilizado y la tabla con la especificación estas conexiones se pueden apreciar en la ilustración 13 y tabla 2 respectivamente.

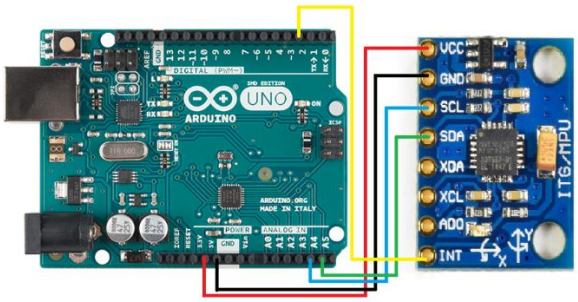


Ilustración 13 *Esquema de conexiones utilizado para el sistema de captura de movimiento mediante sensores inerciales.*

Tabla 2 *Conexiones entre sensor MPU-6050 y Arduino Uno.*

MPU-6050	Arduino Uno
VCC	5V
GND	GND
SCL	A5
SDA	A4
INT	D2

Después de realizar las conexiones se desarrolló el código a cargar en Arduino para tomar los ángulos generados con el movimiento en cada uno de los ejes utilizando la librería desarrollada por Jeff Rowberg disponible en su cuenta de Github para el uso del sensor MPU-6050 con Arduino, a su vez esta

biblioteca trabaja con una biblioteca adicional para la comunicación I2C. El protocolo I2C significa Circuito Interintegrado y es un protocolo de comunicación serial que toma e integra lo mejor de los protocolos SPI y UART. Es un bus muy usado en la industria, principalmente para comunicar microcontroladores y sus periféricos en sistemas integrados (Embedded Systems) y generalizando más, para comunicar circuitos integrados entre sí, que normalmente residen en un mismo circuito impreso [11]. Estas librerías fueron instaladas en el entorno de desarrollo integrado de Arduino.

Se encontraron dos formas de hallar los ángulos generados por el desplazamiento en cada uno de los ejes, la primera forma utiliza el acelerómetro del MPU-6050 teniendo en cuenta que la gravedad es la única fuerza que actúa sobre el mismo. En ese orden de ideas, los valores que se obtienen de los componentes del acelerómetro corresponden a la gravedad y por consecuencia los ángulos generados serán la inclinación del plano del sensor, puesto que la gravedad siempre es vertical [12]. Para calcular los ángulos de inclinación en un espacio 3D tanto en X como Y se utilizan las siguientes formulas:

$$\theta_x = tan^{-1}\left(\frac{a_x}{\sqrt{a_y^2 + a_z^2}}\right)$$

Ecuación 1 *Ángulo de inclinación en espacio 3D en eje X*

$$\theta_y = tan^{-1}\left(\frac{a_y}{\sqrt{a_x^2 + a_z^2}}\right)$$

Ecuación 2 *Ángulo de inclinación en espacio 3D en eje Y*

Calcular los ángulos de esta manera funciona adecuadamente solo si la única fuerza que actúa sobre el sensor es la de la aceleración, si movemos rápidamente el sensor y sin realizar ninguna inclinación, el ángulo obtenido varía generando errores. Otra forma de calcular los ángulos es usando el giroscopio del MPU-6050. El giroscopio entrega la velocidad angular, y para calcular el ángulo se necesita integrar la velocidad y conocer el ángulo inicial del desplazamiento [12]. Esto se puede hacer utilizando la siguiente fórmula:

$$\theta_x = \theta_{x_0} + \omega_x \Delta\tau$$

Ecuación 3 Ángulo de desplazamiento Eje X

$$\theta_y = \theta_{y_0} + \omega_y \Delta\tau$$

Ecuación 4 Ángulo de desplazamiento Eje Y

θx se refiere al ángulo que gira el eje *x* sobre su propio eje como se muestra en la ilustración 15, donde se observa que la velocidad angular es perpendicular al plano de rotación.

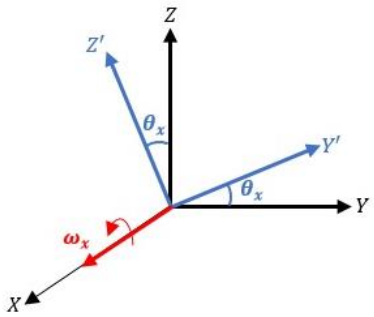


Ilustración 14 *Velocidad angular vs plano de rotación [12].*

Calcular los ángulos de esta manera no proporciona resultados tan precisos, incluso cuando el sensor no se mueve, el ángulo varía, o si se gira el sensor a

cierto ángulo y luego si se regresa a la posición inicial el ángulo no es el mismo debido a que al integrar la velocidad angular y sumar el ángulo inicial existe un error llamado DRIFT, resultado de una mala medición del tiempo o del ruido en la lectura del sensor. Este error en cada interacción se acumula y se hace mayor. Para reducir los efectos del DRIFT se han desarrollado diferentes métodos, en su mayoría son filtros que eliminan el ruido generado de las lecturas del sensor o se pueden usar magnetómetros y acelerómetros, estos sensores pueden calcular los ángulos y mejorar los datos presentados por el giroscopio.

Teniendo en cuenta que este debe ser un proyecto desarrollado a costo mínimo se optó por aplicar un filtro al cálculo de los ángulos llamado filtro complementario, este filtro es muy utilizado ya que no se necesita mucha capacidad de procesamiento como es el caso del filtro de Kalman que, aunque es capaz de entregar mejores resultados su costo de procesamiento computacional lo hace difícil de implementar en Arduino.

El filtro complementario es una combinación del ángulo calculado por acelerómetro y el ángulo calculado por el giroscopio. La ecuación para calcular ángulos utilizando el filtro complementario se puede observar en la ilustración 16, dónde el ángulo del acelerómetro está pasando por un filtro pasa bajos, atenuando variaciones bruscas en la aceleración; y el ángulo calculado por el giroscopio tiene un filtro pasa altos que posee un gran dominio cuando existen rotaciones muy rápidas. Se puede utilizar también otros valores diferentes a 0.98 y 0.02 pero siempre deben sumar 1 [12].

$$ángulo = 0.98\left(ángulo + \boldsymbol{\omega_{giroscopio}}\, dt\right) + 0.02(\boldsymbol{ang_{acelerómetro}})$$

Ilustración 15 *Ecuación para calcular el ángulo usando el filtro complementario*

El código utilizado en ejecución muestra los ángulos generados con el movimiento del sensor a través del monitor Serial del IDE de Arduino (ver Ilustración 17).

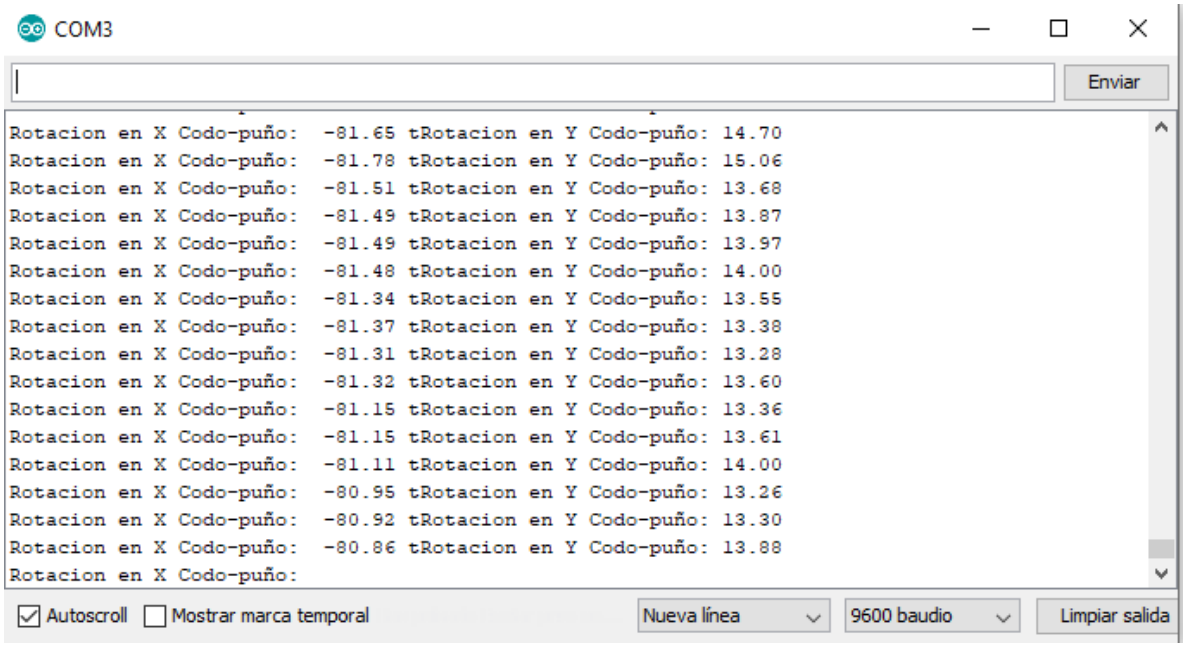


Ilustración 16 *Ángulos generados por el sensor MPU-6050*

2.2. Módulo de Preprocesamiento.

Módulo de objeto de calibración para el sistema de video.

Como se mencionó en el análisis de los datos entregados por el sistema de captura de movimiento mediante video, es necesario encontrar la relación pixeles por métrica para determinar los ángulos basados en una métrica real. Primero se realizó una calibración utilizando un recorte de material azul como un objeto de referencia.

Este objeto de referencia debe contar con dos características muy importantes:

1. Las dimensiones del objeto deben ser conocidas en términos de ancho o alto y en una unidad de medida real como milímetros, pulgadas, etc.
2. El objeto debe ser fácilmente identificable en la escena, ya sea en una posición específica de la imagen, basado en el color; o por su forma.

Para el desarrollo de este módulo de objeto de calibración se utilizaron las bibliotecas Numpy y pickle. Se declaró una variable con el ancho del objeto calibrador (70 mm) y un arreglo de numpy para guardar la relación métrica por pixeles, el objeto a identificar fue un recorte de un material de color azul con una medida de 70 milímetros de ancho, en la **Ilustración 18** se muestra el rango de azules establecido para la identificación del objeto. Este objeto se coloca a una distancia predeterminada de la cámara para identificar cual es la relación métrica por pixeles a una distancia en concreto, si el objeto se aleja la cantidad de pixeles que ocupe será menor y si se acerca será mayor, por lo cual la relación será diferente. Teniendo en cuenta lo anterior, para dar un valor más preciso se guardan veinte registros de la relación en un archivo para luego promediar estos valores.

```
azulBajo = np.array([100,200,25], np.uint8)
azulAlto = np.array([125, 255, 255], np.uint8)
```

Ilustración 17 *Rango de colores establecido para la detección del objeto calibrador.*

Para iniciar la calibración un sujeto sostiene el objeto calibrador a una distancia establecida, en la misma posición a la que se va a realizar la captura del movimiento del brazo, el objeto se debe mantener estático en lo posible, para que el resultado sea más preciso. Una vez se identifica el objeto se dibuja el contorno de este y se obtiene el área del objeto en pixeles (ver Ilustración 20).

```
x,y,w,h = cv2.boundingRect(c)
aspect_ratio = float(w)/h
equi_diametro = np.sqrt(4*area/np.pi)
```

Ilustración 18 *Función utilizada para obtener el área del objeto basado en pixeles*

Una vez obtenida el área del objeto en pixeles se pudo calcular la relación pixeles por métrica dividiendo el ancho del objeto previamente definido en una variable entre el área en pixeles, esta relación se muestra en tiempo real en el fotograma que ejecuta el módulo, como se puede apreciar en la ilustración 20.

Ilustración 19 *Ejecución del módulo de calibración.*

Los registros almacenados como resultado de la calibración luego son leídos por el módulo de captura de movimiento del brazo y se hace el cálculo de los ángulos en base a la relación, es decir que ahora la distancia entre cada articulación puede ser dada en milímetros. Esta función se puede apreciar en la Ilustración 21.

Primero se calcula la distancia entre las articulaciones restando las posiciones de cada articulación en cada eje, el ángulo generado se calcula utilizando la función

atan () de la librería *Math,* esta función recibe la distancia codo-hombro en *y* entre la distancia codo-hombro en *x.*

```
#angulo codo hombro
if azx != 0 and amx != 0 and azy != 0 and amy != 0:
    if azx != amx and azy != amy:
        distCHY=amy-azy #distancia codo hombro en Y
        distCHX=amx-azx #distancia codo hombro en X

        ndistCHY=distCHY*pixelespormetrica
        ndistCHX=distCHX*pixelespormetrica

        nanguloCH=math.atan(ndistCHY/ndistCHX)

        ######################################
        cv2.putText(img,str(int(math.degrees(nanguloCH))),(200,450), font, 0.8,(0,255,255),2,cv2.LIN
        print("angulo codo-hombro:", nanguloCH)
```

Ilustración 20 *Código para el cálculo de ángulos generados entre las articulaciones calculado en base a milímetros.*

Los resultados obtenidos con el objeto calibrador no fueron los suficientemente buenos, puesto que si la cámara al momento de capturar video no se encuentra en un ángulo perfecto de 90 grados “mirando hacia abajo” (como una vista de pájaro) a los objetos o puntos las dimensiones de los objetos en la imagen pueden aparecer distorsionadas. Por otra parte, la cámara web utilizada no había sido calibrada usando los parámetros intrínsecos y extrínsecos. Sin determinar estos parámetros la captura de imagen puede ser propensa a la distorsión radial y tangencial del lente.

Debido a lo anterior se decidió realizar un proceso de calibración adicional encontrando estos parámetros para “distorsionar” la imagen capturada y obtener una mejor aproximación de las medidas y distancias entre los marcadores de las articulaciones.

Calibración de la cámara.

El propósito principal de todo el sistema que se espera desarrollar es utilizar herramientas de bajo costo para obtener datos de captura de movimiento con la mayor precisión posible, por ello, para el sistema de captura de movimiento mediante vídeo se pretende utilizar cualquier tipo de cámara. Para este proyecto se decidió utilizar la cámara integrada del computador portátil HP 14. Entre las dificultades encontradas se detectó que las cámaras integradas a los equipos portátiles por lo general presentan mucha distorsión en las imágenes que capturan, una de ellas es la distorsión radial y la otra es la distorsión tangencial.

A causa de la distorsión radial las líneas rectas aparecerán curvadas y el efecto se hace mayor a medida que un objeto es alejado del centro de la imagen. En la ilustración 22 se puede observar que el borde no es una línea recta y no coincide, las líneas no aparecen rectas sino curvadas [13].

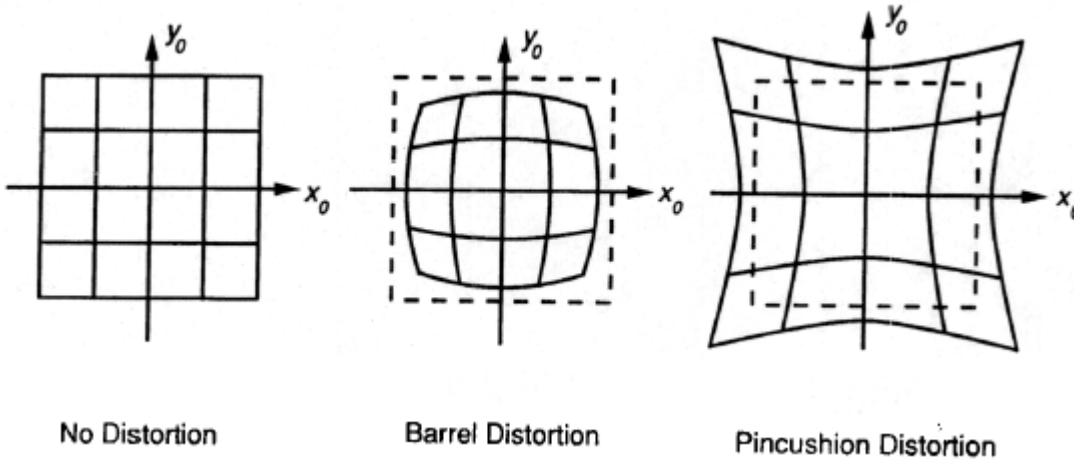


Ilustración 21 *Distorsión Radial presente en cámaras de bajo costo*

La ecuación para resolver la distorsión radial es la siguiente:

$$x_{corrected} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

$$y_{corrected} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

Por otra parte, tenemos la distorsión tangencial que ocurre debido a que la toma de imágenes no está perfectamente alineada en paralelo al plano de la imagen. Por lo cual, algunas áreas en la imagen pueden verse más cerca de lo esperado y los objetos a identificar pueden parecer más grandes. Esta distorsión tangencial puede ser corregida con la siguiente ecuación:

$$x_{corrected} = x + [2p_1 xy + p_2(r^2 + 2x^2)]$$

$$y_{corrected} = y + [p_1(r^2 + 2y^2) + 2p_2 xy]$$

Se necesita encontrar 5 parámetros que son conocidos como coeficientes de distorsión dados por:

$$Distortion\ coefficients = (k_1 \quad k_2 \quad p_1 \quad p_2 \quad k_3)$$

Y además también se necesita encontrar los parámetros intrínsecos y extrínsecos de la cámara. Los llamados parámetros intrínsecos o también llamados matriz de cámara son propios de una cámara. Estos contienen información como la distancia focal (fx, fy), centros ópticos (cx, cy), etc. Dependen solo de la cámara, por lo que una vez se realiza la calibración estos datos pueden ser almacenados para ser utilizados después. En la ilustración 24 se puede apreciar estos parámetros expresados como una matriz de 3x3. Donde *fx, fy* son el ancho y alto de la longitud focal y cx, cy son las coordenadas del punto principal (generalmente el centro de la imagen).

$$camera\ matrix = \begin{matrix} fx & 0 & Cx \\ 0 & fy & Cy \\ 0 & 0 & 1 \end{matrix}$$

Ecuación 5 Parámetros intrínsecos de una cámara expresados en una matriz de 3x3

Los parámetros extrínsecos son los vectores de rotación y translación que convierten las coordenadas de un punto 3D a un sistema de coordenadas. En este caso se está trabajando en un solo plano, por este motivo no es necesario tener en cuenta estos parámetros como sería necesario en las aplicaciones estéreo, es decir aplicaciones que cuentan con dos cámaras para la captura de vídeo.

OpenCV cuenta con funciones de calibración de cámara, una de ellas cv2.findChessBoardCorners(). Está función se basa en la técnica Zhang utilizando ecuaciones geométricas básicas para el reconocimiento de patrones. Las ecuaciones utilizadas dependen de los objetos de calibración elegidos. Actualmente OpenCV admite tres tipos de objetos para la calibración [14]:

- Tablero de ajedrez clásico blanco y negro
- Patrón de círculo simétrico
- Patrón de círculo asimétrico

Se debían tener fotos del patrón a utilizar capturadas con la cámara a calibrar y hacer que openCV a través de la función mencionada anteriormente encontrara los patrones. Se decidió utilizar el tablero de ajedrez, este tablero debía ser asimétrico, es decir, que la cantidad de cuadros en horizontal debe ser diferente a la cantidad de cuadros en sentido vertical. Se imprimió el tablero en una hoja adhesiva y luego se fijó en un pliego de cartón como se puede apreciar en la

ilustración 23, para darle estabilidad a la hoja, puesto que para obtener unos buenos resultados las imágenes que se capturen deben ser planas.

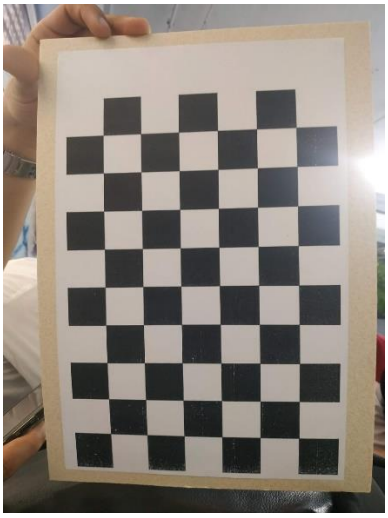


Ilustración 22 *Patrón elegido para calibración de cámara*

Para capturar las imágenes de la calibración se desarrolló un módulo que mediante OpenCV permite acceder a la cámara mostrando vídeo en vivo, y presionando una tecla guarda imágenes en formato JGP con un identificador único dentro de la carpeta del proyecto. El código utilizado se muestra en la ilustración 24.

```python
import cv2
import uuid


cap = cv2.VideoCapture(0)
font=cv2.FONT_HERSHEY_SIMPLEX#tipo de fuente para escribir sobre la imagen


while(cap.isOpened()):
    _,frame = cap.read()
    cv2.putText(frame,"Salir: E ",(5,30), font, 0.5,(0,255,255),2,cv2.LINE_AA)
    cv2.putText(frame,"Captura: T ",(5,50), font, 0.5,(0,255,255),2,cv2.LINE_AA)
    cv2.imshow('ventana', frame)
    key = cv2.waitKey(1) & 0xFF
    if key == ord('t'):
        nombre_foto = str(uuid.uuid4()) + ".png"
        cv2.imwrite(nombre_foto, frame)
        print("foto guardada exitosamente con el nombre {}" .format(nombre_foto))
    if key == ord('e'):
        break

cap.release()
cv2.destroyAllWindows()

```

Ilustración 23 *Código para la captura de imágenes*

El código anterior fue utilizado para capturar un mínimo de once imágenes que serán suficientes para obtener una buena calibración, las imágenes que se capturen del objeto de calibración de cámara deben ser en diferentes ángulos y que sean lo más diferente posible, tal como se muestra en la ilustración 25.

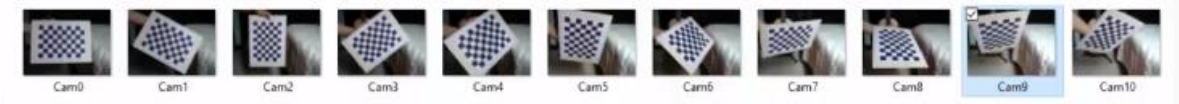


Ilustración 24 *Posiciones de ejemplo en las que se deben capturar las imágenes del patrón para la calibración de cámara.*

Calibración de los sensores inerciales MPU-6090.

El sistema de sensores inerciales también fue sometido a una etapa de calibración en los MPU6050, puesto que, es posible que el sensor no se encuentre completamente en una posición horizontal porque el sensor pudo ser soldado en el módulo con un ligero desnivel, generando un error en cada lectura. Por otra parte, existía la posibilidad que cuando se colocara el módulo en el brazo podía estar desnivelado, aunque a simple vista se notara que estaba correctamente posicionado y sin desnivel. Para resolver este inconveniente se configuran OFFSETS en el sensor y de esta manera se subsanan los errores que se puedan generar en la lectura.

El código utilizado para la calibración básicamente se encarga de modificar constantemente los OFFSETS como un intento para eliminar el error con la medida deseada. ´Al iniciar el código se leen los offsets actuales, se ubica el sensor en una posición horizontal y se debe mantener totalmente inmóvil durante la calibración. Para iniciar la calibración se ingresa un carácter por el puerto serie del IDE Arduino y el programa empieza a realizar las lecturas del acelerómetro y del giroscopio; con el uso de un filtro se estabilizan las lecturas y cada 100 valores se hace una comprobación para determinar si los valores son cercanos a los valores que se desean leer. Obedeciendo a esto se realiza una variación en los offsets disminuyendo o aumentándolos, esto hará que las lecturas que son filtradas coincidan con las mostradas a continuación:

Velocidad angular: p_gx=0, p_gy=0, p_gz=0

Aceleración: p_ax=0, p_ay=0, p_az=+16384

Al observar que los valores son cercanos a los anteriores se desconecta la placa Arduino para que el MPU6050 quede configurado con el último offset calculado como se muestra en la Ilustración 26.

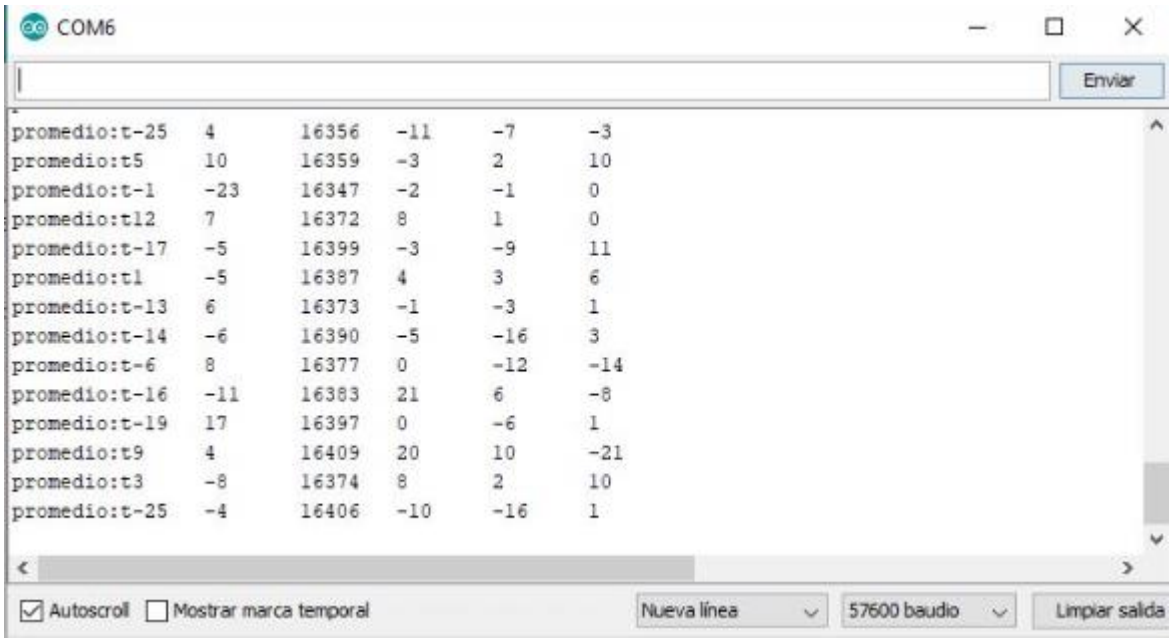


Ilustración 25 *Calibración de sensor inercial MPU6050*

2.3. Validación de los datos entregados por los sistemas de captura de movimiento.

Validación de los datos entregados por el sistema de captura mediante vídeo.

Se utilizó un inclinómetro digital de bajo costo como instrumento para validar los ángulos de las articulaciones entregados por el sistema de vídeo, midiendo la inclinación de un plano horizontal o vertical respecto a la superficie (ilustración 27). Para tomar estos datos fue necesario realizar nuevamente el montaje del sistema de video con los marcadores de colores descritos anteriormente que se adhirieron al cuerpo con cinta pegante, como se muestra en la ilustración 28.

Ilustración 26 *Mini inclinómetro digital utilizado para medir los ángulos.*

Ilustración 27 *Marcadores utilizados para las articulaciones del brazo.*

El objetivo de esta prueba era comparar los ángulos entregados por el sistema de captura mediante video con los entregados por el inclinómetro digital, con el fin de validar la precisión del sistema. Para no tener que tomar estos datos de manera manual se desarrolló una función utilizando la librería Pickle, esta librería permite representar un objeto Python como una cadena de bytes. Se pueden hacer multitud de cosas con dichos bytes, como, por ejemplo, almacenarlos en un archivo o base de datos, o transferirlos a través de una red. En este caso se guardan las posiciones y ángulos del brazo en estos archivos serializados, luego

para poder visualizar los datos se utiliza Pandas, la cual es una extensión de NumPy para manipulación y análisis de datos. Se generan archivos individuales para cada posición x, y de las articulaciones y para cada ángulo. Posteriormente estos archivos se juntan utilizando Pandas y se muestran de manera ordenada en una tabla como se muestra en la ilustración 29.

	codo_X	codo_Y	mano_X	mano_Y	hombro_X	hombro_Y	anguloCP	anguloCH
0	560.0	177.0	560.0	177.0	204.0	191.0	-0.278963	-0.039306
1	562.0	171.0	562.0	171.0	202.0	186.0	-0.282257	-0.041643
2	550.0	185.0	550.0	185.0	199.0	184.0	-0.252867	0.002849
3	545.0	186.0	545.0	186.0	202.0	184.0	-0.260928	0.005831
4	549.0	187.0	549.0	187.0	206.0	189.0	-0.267345	-0.005831
5	543.0	189.0	543.0	189.0	205.0	186.0	-0.256035	0.008876
6	534.0	192.0	534.0	192.0	200.0	188.0	-0.259434	0.011975
7	536.0	189.0	536.0	189.0	209.0	188.0	-0.263964	0.003058
8	530.0	193.0	530.0	193.0	207.0	192.0	-0.268814	0.003096
9	517.0	206.0	517.0	206.0	209.0	192.0	-0.244979	0.045423
10	516.0	210.0	516.0	210.0	200.0	191.0	-0.229677	0.060054

Ilustración 28 *Visualización de los datos guardados en objetos Pickle*

Para registrar los datos entregados por el inclinómetro a medida que se iban registrando los datos en los archivos, se hizo una modificación en el código principal agregando el texto con los datos del movimiento en el frame principal del algoritmo, luego con la función *videoWriter ()* de OpenCV se guarda un video con extensión .AVI que posibilita al final, asociar los ángulos entregados por el inclinómetro con el número de registro guardado como se puede apreciar en la ilustración 30.

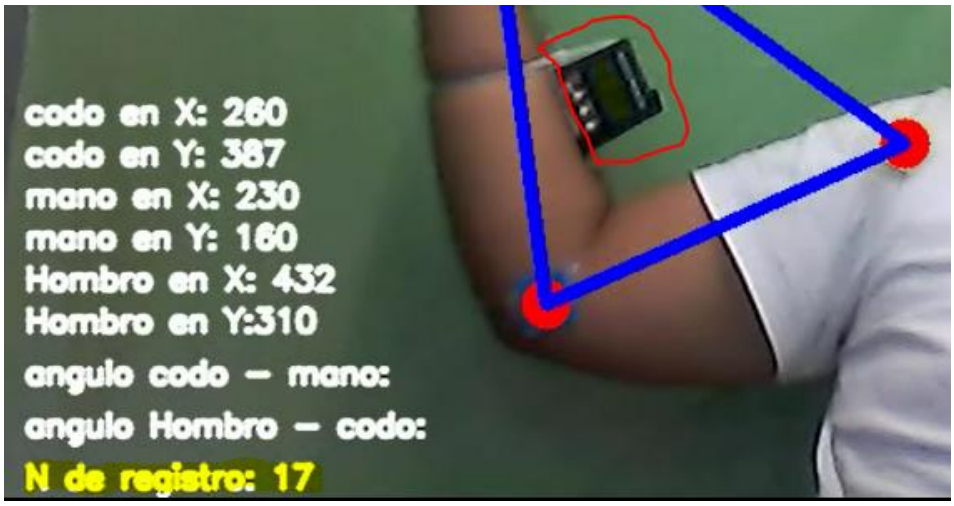


Ilustración 29 *Registros en frame utilizados para comparar los ángulos entregados por el sistema con los del inclinómetro.*

El inconveniente de realizar una grabación y a partir de ahí intentar observar el dato que mostraba el inclinómetro fue que por momentos el algoritmo identificaba el display de este como un color en el rango de los azules, lo cual causaba que se confundiera este display como el marcador del codo y se registraran datos erróneos del movimiento. Para corregir este problema se hicieron cambios en el código para cambiar de azul a rojo como color que se utiliza para el marcador del codo. Otra complicación de registrar los datos del inclinómetro de esta manera fue que no se podían observar con claridad los números que se estaban mostrando en el display del inclinómetro debido a la baja resolución de la cámara y factores de iluminación. Para corregir este problema se utilizó una linterna con trípode (ilustración 31) que alumbraba directamente al display permitiendo ver los números con una mayor nitidez.

Ilustración 30 *Linterna utilizada para obtener una mejor visualización de los datos entregados por el inclinómetro.*

Para el diseño de la prueba final del sistema de video había que encontrar el ángulo máximo y mínimo que era posible identificar por el sistema de captura de movimiento mediante video, además asociar manualmente los datos de los ángulos registrados en los archivos con los que mostraba el display.

Se tomaron 200 registros de los ángulos de manera manual y se organizaron en una tabla que se puede ver resumida en 10 registros tomados (ver tabla 3) generando una tabla para cada ángulo, ya que solo se contaba con un inclinómetro. Finalmente, estos datos iban a ser analizados para el desarrollo del experimento final del sistema.

TABLA DE ÁNGULO HOMBRO - CODO							
ÁNGULO INCLINOMETRO	HOMBRO CODO	CODO - MUÑECA	ERROR ABSOLUTO	ERROR RELATIVO		LIMITES DE DETECCIÓN	
5	9	95	-4	-80%		MINÍMO	70
14	8	86	6	43%		MÁXIMO	121
11	12	86	-1	-9%			
25	24	110	1	4%			
4	2	102	2	50%			
10	20	72	-10	-100%			
4	7	105	-3	-75%			
1	4	106	-3	-300%			
16	20	110	-4	-25%			
1	3	107	-2	-200%			
16	20	110	-4	-25%			

Tabla 3 *Ángulos obtenidos en hombro - codo*

Validación de los datos entregados por sistema de captura de movimiento mediante sensores inerciales.

Los datos recibidos mediante el sistema de sensores inerciales debían ser recibidos y posteriormente tratados con Python, por lo cual se utilizó la biblioteca pyserial que sirve para encapsular el acceso para el puerto serie. Se utilizó un MPU-6050 conectado a un Arduino para cada articulación del brazo; hombro, codo y muñeca; cada Arduino conectado de manera independiente a un puerto USB como se puede observar en la ilustración 32.

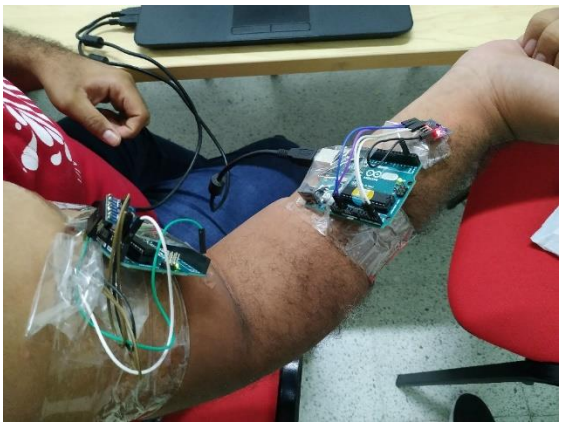


Ilustración 31 *Toma de datos con sensores inerciales.*

El código utilizado para leer los puertos seriales desde Python fue basado en el ejemplo AnalogInOutSerial de Arduino (CC-BY-SA 3.0) y se muestra a continuación:

```
import sys
import time

try:
    import serial
    arduino = serial.Serial('COM3', 9600, timeout=1.0)
    arduino2 = serial.Serial('COM4', 9600, timeout=1.0)

    # provocamos un reseteo manual de la placa para leer desde el principio
    arduino.setDTR(False)
    arduino2.setDTR(False)
    time.sleep(1)
    arduino.flushInput()
    arduino2.flushInput()
    arduino.setDTR(True)
    arduino2.setDTR(True)

except (ImportError, serial.SerialException):
    # No hay módulo serial o placa Arduino disponibles
    import io

    class FakeArduino(io.RawIOBase):
        """Clase para representar un "falso Arduino"
        """
        def readline(self):
            time.sleep(0.1)
            return b'sensor = 0\toutput = 0\r\n'

    arduino = FakeArduino()
    arduino2 = FakeArduino()


# Con la sentencia with el arduino se cierra automáticamente, ver
with arduino:
    while True:
        try:
            # En Python 3 esta función devuelve un objeto bytes
            line = arduino.readline()
            line2 = arduino2.readline()
```

```python
# Con la sentencia with el arduino se cierra automáticamente, ver
with arduino:
    while True:
        try:
            # En Python 3 esta función devuelve un objeto bytes
            line = arduino.readline()
            line2 = arduino2.readline()
            # Con errors='replace' se evitan problemas con bytes erróneos.
            # Con end='' se evita un doble salto de línea
            print(line.decode('ascii', errors='replace'), end='')
            print(line2.decode('ascii', errors='replace'), end='')

            tecla = sys.stdin.read(1)
            if tecla == 's':
                time.sleep(5)

        except KeyboardInterrupt:
            print("Exiting")
            break
```

Ilustración 32 *Código utilizado para leer los puertos seriales desde Python.*

En la ilustración 34 se muestra la salida que genera el código en ejecución.

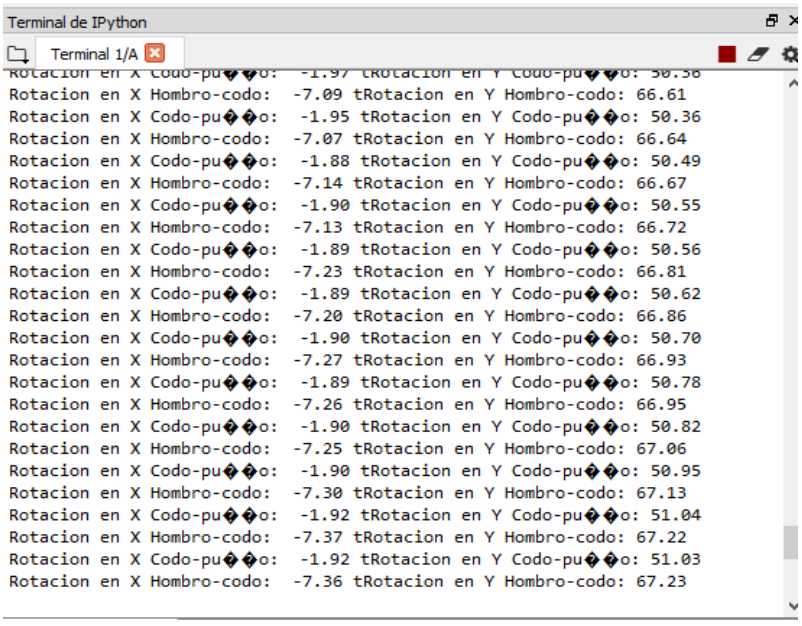


Ilustración 33 *Código para leer los puertos seriales en ejecución.*

Los datos obtenidos son los ángulos de rotación generados por el movimiento de cada articulación, estos datos de la misma manera que en el sistema de captura

mediante video, fueron guardados en archivos serializados con librería pickle a través de Numpy; luego fueron visualizados con un dataframe utilizando la librería Pandas. Lo anterior con el fin de facilitar los datos para la fusión que se propone realizar posterior al presente proyecto.

	Shombro_X	Shombro_Y	Smuneca_X	Smuneca_Y
0				
1	-0.30	-0.30	-0.30	-0.30
2	-1.46	-1.46	-1.46	-1.46
3	-0.59	-0.59	-0.59	-0.59
4	-2.91	-2.91	-2.91	-2.91
5	-0.88	-0.88	-0.88	-0.88
6	-4.32	-4.32	-4.32	-4.32
7	-1.16	-1.16	-1.16	-1.16
8	-5.69	-5.69	-5.69	-5.69
9	-1.44	-1.44	-1.44	-1.44
10	-7.03	-7.03	-7.03	-7.03

Ilustración 34 *datos de los sensores inerciales organizados en un data frame.*

3. RESULTADOS

Al analizar el sistema de captura mediante video se evidenciaron 2 aspectos importantes a corregir; por una parte, las posiciones y los ángulos que este algoritmo arrojaba estaban siendo calculados en base a pixeles, lo cual no representa ningún sistema métrico real, por lo cual fue necesario hallar la relación métrica – pixeles siendo elegidos los milímetros para este fin. Se desarrolló un módulo de objeto de calibración que básicamente usa la misma técnica que la utilizada para reconocimiento de las articulaciones, es decir, a través de color, utilizando el azul como color del objeto de calibración (ver Ilustración 36).

***Ilustración** 35 Etapa de calibración del sistema de captura de movimiento mediante video.*

Este módulo detecta el color azul en la imagen (Ver Ilustración 37), dibuja su contorno, calcula su punto medio, y determina el ancho en pixeles. Mediante una función se obtiene la relación métrica-pixeles y son guardados 20 registros en un archivo como se aprecia en la Ilustración 38, El resultado individual de cada

medición se promedia para obtener una relación más precisa, obteniendo un promedio total de 93.8%.

Out[4]:

	Relacion Metrica-Pixel
0	0.925585
1	0.930415
2	0.938895
3	0.929996
4	0.934248
5	0.934248
6	0.940416
7	0.945101
8	0.945101
9	0.940209
10	0.940209
11	0.941589
12	0.941589
13	0.942555
14	0.942555

```
In [5]: print(np.mean(PixelesxMetrica))

0.938856347786532
```

Ilustración 37 Resultados de la calibración a una distancia de 44.5 cm del lente de la cámara.

Ilustración 36 Frame de calibración.

El otro aspecto para corregir en el sistema de captura mediante video fue la distorsión de la cámara, para ello se utilizó la técnica de calibración con tablero de ajedrez asimétrico (ver Ilustración 39) proporcionado por OpenCV. Con esto se obtuvieron los parámetros intrínsecos y extrínsecos de la cámara que para trabajos futuros pueden ser usados para realizar una estimación 3d desde el plano 2D.

Seguido de esto, se obtuvieron los datos mediante el proceso de calibración (ver Ilustración 38) de cámara, y se guardaron como una matriz en un archivo (ver ilustración 39).

El error de calibración total obtenido fue de: 0.1566693243193671, lo cual se considera una buena calibración, teniendo en cuenta que entre más cercano a cero se encuentre este número mejor habrá sido el resultado de la calibración.

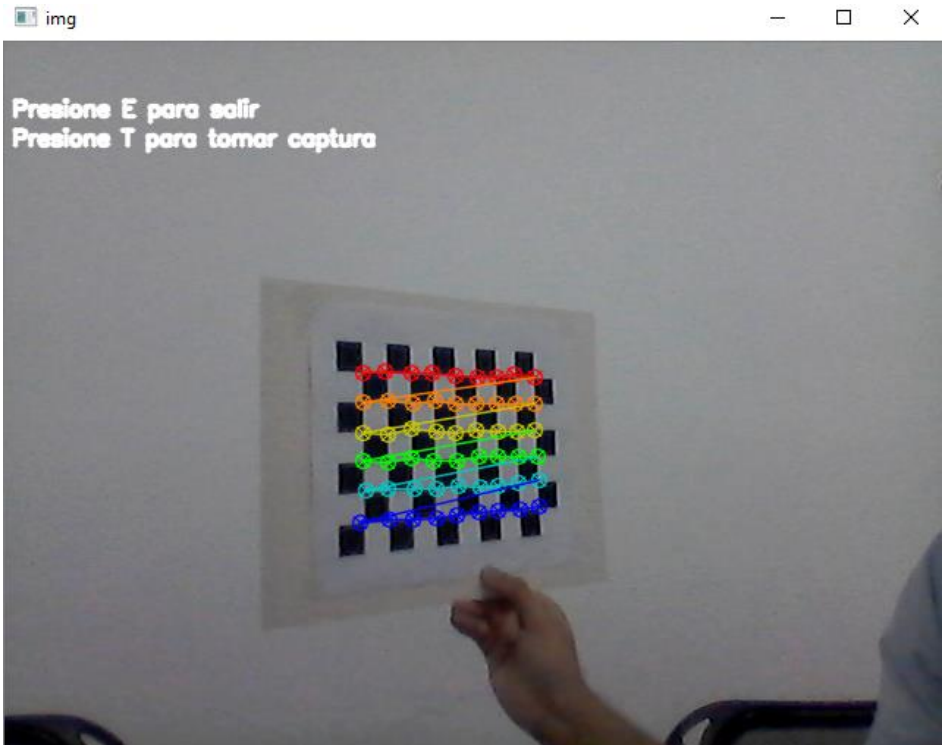


Ilustración 38 *frame de calibración de cámara.*

```
{'mtx': array([[711.76577821,   0.        , 329.86144259],
       [  0.        , 710.64307305, 226.82242021],
       [  0.        ,   0.        ,   1.        ]]), 'dist': array([[ 3.10284626e-01,
        -1.20906443e-02,  9.38117627e+00]]), 'rvecs': [array([[0.27919424],
       [0.0066694 ],
       [0.07242155]]), array([[-0.06974452],
       [ 0.60860286],
```

Ilustración 39 *Resultados de la calibración de cámara.*

Una vez obtenido los parámetros intrínsecos de la cámara, se realiza el proceso de corrección de la distorción de la misma, usando la función

cv2.imshow("undistort", dst) e ingresando como argumento la imagen sin distorsión con la librería OpenCV (Ver anexo A). En la Ilustración 40 se muestra la comparación de las imágenes antes y despues de la corrección.

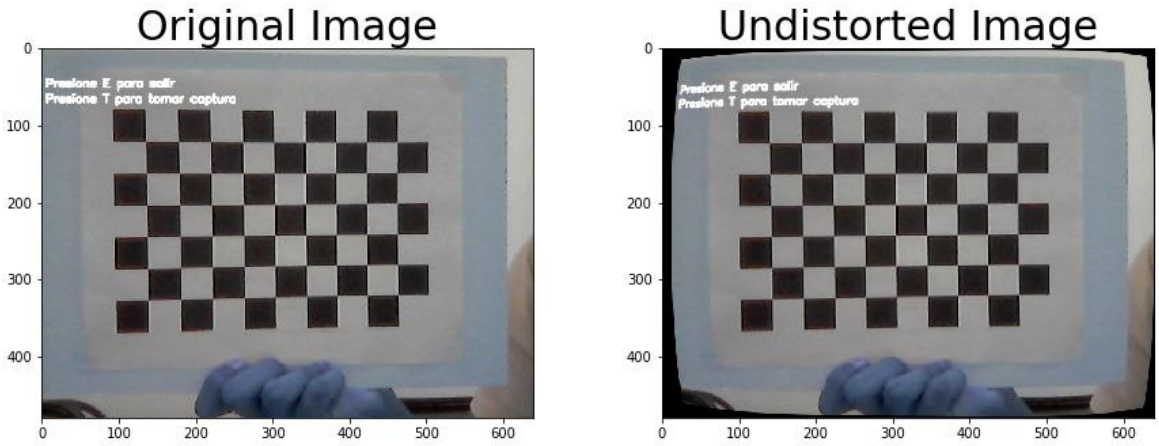


Ilustración 40 *Distorsión corregida de la cámara. (fuente propia).*

Posterior a esto, se inició la construcción del sistema de captura mediante sensores inerciales, para lo cual se usaron tres (3) IMU-6050 y tres (3) placas Arduino uno, que entregan los ángulos de rotación de las articulaciones del brazo implementando un filtro de complemento acelerómetro + giroscopio. De esta manera, los IMU amortiguan las variaciones bruscas de aceleración y el giroscopio con un filtro que tiene gran influencia cuando hay rotaciones rápidas. Los ángulos de rotación son procesados y guardados en archivos serializados a través del lenguaje de programación Python, facilitando el análisis de los datos generados por el movimiento del miembro superior.

Continuando con los pasos establecido en la metodología se calculó el error relativo de los datos, validándolos con el inclinómetro en el ángulo codo – hombro. Este error relativo corresponde en porcentaje a la diferencia entre el ángulo marcado por el inclinómetro y el ángulo marcado por el sistema. Teniendo como

error relativo máximo un 13% y un promedio de error relativo del 3% (ver Ilustración 41).

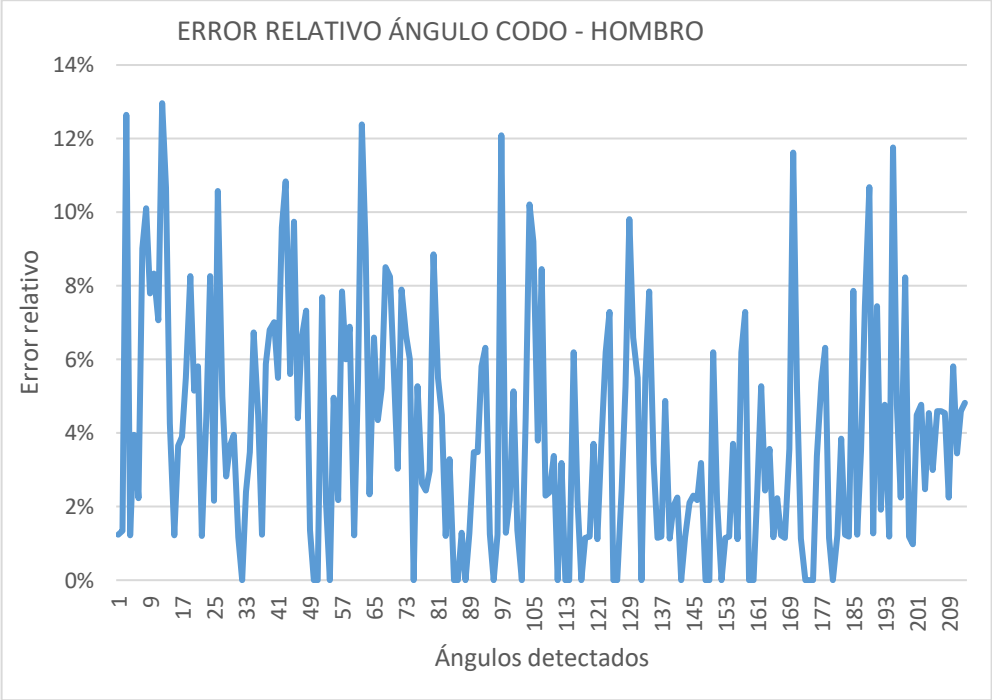


Ilustración 41 *Resultados de pruebas para sistema de captura de video.*

Del mismo modo la gráfica de la Ilustración 42 muestra el error relativo de los datos validados con el inclinómetro del ángulo de rotación en el codo del sistema de captura de movimiento mediante sensores inerciales. Teniendo como error relativo máximo un 16% y un promedio de error relativo del 4%.

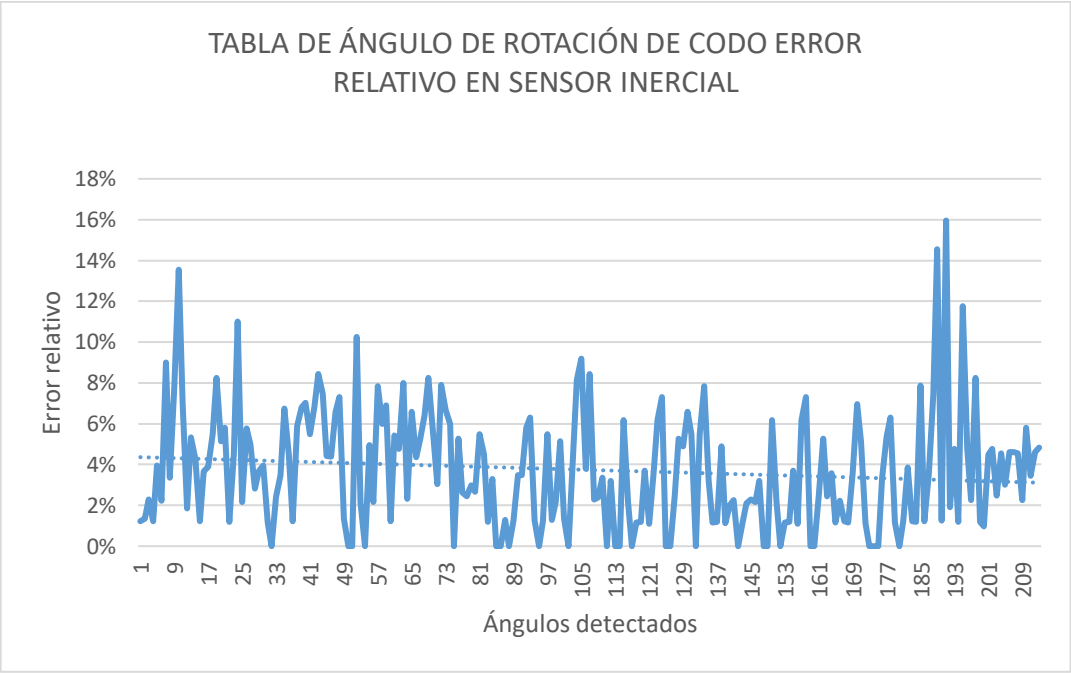


Ilustración 42 *Resultados de pruebas para sistema de sensores inerciales.*

Con el propósito de utilizar materiales y recursos de hardware de bajo costo, los gastos fueron minimizados al máximo. La Tabla 4 muestra los materiales utilizados y los precios de cada uno a fecha de 15 de noviembre de 2019. Obteniendo un costo total de $205.000 pesos colombianos, este costo solo representa un 13% del costo de un sensor inercial de calidad profesional de la marca XSENS [1].

Tabla 4 *Costos de los materiales y recursos de hardware utilizados.*

Cantidad	Materiales	Precio Unitario	Precio Total
3	Sensores MPU-6050	$14.000	$42.000
3	Arduino Nano	$14.000	$42.000
1	Marcadores para articulaciones	$2.000	$2.000
1	Cinta doble faz 3M	$11.300	$11.300
-	Software (OpenCV)	Libre	Libre
1	Paquete Jumpers hembra-macho	$5.700	$5.700
3	Cables USB Tipo A	$4.000	$12.000
1	Tela Verde Chroma key	$55.000	$55.000
1	Cámara Web	$35.000	$35.000
Precio Total			$205.000

4. CONCLUSIONES Y RECOMENDACIONES

Con el desarrollo de este proyecto se construyó un prototipo de dos sistemas de captura de movimiento, en uno de ellos utilizando captura de video y en el otro utilizando sensores inerciales MPU-6050. Estos sistemas permiten obtener las posiciones y ángulos de las articulaciones del brazo en el espacio. Para lograr este fin, fue necesario iniciar con una fase de análisis de los sistemas a desarrollar para la captura de movimiento: captura por video y captura mediante sensores inerciales, que corresponde al primer objetivo de este proyecto.

Para el sistema de captura por video, se inició con el análisis de la librería OpenCV para comprender el funcionamiento de este sistema que ya había sido desarrollado en trabajos previos; también la técnica de identificación de las articulaciones que fue utilizada, así como también la forma en que se hace el cálculo de los ángulos generados por el movimiento y los ajustes posteriores a realizar en el código del sistema. Como resultado de esta fase se definieron las acciones a realizar en el sistema de captura mediante video: establecer la relación métrica por pixeles, obtener los ángulos y posiciones generadas con base a la métrica real, obtención de los parámetros intrínsecos de la cámara realizando una fase de calibración y, por último, guardar en archivos serializados los datos generados en las pruebas para poder analizarlos.

Seguido de esto, se realizó una fase de análisis de los sensores a utilizar para recoger los datos del movimiento del brazo y se definieron también las tareas a realizar para este sistema: desarrollo del código para cargar en las placas Arduino, calibración de los sensores inerciales y desarrollo del código para recibir los datos generados por el sistema con el lenguaje de programación Python.

Siguiendo la metodología propuesta y dando respuesta al segundo objetivo planteado, se procedió a construir los sistemas de captura de movimiento realizando pruebas de concepto y teniendo en cuenta las recomendaciones de los

proyectos definidos en el estado del arte. Una vez se logró la construcción de los sistemas se iniciaron las pruebas de validación de los datos entregados contrastándolos contra un inclinómetro digital para calcular el error relativo y absoluto presentes en los sistemas. Para ello se guardaron los datos en archivos serializados utilizando Pickle, Numpy y posteriormente pudieron ser visualizados y manipulados utilizando Pandas.

Dando respuesta al tercer objetivo planteado, se calculó el de error relativo en cada sistema, dando como resultado un 13% para el sistema de captura por video y 16% para el sistema de captura por sensores inerciales, los cuales fueron contrastados con un inclinómetro digital, pero este dispositivo en realidad no ofrece una precisión confiable ya que no cuenta con una certificación oficial del margen de error en los datos que ofrece, además la forma en la que se sujetó al cuerpo para realizar las pruebas tampoco garantiza que estuviera completamente nivelado por la forma natural del brazo. Sin embargo, estos datos pueden ser tomados como una referencia para que al momento de realizar la fusión que se plantea en el proyecto macro se pueda evidenciar si hubo una mejora en los resultados obtenidos.

Se logró la captura de movimiento del miembro superior utilizando dos sistemas de captura de movimiento de diferente técnica utilizando materiales y recursos de hardware de bajo costo. El sistema de captura de movimiento mediante video es capaz de obtener las posiciones y ángulos del brazo basándose en una métrica real (milímetros) y corrigiendo la distorsión que pueda presentar la cámara gracias a su calibración.

Los sistemas construidos capturan de forma correcta el movimiento, logrando con esto cumplir con el propósito general del proyecto, aun así ,el sistema de captura por video cuenta con algunas limitaciones en cuanto al rango del movimiento que es capaz de detectar, para el codo debe estar mínimo en un ángulo de 68 grados y un ángulo máximo de 114 grados, aunque es muy poco el rango que se puede

captar debido a la técnica utilizada, se podría en investigaciones posteriores realizar una estimación 3D desde el plano 2D para tener los datos más cerca al formato de datos que manejan los sensores inerciales y de este modo facilitar la fusión de los datos de ambas fuentes, dando más peso a cada sistema en determinados casos. De manera análoga el sistema de sensores inerciales es capaz de captar el movimiento utilizando un filtro complemento entre el acelerómetro y giroscopio para obtener un mejor resultado en las medidas.

El resultado obtenido con este proyecto es una contribución técnica al macroproyecto sobre el cual se enmarca este proyecto, el cual pretende el desarrollo de un algoritmo de fusión de datos mediante captura de video y sensores inerciales para robótica colaborativa. También facilitará el desarrollo de productos de entretenimiento digital, como videojuegos que involucren realidad virtual y permitan al jugador interactuar de manera más vivida con el espacio; o producción de contenidos animados para videojuegos, animación 3D y producción audiovisual. También pude permitir el crecimiento en investigaciones en torno a la robótica colaborativa y la forma de dotar a una máquina de un sistema que pueda anticiparse a los movimientos de un ser humano para evitar accidentes en entornos productivos.

Para trabajos futuros recomendamos tener en cuenta los siguientes aspectos:

- Contrastar los resultados obtenidos con sistemas profesionales de captura de video con marcadores tales como Vicon para tener pruebas válidas según lo establecido por los investigadores del tema a nivel internacional.
- Desarrollar una interfaz que permita inicializar fácilmente los scripts correspondientes a cada sistema.
- Realizar una estimación 3D para el sistema de captura de movimiento mediante video con el fin de homogenizar los datos para la fusión utilizando los parámetros obtenidos en la calibración de cámara.

- Utilizar una buena iluminación para el uso del sistema de captura de movimiento mediante video.

BIBLIOGRAFÍA

[1] xsens, «xsensshop.com,» [En línea]. Available: https://shop.xsens.com/shop/. [Último acceso: 23 10 2019].

[2] Infomed Especialidades, «Medicina de rehabilitación biomecanica,» [En línea]. Available: http://www.sld.cu/sitios/rehabilitacion-bio/temas.php?idv=20594. [Último acceso: 15 11 2019].

[3] F. J. P. Delgado, Problemario de algoritmos resueltos con diagramas de flujo y Pseudocodigo, Aguascalientes: Departamento Editorial de la Dirección General de Difusión y Vinculación, 2014.

[4] E. M. i. García, «Visión artificial,» 2012. [En línea]. Available: https://openlibra.com/es/book/vision-artificial. [Último acceso: 10 04 2019].

[5] etitudela, «etitudela.com,» [En línea]. Available: http://www.etitudela.com/celula/downloads/visionartificial.pdf. [Último acceso: 11 04 2019].

[6] M. W. Brian Kitawa, MoCap for Artists: Workflow and Techniques for Motion Capture, FocalPress, 2008.

[7] J. R. M. Wendy Janett Guzmán González, «Fusión de datos en redes de sensores,» 2014.

[8] D. Hall, Mathematical Techniques in Multisensor Fusion, 2 ed., Boston: Artech House, 2004.

[9] D. Abeijón, «Universitat Politécnica de Catalunya Barcelonatech,» 2007. [En línea]. Available: https://upcommons.upc.edu/bitstream/handle/2099.1/4376/03.pdf?sequence=28.

[10] J. L. Elkin Torres, *Sistema de captura de movimiento del miembro superior mediante opencv,* Cartagena: Universidad del Sinú, 2019.

[11] Aprendiendo Arduino, «Aprendiendo arduino,» 09 07 2017. [En línea]. Available: https://aprendiendoarduino.wordpress.com/2017/07/09/i2c/. [Último acceso: 18 10 2019].

[12] NAYLAMP MECHATRONICS, «NAYLAMP MECHATRONICS,» 2016. [En línea]. Available: https://naylampmechatronics.com/blog/45_Tutorial-MPU6050-Aceler%C3%B3metro-y-Giroscopio.html. [Último acceso: 20 10 2019].

[13] UniPython, «Calibración de la Camará OpenCV,» 2018. [En línea].

[14] OpenCV, «docs.opencv.org,» 2015 junio 2019. [En línea]. Available: https://docs.opencv.org/2.4/doc/tutorials/calib3d/camera_calibration/camera_calibration.html. [Último acceso: 30 09 2019].

[15] H. L. A. K. Nour-Eddin El Faouzi, «Fusión de datos en sistemas de transporte inteligentes: avances y desafíos,» 2011. [En línea]. Available: https://dl.acm.org/citation.cfm?id=1862592.

[16] D. Abeijon, «upcommons.upc.edu,» 2007. [En línea]. Available: https://upcommons.upc.edu/bitstream/handle/2099.1/4376/03.pdf?sequence=28. [Último acceso: 10 04 2019].

[17] D. L. H. J. Llinas, *An introduction to multisensor data fusion,* 1997.

[18] H. L. A. K. Eddin El Faouzi, *Data fusion in intelligent transportation systems: progress and challenges,* 2010.

[19] V. T. Guillermo Navarro-Arribas, *Information fusion in data privacy: A survey,* 2012.

[20] A. Khalifa F. y Alouani, *Survey of watershed modeling and sensor data fusion. System Theory,* 2009.

[21] M. Ahmed M. y Abdel-Aty, *A data fusion framework for real-time risk assessment on freeways, Transportation Research Part C: Emerging Technologies.,* 2013.

[22] S. Vasudevan, *Data fusion with gaussian processes, Robotics and Autonomous Systems.,* 2012.

[23] R. C. A. G. y. M. J. M. Dutta, *3D mapping of buried underworld infrastructure using dynamic Bayesian network based multi-sensory image data fusion, Applied Geophysics,* 2013.

[24] E. H. T. y. A. A. Garcia, *Bayes filter for dynamic coordinate measurements – accuracy improvment, data fusion and measurement uncertainty evaluation, Measurement.,* 2013.

[25] S. Grasing D. y Desai, *Data fusion methods for small arms localization solutions. Information Fusion (FUSION),* 2012.

[26] Zervas, E., Mpimpoudis, A., Anagnostopoulos, C., Sekkas, O. y Hadjiefthymiades, S, *Multisensor data fusion for fire detection, Information Fusion,* 2011.

[27] M. N. N. y. K. M. Badamchizadeh, *Optimization of data fusion method based on kalman filter using genetic algorithm and particle swarm optimization. Computer and Automation Engineering (ICCAE),* 2010.

[28] P. S. G. C. y. L. J. N. Chatziagorakis, *Design automation of cellular neural networks for data fusion applications, icroprocessors and Microsystems,* 2012.

[29] J. A. Rodger, *Toward reducing failure risk in an integrated vehicle health maintenance system: A fuzzy multi-sensor data fusion kalman filter approach for fIVHMSg, Expert Systems with Applications,* 2012.

[30] A. M. C. A. G. V. F. y. P. P. Pinto, *An approach to implement data fusion techniques in wireless sensor networks using genetic machine learning algorithms, Information Fusion,* 2014.

[31] H. R. Cristóbal Chérigo, «Evaluación de algoritmos de fusión de datos para estimación de la orientacion de vehiculos aereos no tripulados,» Ciudad de Panama, 2017.

[32] QuantDare, «QuantDare.com,» 28 03 2014. [En línea]. Available: https://quantdare.com/filtro-kalman/. [Último acceso: 19 04 2019].

[33] A. G. I. B. M. M. D. R. Jorge Garcia, «Deteccion y Seguimiento de Personas Basado en Estereovision y Filtro de Kalman,» *Revista Iberoamericana de Automática e Informática industrial ,* nº 9, pp. 453-461, 2012.

[34] G. M. E. B. Julio Muñoz, *Método de fusión de datos de fuentes heterogéneas para mantener la consistencia de los datos,* Veracruz, 2017.

[35] R. y. c. Patología, «Inclinómetros, descripción, uso y recomendaciones,» 2015. [En línea].

[36] J. E. M. F. A. B. D. J. L. T. Andrés Leonardo González Gómez, «http://www.scielo.org.co,» junio 2013. [En línea]. Available: http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S1692-17982013000100002. [Último acceso: 29 09 2019].

[37] J. Rowberg, «github,» 2014. [En línea]. Available: https://github.com/jrowberg/i2cdevlib/tree/master/Arduino/MPU6050. [Último acceso: 15 10 2019].

[38] mathworks , «mathworks.com,» [En línea]. Available: https://la.mathworks.com/discovery/kalman-filter.html. [Último acceso: 3 11 2019].

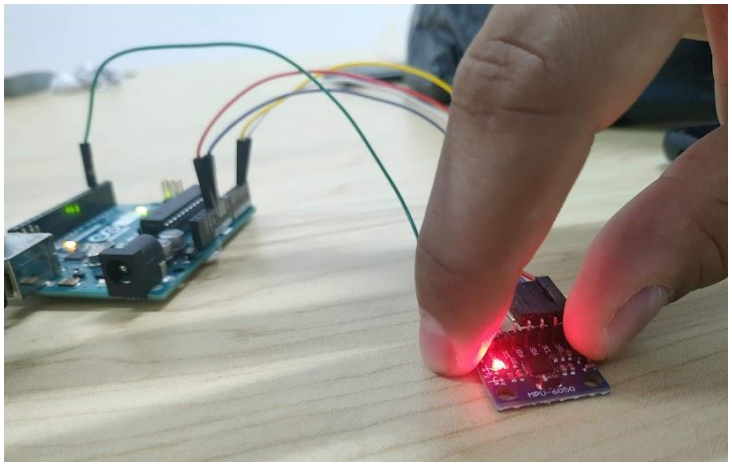
[39] AnaliticaWeb, «analiticaweb.es,» 20 07 2017. [En línea]. Available: https://www.analiticaweb.es/algoritmo-knn-modelado-datos/. [Último acceso: 4 11 2019].

[40] M. M. Juan Vázquez, «Researchgate,» [En línea]. Available: FUSIÓN DE ALGORITMOS PARA DETECTAR OBJETOS EN MOVIMIENTO Juan Vázquez1, Manuel Mazo. [Último acceso: 23 03 2019].

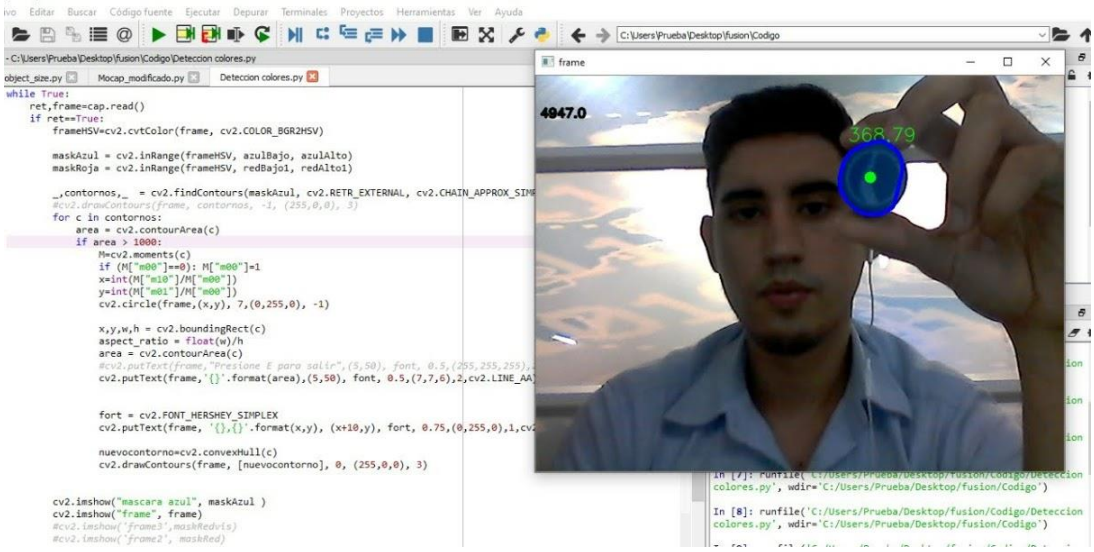
ANEXOS

Fase de pruebas.

Calibración de IMU-6050.



Construcción del módulo de calibración en video.



Fase de transferencia de la metodología utilizada.