UNIVERSIDAD NACIONAL DEL ALTIPLANO

FACULTAD DE INGENIERÍA MECÁNICA ELÉCTRICA, ELECTRÓNICA Y SISTEMAS

ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA

"ANÁLISIS Y DISEÑO DE UN SISTEMA COMPUTARIZADO PARA LA INTERPRETACIÓN COLORIMÉTRICA Y MORFOMÉTRICA DEL CHENOPODIUM"

TESIS

PRESENTADO POR:

JHON WILBER TAPIA PINTO

PARA OPTAR EL TITULO PROFESIONAL DE:

INGENIERO ELECTRÓNICO

PUNO - PERÚ

2014

UNIVERSIDAD NACIONAL DEL ALTIPLANO

FACULTAD DE INGENIERÍA MECÁNICA ELÉCTRICA, ELECTRÓNICA Y SISTEMAS

ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA

"ANÁLISIS Y DISEÑO DE UN SISTEMA COMPUTARIZADO PARA LA INTERPRETACIÓN COLORIMÉTRICA Y MORFOMÉTRICA DEL CHENOPODIUM"

TESIS PRESENTADA POR:

JHON WILBER TAPIA PINTO

PARA OPTAR EL TITULO PROFESIONAL DE:

INGENIERO ELECTRÓNICO

APROBADA POR EL JURADO REVISOR CONFORMADO POR:

PRESIDENTE : _____
Mg. Ing. IVAN DELGADO HUAYTA

PRIMER MIEMBRO : _____
Ing. MARCO ANTONIO RAMOS GONZALES

SEGUNDO MIEMBRO : _____
Ing. KARLOS ALEXANDER CCANTUTA CHIRAPO

DIRECTOR : _____
Mg. Ing. MARCO ANTONIO QUISPE BARRA

ASESOR : _____
Ing. IRENIO LUIS CHAGUA ADUVIRI

PUNO PERU
2014

ÁREA: Automatización e instrumentación
TEMA: Modelamiento y simulación de sistemas de control

DEDICATORIA

A Dios y a mis Padres

AGRADECIMIENTOS

Este trabajo no se habría podido realizar sin la colaboración de las personas que me han brindado su ayuda, sus conocimientos y su apoyo. Quiero agradecerles a todos ellos cuanto han hecho por mí, para que este trabajo saliera adelante de la mejor manera posible.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

LISTA DE ABREVIATURAS

CPU : Unidad Central de Procesamiento

HDMI : High-Definition Multimedia Interface

Mhz : Megahercio

Mm : Milímetro

Ms : Milisegundo

PAL : Phase Alternation Line

Px : Pixel

RGV : Red, Green, Blue (Modelo de color)

SDK : Software Development Kit

TCP : Transfer Control Protocol.

UDP : User Datagram Protocol

USB : Universal Serial Bus

RESUMEN

El crecimiento de la importancia de la quinua como fuente alimentación a nivel mundial conlleva a la búsqueda de la mejora de los procesos de producción y comercialización del grano de quinua con el objetivo de satisfacer la demanda tanto de calidad como de cantidad, y una buena alternativa de solución a este problema es la automatización de los procesos de producción. Actualmente el productor de la región está en un proceso de cambio y asimilación de nuevas tecnologías aplicados a la producción. Sin embargo el proceso de selectividad y control de calidad que cada vez cobra mayor importancia aún no se implementa debidamente o está en una fase inicial de implementación en algunos centros de producción y peor aún para los productores locales debido a los costos elevados para su implantación. Es por tal motivo que en la presente investigación se analiza y diseña un sistema para la interpretación de las características colorimétricas y morfométricas de granos de quinua para su posterior aplicación en máquinas de control de calidad y/o máquinas de selección óptica de quinua. La investigación contempla el diseño del sistema estructural y de comportamiento basado principalmente en un dispositivo de captura de imágenes y un computador de procesamiento de imágenes, aplicándose un proceso de captura, adquisición de datos, procesamiento y transferencia de resultados tanto como a una interfaz de usuario como a un elemento de control. Este diseño está orientado para su aplicación futura en máquinas de control de calidad y selección de granos.

Palabras Clave: Chenopodium quinoa, producción de quinua, clasificación, maquina clasificadora, visión por computadora.

ABSTRACT

The growing importance of quinoa as a food source worldwide search leads to improved production processes and marketing of quinoa grain in order to meet the demand of both quality and quantity, and a good alternative solution to this problem is the automation of production processes. Currently the producer of the region is in a process of change and assimilation of new technologies applied to production. However, the process of selectivity and quality control increasing importance is even not properly implemented or is in an early stage of implementation in some production facilities and even worse for local producers due to high costs for implementation. it is for this reason that in the present research analyzes and designs a system for the interpretation of the colorimetric and morphometric characteristics of quinoa grain for subsequent application in quality control machines and / or machines for optical selection of research quinua. La includes the design of the structural system and behavior based primarily on an imaging device and a computer image processing, applying a process of capture, data acquisition, processing and transfer of results both as a user interface as a control element. This design is targeted for future application in quality control machines and selection of grains.

Key Words: Chenopodium quinoa, quinoa production, classification, sorting machine, computer vision.

INTRODUCCIÓN

El comercio de alimentos en la actualidad ha tomado nuevas características influenciado por la globalización, la competitividad y las ventajas comparativas, es así que los cultivos andinos poco destacados y comercializados en el pasado, ahora tienen la oportunidad de integrarse en el comercio internacional y con mayor dinámica en el comercio nacional.

Uno de estos cultivos es la Quinua (Chenopodium Quinoa), grano andino que contribuye de manera importante en la alimentación de numerosas familias rurales y urbanas del Perú y del mundo y cuya demanda se está incrementando en los últimos años, siendo necesario que los productores, procesadores y comercializadores se adecúen a las nuevas exigencias del mercado, tanto en calidad como en tiempo.

La quinua es un alimento rico ya que posee los 8 aminoácidos esenciales para el humano, lo cual hace que la quinua sea un alimento muy completo y de fácil digestión por lo que es el elemento principal de alimentación y una fuente de recurso económico para el productor puneño con mucho potencial en el mercado internacional.

Actualmente debido al crecimiento de la demanda tanto en cantidad como en calidad, la producción de quinua requiere de estrictos procesos de producción y control de calidad, por lo que es necesario contar con sistemas automatizados capaces de realizar este trabajo aprovechando recursos y tiempo.

En el proceso de producción de la quinua, uno de los subprocesos más importantes es la clasificación de la quinua según parámetros establecidos originados por las exigencias de calidad del consumidor y el destino de uso del pseudo-cereal como materia prima en otros productos derivados. Es así que los sistemas de clasificación y selección de quinua se convierten en elementos indispensables entre los procesos de producción y comercialización de estos granos, utilizados actualmente en las industrias de producción y transformación de alimentos y en un proceso de incorporación por los productores enfocados a la producción de quinua.

El presente trabajo se muestra en capítulos los cuales tratan los siguientes temas:

Planteamiento y Formulación del problema, Mostramos el planteamiento del problema, formulación, justificación y objetivos. Se plantea la hipótesis que se intenta demostrar y las limitaciones de la investigación.

Marco teórico y Conceptual, Se presentan los antecedentes de la investigación, el marco teórico donde se definen los conceptos que son el apoyo teórico sobre las cuales se apoya el desarrollo de la presente investigación, además se describe en el marco conceptual términos técnicos que ayudaran a comprender de una mejor manera los conceptos usados.

Método de Investigación, Se describe el tipo de investigación, el ámbito del estudio, la metodología de desarrollo del sistema.

Exposición y análisis de los resultados, Se describe la estructura, funcionalidad y resultados del sistema desarrollado.

Conclusiones, Presentamos conclusiones del trabajo de investigación.

Recomendaciones, Se muestra algunas recomendaciones.

CAPITULO I.

PLANTEAMIENTO DEL PROBLEMA, ANTECEDENTES Y OBJETIVOS DE LA INVESTIGACIÓN

1.1. PLANTEAMIENTO DEL PROBLEMA

La producción de quinua en grandes cantidades requiere de estrictos procesos de clasificación y control de calidad de cada una de las diferentes variedades de quinua, por lo es necesario contar con sistemas automatizados capaces de realizar este trabajo aprovechando recursos y tiempo.

Actualmente existen sistemas muy complejos y costosos capaces de reconocer e interpretar patrones en imágenes aplicados a la selección y control de alimentos siendo estos accesibles solo para centros de investigación o laboratorios. Estos sistemas están dirigidos a la aplicación en diferentes áreas en la industria alimenticia teniendo como objetivo el procesamiento de los alimentos más comerciales y conocidos a nivel mundial donde la Quinua aún no está incluida.

Actualmente existen pocos centros de procesamiento de alimentos que poseen máquinas o sistemas de selección óptica capaces de clasificar cereales, y que estas puedan adaptarse a la selección de granos de quinua de manera eficaz. Además en la mayoría de estos centros de producción de la región solo se hace el uso de las máquinas y/o sistemas automáticos, mas no se desarrolla investigación para el mejoramiento de los sistemas existentes ya que resulta una gran inversión económica y de tiempo.

Un aspecto importante y decisivo en el proceso de selección y clasificación de granos mediante sistemas automatizados, específicamente en sistemas electromecánicos con censado óptico, es contar con un subsistema computacional de análisis y procesamiento de imágenes eficiente, eficaz y flexible, capaz de procesar las diferentes variedades de quinua con un alto grado de confiabilidad en los resultados.

Actualmente estos sistemas o máquinas de selección óptica lamentablemente no soy muy comerciales en el Perú y tiene costos inaccesibles para los productores emprendedores de nuestro país. Es así que se pretende realizar el diseño de un sistema accesible y eficaz para el propósito descrito anteriormente, desarrollado un subsistema electrónico e informático capaz de identificar las características colorimétricas y morfométricas de la quinua mediante el procesamiento de imágenes con resultados fáciles de interpretar y abriendo la posibilidad de usar estos resultados en un sistema mecánico - electrónico capaz de seleccionar y/o clasificar grandes cantidades de quinua según el color, la forma o el tamaño de cada grano de manera automática.

1.2. FORMULACIÓN DEL PROBLEMA

Determinar el diseño estructural y de comportamiento de un sistema capaz de analizar y procesar las características colorimétricas y morfológicas de los granos de Quinua en grandes cantidades mediante procesamiento digital de imágenes.

1.3. JUSTIFICACIÓN DE LA INVESTIGACIÓN

La importancia del diseño de un sistema de interpretación de características de la quinua mediante procesamiento de imágenes se justifica:

- En la particularidad y relevancia ya que es una investigación nueva y que aún no se conoce sobre investigaciones específicas sobre el tema en el país o en el mundo, y es muy relevante porque la producción de la quinua está en aumento.
- Se justifica en el aporte al desarrollo tecnológico de la región, puesto que representa un primer paso en el proceso de automatización de los procesos de control de calidad y clasificación de quinua para su aplicación en los centros de producción de la región de Puno.
- El resultado de la presente investigación, podrá ser aplicada en la construcción de máquinas de control de calidad y principalmente en máquinas clasificadoras de quinua y pseudo-cereales y de esta investigación dependerá la eficacia y eficiencia del desempeño de las máquinas en las que podrían ser aplicados.

1.4. ANTECEDENTES DE LA INVESTIGACIÓN

La tesis "CARACTERIZACIÓN MORFOMÉTRICA Y COLORIMÉTRICA DEL GRANO PULIDO DE ARROZ (Oryza sativa L.) 'MORELOS-A98' Y COMERCIAL, UTILIZANDO RECONOCIMIENTO DE PATRONES Y ANÁLISIS FRACTAL DE IMÁGENES DIGITALES (RPAFID)". Elaborado por Jacqueline Valenzuela Riaño en el año 2007. Con el planteamiento del Objetivo General: "Caracterizar y diferenciar morfométrica y colorimétricamente granos de arroz de la variedad Morelos de otros materiales de arroz, utilizando el Reconocimiento de Patrones (RP) y el Análisis Fractal de Imágenes Digitales (AFID)", dando las siguientes conclusión principal: Se estableció y validó estadísticamente una metodología para la captura y procesamiento de imágenes a través del reconocimiento de patrones y el análisis fractal (RP-AFID), con la que se realizó la caracterización morfométrica del grano de arroz de la variedad "Morelos A-98". Utilizando esta técnica se

encontraron diferencias estadísticas en todos los descriptores de medición entre la variedad “Morelos A-98” y al menos nueve productos comerciales.

Perspectivas del Trabajo: Este trabajo constituye un esfuerzo inicial para aplicar una serie de técnicas modernas de análisis, utilizando el reconocimiento de patrones y los conceptos de la geometría de fractales, aunados al procesamiento digital de imágenes. Los resultados obtenidos plantean diversas interrogantes que podrían delinear el rumbo para trabajos futuros. Un primer acercamiento podría dirigirse principalmente a plantear la modificación de las normas de calidad con el objeto de obtener lo que se conoce como “denominación de origen” y que constituye un reconocimiento a la calidad del producto, en este caso, el arroz variedad “Morelos A-98”. Para ello, es importante conjuntar la caracterización morfométrica con otros resultados, particularmente con el análisis de la microestructura del endospermo, la caracterización de los gránulos de almidón (relación amilosa y amilopectina) y la disposición estructural de éstos para conformar la característica genotípica conocida como “panza blanca”. (Valenzuela Riaño, 2007).

La tesis “DISEÑO Y CONSTRUCCIÓN DE UNA MÁQUINA CLASIFICADORA DE MAÍZ PARTIDO” Elaborado por Amangandi Aguilar Jorge Washington y Lamiña Maygua Fabián Rolando en el año 2013. Objetivo General: Diseñar y construir una máquina clasificadora de maíz partido.

Conclusiones: Luego de haber analizado cuidadosamente cada opción de las diferentes alternativas, mediante una matriz de decisión, se determinó que una máquina clasificadora basado en el mecanismo biela-manivela, es la más idónea para los objetivos planteados en este proyecto. El objetivo principal se ha cumplido al diseñar y construir una máquina clasificadora de maíz partido, que satisface con los parámetros funcionales y requerimientos operacionales planteados al inicio de este trabajo (Amangandi Aguilar y Lamiña Maygua, 2013).

La tesis “DISEÑO DE UNA MÁQUINA CLASIFICADORA DE TOMATE DE 700[kg/h] DE CAPACIDAD”. Elaborado por Christian Fernando Alcalde Cajamarca en el año 2013. Con el planteamiento del *Objetivo General:* “Diseñar una máquina

clasificadora de tomate de 700 [kg/h] de capacidad.", dando las siguientes conclusiones: Del estudio de campo realizado, en la mayoría de sectores en donde se cultiva el tomate, la clasificación se lo realiza de forma manual y artesanal, por lo cual otro objetivo específico de este estudio fue alcanzado, la automatización en el proceso de post cosecha del tomate se logra mediante la utilización de esta máquina en los invernaderos productores. Para escoger el material para el diseño y construcción de los diferentes elementos que componen la máquina, se han tomado en cuenta parámetros como: la funcionalidad y la disponibilidad en el mercado local; el mecanismo que se 168 encuentran en contacto directo con el producto a clasificar se ha seleccionado el mejor material disponible de tal manera que no presente peligro alguno para el consumidor.

Recomendaciones: Durante el proceso de construcción se debe tomar especial precaución en la elaboración de las rieles que soportan los rodillos, se debe cumplir al pie de la letra los procesos establecidos en los planos adjuntos de tal manera que al momento del ensamblaje no sea necesario realizar variaciones en los demás conjuntos de la máquina y funcione con normalidad. Para el montaje de la máquina se debe seguir los siguientes pasos: en la estructura soportante de la máquina se debe realizar una inspección visual de las juntas soldadas, se emperna el motor a la estructura, por otro lado en los tornillos de potencia y el eje de transmisión se colocan los rodamientos respectivos para ensamblarlos al conjunto máquina, los rodillos se les van ensamblando uno a uno en las rieles fijas por medio de los eslabones y asegurándolos con los anillos de seguridad, después se realiza el ensamblaje de las poleas y catalinas en el sistema de transmisión de potencia desde el motor hacia los tornillos de potencia, por último se empernan las tapas de protección, la enrazadera, las bandejas de salida y la tolva de distribución. La superficie en donde se va a situar la máquina debe ser lisa y nivelada, de tal manera que se garantice su estabilidad por su peso propio, además la máquina se debe ubicar cerca de una fuente de energía eléctrica y en un lugar cerrado para evitar oxidación por humedad en las partes mecánicas (Alcalde Cajamarca, 2013).

1.5. OBJETIVO DE LA INVESTIGACIÓN

1.5.1. Objetivo General

Analizar y diseñar un sistema de interpretación de la caracterización colorimétrica y morfométrica de granos de quinua por bloques mediante procesamiento digital de imágenes, para su aplicación en una máquina de selección Óptica de Quinua.

1.5.2. Objetivos Específicos

Diseñar la estructura electrónica y mecánica del subsistema de captura de imágenes, procesamiento de imágenes y transferencia de resultados a la etapa de control usando medios de comunicación estandarizados (HDMI y USB).

a. Analizar y Diseñar el software de control y configuración como interfaz de usuario en la cual se podrán establecer los parámetros de configuración del sistema, monitorizar el estado y los datos de funcionamiento y principalmente la lectura de imágenes y su procesamiento respectivo. Este subsistema debe ser capaz de procesar imágenes a gran velocidad y asignarles una calificación según las características de cada grano en tiempo real y así poder ordenar al subsistema de control la acción predeterminada para cada resultado unitario.

b. Diseñar un modelo integrado de hardware y software a partir de los subsistemas desarrollados en los objetivos anteriores.

CAPITULO II.

MARCO TEÓRICO Y CONCEPTUAL

2.1. MARCO TEÓRICO

2.1.1. Chenopodium Quinoa

La quinua, quínoa o kinwa (Chenopodium quinoa Willd.) es un pseudocereal perteneciente a la subfamilia Chenopodioideae de las amarantáceas. Es un cultivo que se produce en los Andes de Perú, Argentina, Bolivia, Chile, Colombia y Ecuador, y en los Estados Unidos. Bolivia el primer productor mundial, seguido de Perú y de los Estados Unidos. Se la denomina pseudocereal porque no pertenece a la familia de las gramíneas en que están los cereales "tradicionales", pero debido a su alto contenido de almidón su uso es el de un cereal.

El altiplano boliviano, con un área sembrada de 69.000 ha, es el principal cultivador mundial de quinua, la zona con mayor producción de quinua se encuentra en el departamento de Potosí con el 80% del total producido.

El segundo país productor es Perú cuya superficie cultivada asciende a las 55.000 ha, y se producen más de 41.000 t al año el cultivo de quinua es muy importante para los agricultores de este país; principalmente para las más de 70.000 unidades campesinas y pequeños agricultores, de Puno. En Ecuador unas 1700 ha se dedican a la producción de quinua y en Colombia, unas 700 ha, casi todas al sur de Nariño. En las zonas de cultivo de estos cuatro países, es más común encontrar la quinua sembrada en asociación con maíz, frijol y haba o como cercado alrededor de sementeras de papa. (Wikipedia, 2013).

2.1.1.1. Diversidad y variabilidad genética

El Perú por sus características climáticas, suelos, pisos ecológicos, diversidad de especies, zonas de producción, sistemas de producción y ecosistemas productivos y culturas, es considerado entre los 10 países de mayor diversidad del mundo y reconocido como país mega-diverso.

En la región Puno, el cultivo de la quinua en los últimos 10 años adquiere importancia técnica, económica y social. Su producción estandarizada y sostenida debe estar sustentada en la variabilidad genética, producción y abastecimiento de semillas y buenas prácticas agrícolas en la instalación, cosecha y post-cosecha. (Dirección Regional Agraria Puno, 2011)

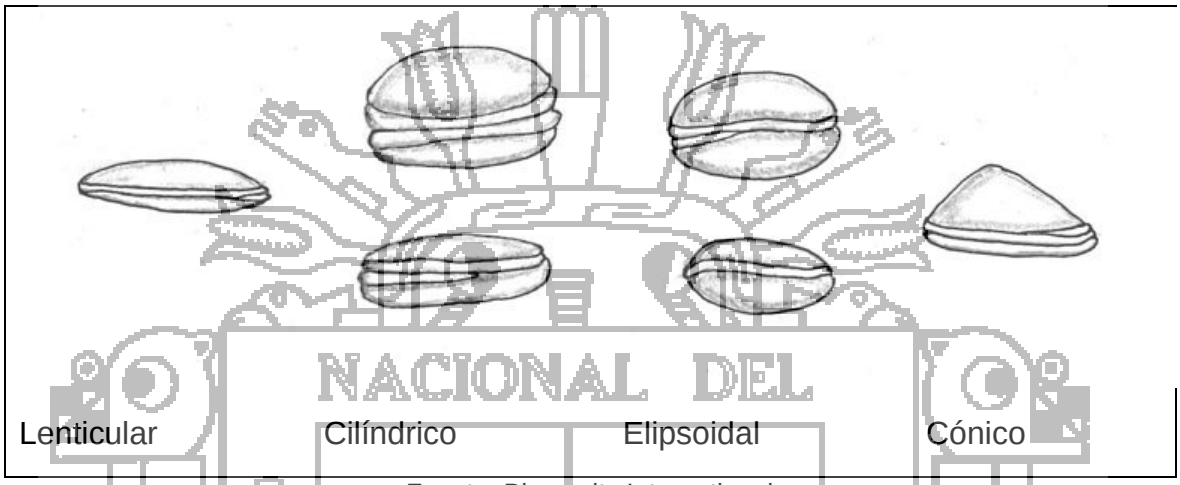
Por ejemplo tenemos las variedades mejor reconocidas:

1. Blancas, jank´o Blanco/blanco Mediana
2. Chullpi o hialinas Blanco/transparente
3. Witullas, Rojo/rojo, púrpura Alta
4. Wariponcho Amarillo/amarillo
5. Kcoito Blanco o plomo/ marrón
6. Pasancallas Rojo, blanco/rojo Alta
7. Cuchi wila Rojo/negro Alta

2.1.1.2. Forma del grano

Dependiendo de la variedad los granos de quinua pueden presentar las siguientes formas:

Ilustración 1: Forma del grano según su variedad



Fuente: Bioversity International

2.1.1.3. Diámetro del grano de por variedad

Promedio del diámetro de grano en los 30 cultivares de quinua en (milímetros):

Tabla 1: Diámetro del grano de por variedad

Variedad	Diámetro (mm)
1 Kancolla	3.71
2 Ratuqui	4.35
3 Utusaya	4.10
4 NL-6	3.85
5 03 -21-79BB	3.70
6 03-08-51	3.50
7 Blanca de Junin	3.60
8 Koyto	3.50
9 Pandela	4.10
10 Ingapirca	3.60
11 Uyuca	4.00
12 Salcedo- INIA	3.75
13 Nariño	3.65
14 Blanca de Juli	3.90
15 Pasankalla	4.40

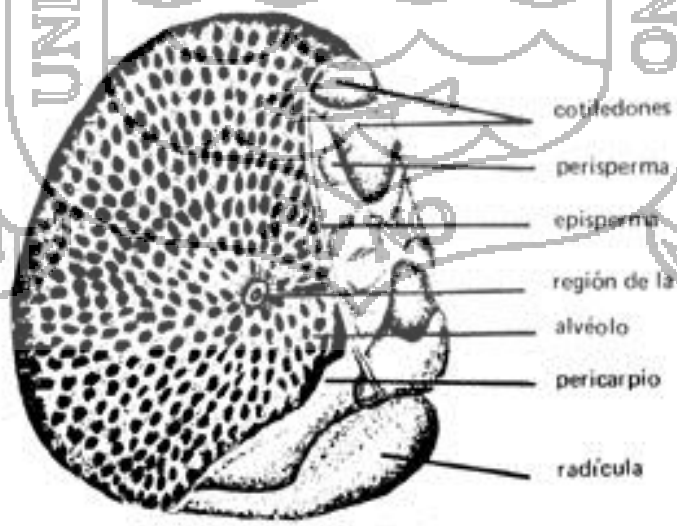
16 Witulla	3.55
17 Cheweca	3.50
18 Chucapaca	3.90
19 Kamiri	4.00
20 Masal 389	3.75
21 ECU- 405	3.75
22 Huariponcho	3.70
23 G-205-95	4.00
24 Amarilla Marangani	3.90
25 Achachino	4.20
26 Ayara	3.50
27 Chullpi	3.85
28 Real	4.00
29 03-21-072RM	3.70
30 Sayaña	4.20

Fuente: Marca Florez, Maritza

2.1.1.4. Variación del color en el pericarpio, episperma y perisperma

Tomaremos como ejemplo los resultados de la Caracterización de Variables Continuas y Discretas del Grano de Quinua (Chenopodium quinoa Willd.) del Banco de Germoplasma de la Estación Experimental Patacamaya.

Ilustración 2: Anatomía del grano de quinua



Fuente: Mario E. Tapia, 2000

Donde según el recuento de las variables, aspecto y color del pericarpio, el color del episperma y la apariencia del perisperma, se logró diferenciar 66 grupos en todo el material ver **Anexo 01**. (Benson Agriculture & Food Instituto)

2.1.2. Procesamiento digital de imágenes

El procesamiento digital de imágenes es el conjunto de técnicas que se aplican a las imágenes digitales con el objetivo de mejorar la calidad o facilitar la búsqueda de información.

Es el conjunto de técnicas englobadas dentro del preprocesamiento de imágenes cuyo objetivo fundamental es obtener, a partir de una imagen origen, otra final cuyo resultado sea más adecuado para una aplicación específica mejorando ciertas características de la misma que posibilite efectuar operaciones del procesado sobre ella.

Los principales objetivos que se persiguen con la aplicación de filtros son:

- Suavizar la imagen: reducir la cantidad de variaciones de intensidad entre píxeles vecinos.
- Eliminar ruido: eliminar aquellos píxeles cuyo nivel de intensidad es muy diferente al de sus vecinos y cuyo origen puede estar tanto en el proceso de adquisición de la imagen como en el de transmisión.
- Realzar bordes: destacar los bordes que se localizan en una imagen.
- Detectar bordes: detectar los píxeles donde se produce un cambio brusco en la función intensidad.

Por tanto, se consideran los filtros como operaciones que se aplican a los píxeles de una imagen digital para optimizarla, enfatizar cierta información o conseguir un efecto especial en ella.

El proceso de filtrado puede llevarse a cabo sobre los dominios de frecuencia y/o espacio. (Wikipedia, 2013)

2.1.3. Reconocimiento de patrones

El reconocimiento de patrones es la ciencia que se ocupa de los procesos sobre ingeniería, computación y matemáticas relacionados con objetos físicos o abstractos, con el propósito de extraer información que permita establecer propiedades de entre conjuntos de dichos objetos

El reconocimiento de patrones también llamado lectura de patrones, identificación de figuras y reconocimiento de formas consiste en el reconocimiento de patrones de señales. Los patrones se obtienen a partir de los procesos de segmentación, extracción de características y descripción dónde cada objeto queda representado por una colección de descriptores. El sistema de reconocimiento debe asignar a cada objeto su categoría o clase (conjunto de entidades que comparten alguna característica que las diferencia del resto). Para poder reconocer los patrones se siguen los siguientes procesos:

- Adquisición de datos
- Extracción de características
- Toma de decisiones

El punto esencial del reconocimiento de patrones es la clasificación: se quiere clasificar una señal dependiendo de sus características. Señales, características y clases pueden ser de cualquiera forma, por ejemplo se puede clasificar imágenes digitales de letras en las clases «A» a «Z» dependiendo de sus píxeles o se puede clasificar ruidos de cantos de los pájaros en clases de órdenes aviares dependiendo de las frecuencias. (Wikipedia, 2013)

2.1.4. Modelos de color

2.1.4.1. El modelo RGB

Éste es un modelo de color basado en lo que se conoce como síntesis aditiva, con lo que es posible representar un color por la mezcla por adición de los tres colores primarios con los que se forma: rojo, verde y azul. Le asignamos un valor a cada uno de los colores primarios para indicar con qué proporción se mezcla cada color. Así, por ejemplo, el valor 0 significa que no interviene en la mezcla, y en la medida

que ese valor aumenta, aportará más intensidad a la mezcla, hasta llegar al valor 255 o 1 si se normaliza. El valor de cada color primario se representa con un byte por lo que su valor puede llegar hasta 255.

Este modelo está basado en el sistema de coordenadas cartesianas. El subespacio de color de interés es el tetraedro mostrado en la figura 2.1, en el cual los valores RGB están en tres vértices; cyan, magenta y amarillo se sitúan en otros tres vértices, el negro corresponde al origen y el blanco en el vértice más alejado del origen.

En este modelo la escala de grises se extiende desde el negro al blanco a lo largo de la diagonal que une esos dos puntos, y los colores son puntos dentro del tetraedro, definidos por vectores desde el origen. (Aristondo Etxeberria, 2010).

2.1.4.2. El modelo YUV

El modelo YUV define un espacio de color en términos de una componente de luminancia (Y) y dos componentes de crominancia (U,V). Codifica una imagen en color teniendo en cuenta la percepción humana, de este modo hace que sea más robusto ante cambios de iluminación. Este modelo es el utilizado en los sistemas de difusión de televisión PAL y NTSC, los cuales son los estándares en la mayoría del mundo.

Es posible obtener los valores YUV de un píxel partiendo de los valores RGB del mismo mediante la utilización de la ecuación:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.515 & -0.100 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Se asume que R, G y B están en el rango 0 a 1, con 0 representando la intensidad mínima y 1 la máxima. Y está en el rango 0 a 1, U está en el rango -0.436 a 0.436 y V está en el rango -0.615 a 0.615.

Éste modelo de color es el empleado en la función recogepelotas desarrollado por Tekniker previamente a este proyecto. Entonces solo se tenía en cuenta el color para discriminar la pelota amarilla en un entorno cerrado de color negro.

2.1.4.3. El modelo HSL

El modelo HSL es un modelo de representaciones considerado "natural", ya que se acerca bastante a la percepción fisiológica del color que tiene el ojo humano. En efecto, el modelo RGB (rojo, verde, azul) puede resultar adecuado para la representación de colores en el equipo o para su presentación en dispositivos de visualización, pero no permite seleccionar los colores con facilidad.

El modelo HSL define el espacio de color con tres características: tono (Hue), saturación (Saturation) y luminosidad (Luminance). Se denomina longitud de onda dominante a la luz que se ve y que corresponde a una determinada cantidad de tono.

La saturación corresponde a la pureza del color. Corresponde a la proporción de la luz pura de longitud de onda dominante y la luz blanca necesaria para definir un color. Un color 100% puro tiene una saturación del 100% y no contiene luz blanca. Colores que se alejen del color puro contienen una mezcla entre la luz blanca y el color puro comprendida entre el 0% y 100%.

La luminosidad es la cantidad de luz que tiene un color determinado. Cualquier valor de tono con una luminosidad del 100% será el color blanco. En cambio si presenta un valor de 0% de luminosidad se tratará del color negro.

2.1.5. UML (Lenguaje de Modelamiento Unificado)

Según Paul Kimmel: “Es un lenguaje estándar que sirve para escribir los planos del software, puede utilizarse para visualizar, especificar, construir y documentar todos los artefactos que componen un sistema con gran cantidad de software.

UML puede usarse para modelar desde sistemas de información hasta aplicaciones distribuidas basadas en la Web, pasando por sistemas empotrados de tiempo real. UML es solamente un lenguaje por lo que solo es una parte de un método de desarrollo de software, es independiente del proceso aunque para que sea óptimo debe usarse en un proceso dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental”. (Stevens, 2002)

2.1.6. OpenCV

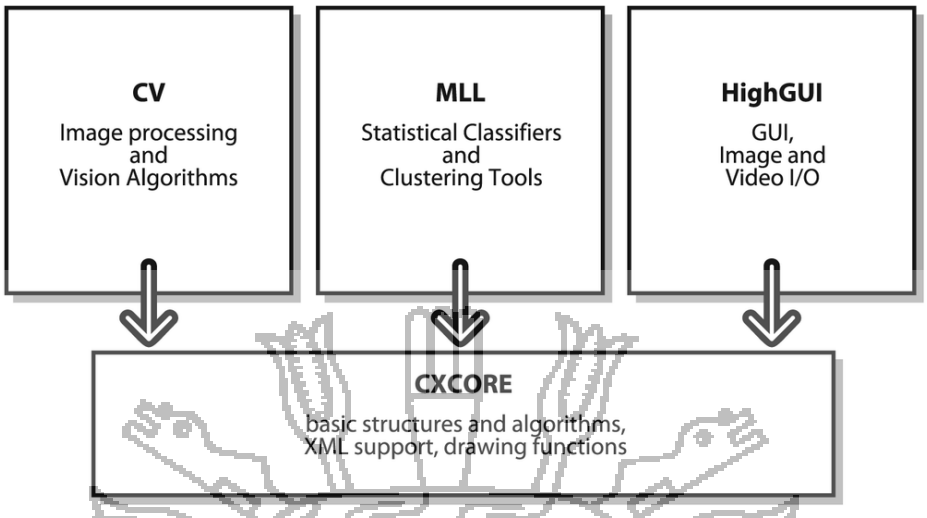
OpenCV es una biblioteca libre de visión artificial originalmente desarrollada por Intel. Desde que apareció su primera versión alfa en el mes de enero de 1999, se ha utilizado en infinidad de aplicaciones. Desde sistemas de seguridad con detección de movimiento, hasta aplicativos de control de procesos donde se requiere reconocimiento de objetos. Esto se debe a que su publicación se da bajo licencia BSD, que permite que sea usada libremente para propósitos comerciales y de investigación con las condiciones en ella expresadas.

Open CV es multiplataforma, existiendo versiones para GNU/Linux, Mac OS X y Windows. Contiene más de 500 funciones que abarcan una gran gama de áreas en el proceso de visión, como reconocimiento de objetos (reconocimiento facial), calibración de cámaras, visión estérea y visión robótica.

El proyecto pretende proporcionar un entorno de desarrollo fácil de utilizar y altamente eficiente. Esto se ha logrado, realizando su programación en código C y C++ optimizados, aprovechando además las capacidades que proveen los procesadores multi núcleo. OpenCV puede además utilizar el sistema de primitivas de rendimiento integradas de Intel, un conjunto de rutinas de bajo nivel. (opencv.org, Wikipedia 2013).

OpenCV se estructura en líneas generales en cinco componentes principales, cuatro de los cuales se muestran en la figura a continuación.

Ilustración 3: Componentes de OpenCV



Fuente: Gary Bradski and Adrian Kaebler

- Componente CV contiene el procesamiento básico de imágenes y algoritmos de visión por ordenador de alto nivel.
- ML es la biblioteca de aprendizaje automático, que incluye muchos clasificadores estadísticos y herramientas de clustering.
- HighGUI contiene rutinas y funciones de E / S para el almacenamiento y el vídeo y las imágenes de carga.
- CXCore contiene las estructuras de datos básicas y contenidos.
- Otro componente es CvAux, que contiene las dos áreas difuntos (incrustado HMM reconocimiento facial) y algoritmos experimentales (fondo / primer plano de segmentación).

2.1.7. Raspberry PI

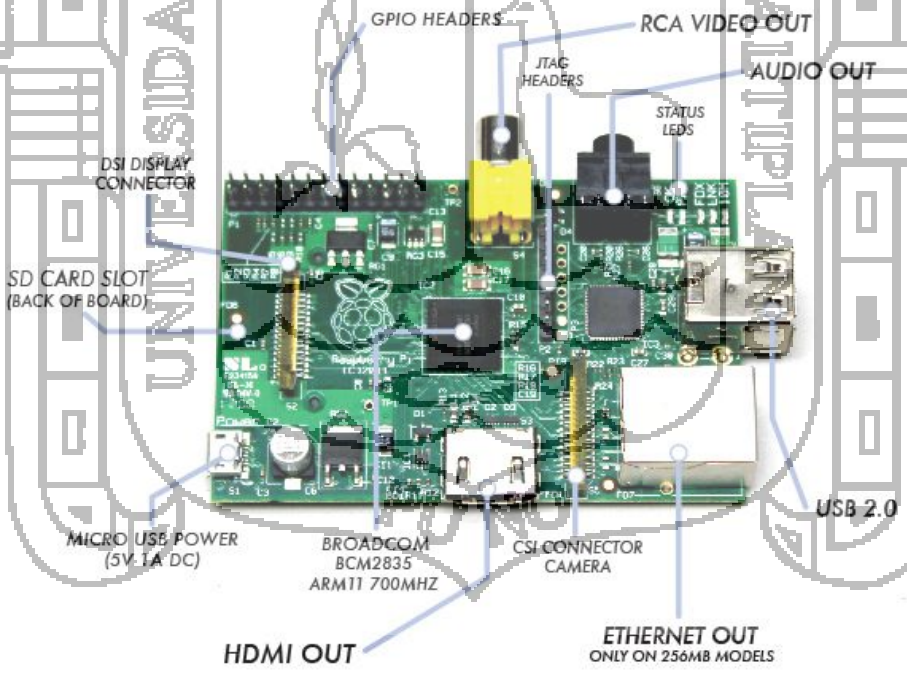
Raspberry Pi es un ordenador de placa reducida o (placa única) (SBC) de bajo costo, desarrollado en Reino Unido por la Fundación Raspberry Pi, con el objetivo de estimular la enseñanza de ciencias de la computación en las escuelas.

El diseño incluye un System-on-a-chip Broadcom BCM2835, que contiene un procesador central (CPU) ARM1176JZF-S a 700 MHz (el firmware incluye unos modos “Turbo” para que el usuario pueda hacerle overclock de hasta 1 GHz sin perder la garantía), un procesador gráfico (GPU) VideoCore IV, y 512

MiB de memoria RAM (aunque originalmente al ser lanzado eran 256 MiB). El diseño no incluye un disco duro ni unidad de estado sólido, ya que usa una tarjeta SD para el almacenamiento permanente; tampoco incluye fuente de alimentación ni carcasa. El 29 de febrero de 2012 la fundación empezó a aceptar órdenes de compra del modelo B, y el 4 de febrero de 2013 del modelo A.

La fundación da soporte para las descargas de las distribuciones para arquitectura ARM, Raspbian (derivada de Debian), RISC OS , Arch Linux ARM (derivado de Arch Linux) y Pidora (derivado de Fedora); y promueve principalmente el aprendizaje del lenguaje de programación Python, y otros lenguajes como Tiny BASIC, C y Perl.

Ilustración 4: Modulo Raspberry Pi Mod. B



Fuente: raspberrypi.org

2.1.7.1. Especificaciones técnicas

Tabla 2: Especificaciones Tecnicas Raspberry PI Mod B

	Modelo A	Modelo B
SoC:5	Broadcom BCM2835 (CPU + GPU + DSP + SDRAM + puerto USB)3	
CPU:	ARM 1176JZF-S a 700 MHz (familia ARM11)3	
GPU:	Broadcom VideoCore IV,59 , OpenGL ES 2.0, MPEG-2 y VC-1 (con licencia),57 1080p30 H.264/MPEG-4 AVC3	
Memoria (SDRAM):	256 MiB (compartidos con la GPU)	512 MiB (compartidos con la GPU)4 desde el 15 de octubre de 2012
Puertos USB 2.0:53	1	2 (vía hub USB integrado)52
Entradas de vídeo:60	Conector MIPI CSI que permite instalar un módulo de cámara desarrollado por la RPF	
Salidas de vídeo:	Conector RCA (PAL y NTSC), HDMI (rev1.3 y 1.4),61 Interfaz DSI para panel LCD62 63	
Salidas de audio:	Conector de 3.5 mm, HDMI	
Almacenamiento integrado:	SD / MMC / ranura para SDIO	
Conectividad de red:	Ninguna	10/100 Ethernet (RJ-45) via hub USB52
Periféricos de bajo nivel:	8 x GPIO, SPI, I²C, UART59	
Reloj en tiempo real:	Ninguno	
Consumo energético:	500 mA, (2.5 W)5	700 mA, (3.5 W)

Fuente de alimentación:	5 V vía Micro USB o GPIO header	
Dimensiones:	85.60mm × 53.98mm64 (3.370 × 2.125 inch)	
Sistemas operativos soportados:	GNU/Linux: Debian (Raspbian), Fedora (Pidora), Arch Linux (Arch Linux ARM), Slackware Linux.	
	RISC OS2	

2.1.7.2. Software

El Raspberry Pi usa mayoritariamente sistemas operativos basados en el núcleo Linux. Raspbian, una distribución derivada de Debian que está optimizada para el hardware de Raspberry Pi, se lanzó durante julio de 2012 y es la distribución recomendada por la fundación para iniciarse.

Slackware ARM (también llamada ARMedslack) versión 13.37 y posteriores arranca sin ninguna modificación.76 77 78 Los 128-496 MiB de memoria RAM disponible en la Raspberry Pi, cubren los necesarios 64 MiB de RAM para arrancar esta distribución en sistemas ARM y i386 sin usar interfaz gráfica (el administrador de ventanas Fluxbox que funciona bajo X Window System requiere 48 MiB de memoria RAM adicional).79 80 Por otro lado, se están creando distribuciones más específicas y ligeras como IPfire (distribución para ser usada como firewall),81 o OpenELEC y Raspbmc (distribuciones con el centro multimedia XBMC).

A la GPU se accede mediante una imagen del firmware de código cerrado, llamado blob binario, que se carga dentro de la GPU al arrancar desde la tarjeta SD. El blob binario está asociado a los drivers Linux que también son de código cerrado. Las aplicaciones hacen llamadas a las librerías de tiempo de ejecución que son de código abierto, y estas librerías hacen llamadas a unos drivers de código abierto en el kernel de Linux.

2.1.7.3. Sistemas operativos soportados

Esta es una lista de sistemas operativos que funcionan, se han portado, o están en proceso de ser portados a Raspberry Pi:

Sistemas operativos completos:

- AROS
- Linux
  - Android97
  - Arch Linux ARM
  - Debian Whezzy Soft-Float, versión de Debian sin soporte para coma flotante por hardware
  - Firefox OS
  - Gentoo Linux98
  - Google Chromium OS
  - Kali Linux
  - Open webOS99
  - PiBang Linux100 , distribución Linux derivada de Raspbian con diferente escritorio y aplicaciones
  - Pidora, versión Fedora Remix optimizada101
  - QtonPi, distribución linux con un framework de aplicaciones multiplataforma basado en Qt framework
  - Raspbian102 , versión de Debian Wheezy para ARMv6 con soporte para coma flotante por hardware
  - Slackware ARM, también conocida como ARMedslack
- Plan 9 from Bell Labs103 104
- RISC OS 52
- Unix
  - FreeBSD105
  - NetBSD106 107
- Distribuciones ligeras multipropósito:
  - Moebius, distribución ligera ARM HF basada en Debian que usa el repositorio de Raspbian y que cabe en una tarjeta SD de 1GB, usa pocos servicios y está optimizada para usar poca memoria
  - Squeezed Arm Puppy, una versión de Puppy Linux (Puppi) para ARMv6 (sap6) específicamente para Raspberry Pi108
  - Minibian, distribución ligera basada en Raspbian.
- Distribuciones ligeras de único propósito:
  - Instant WebKiosk, sistema operativo con solo un navegador

- IPFire
- Micro Elastix, solución de código abierto para comunicaciones unificadas109
- OpenELEC
- Raspbmc
- Xbian

2.1.8. Raspbian

Raspbian es un sistema operativo libre basado en Debian optimizado para el hardware Raspberry Pi. Un sistema operativo es el conjunto de programas básicos y utilidades que hacen que funcione su Raspberry Pi. Sin embargo, Raspbian ofrece más que un SO puro; viene con 35000 paquetes, programas precompilados incluido en un formato agradable para una fácil instalación en el Raspberry Pi. (raspbian.org, 2014)

Raspbian es una distribución del sistema operativo GNU/Linux y por lo tanto libre basado en Debian Wheezy (Debian 7.0) para la placa computadora (SBC) Raspberry Pi, orientado a la enseñanza de informática y la electrónica.

2.2. MARCO CONCEPTUAL

2.2.1. Base de Conocimiento

Es un tipo especial de base de datos para la gestión del conocimiento. Provee los medios para la recolección, organización y recuperación computarizada de conocimiento.

2.2.2. Base de Datos

Conjunto de datos organizados de modo tal que resulte fácil acceder a ellos, gestionarlos y actualizarlos. (Castro, 2011)

2.2.3. Hardware

Corresponde a todas las partes tangibles de una computadora: sus componentes eléctricos, electrónicos, electromecánicos y mecánicos; sus cables, gabinetes o cajas, periféricos de todo tipo y cualquier otro elemento físico involucrado.

2.2.4. Monitorización

Es el proceso por medio del cual, nos aseguramos que nuestro proceder está encaminado adecuada y eficazmente hacia un resultado final, evitando las posibles desviaciones que pudieran presentarse.

2.2.5. Optimizar

Buscar la mejor manera de realizar una actividad, proceso o una tarea.

2.2.6. Proceso

Se denomina proceso al conjunto de acciones o actividades sistematizadas que se realizan o tienen lugar con un fin.

2.2.7. Recursos de Información

Conjunto de elementos físicos y no físicos destinados al uso y manejo de la información para resolver una necesidad o llevar a un buen término el negocio de una empresa.

2.2.8. Software

Se conoce como software al equipamiento lógico o soporte lógico de una computadora digital; comprende el conjunto de los componentes lógicos necesarios que hacen posible la realización de tareas específicas.

2.2.9. Visión Artificial

La visión artificial, también conocida como visión por computador (del inglés computer vision) o visión técnica, es un subcampo de la inteligencia artificial. El propósito de la visión artificial es programar un computador para que "entienda" una escena o las características de una imagen.

2.3. HIPÓTESIS DE LA INVESTIGACIÓN

El análisis y diseño de un sistema computarizado para la interpretación de características colorimétricas y morfométricas de granos de quinua permitirá la implementación de sistemas automáticos de control de calidad y de selección en el proceso de producción de Quinua.

2.4. OPERACIONALIZACIÓN DE VARIABLES

Tabla 3: Operacionalización de variables

VARIABLES	DIMENSIONES	INDICADORES	ESCALA
VARIABLE INDEPENDIENTE: Sistema de interpretación colorimétrica y morfométrica de granos de quinua.	Interpretación de característica colorimétrica y morfométrica	Color Brillo	Porcentaje de confiabilidad
		Forma Diámetro	Porcentaje de confiabilidad
	Velocidad de procesamiento	Cantidad de granos por minuto	Numérica
	Eficiencia	Recursos de Hardware y Software	- Alto - Medio - Bajo
		Tiempo de Implementación	Tiempo semanas
	Accesibilidad	Complejidad de Adaptación	- Alto - Medio - Bajo
VARIABLE DEPENDIENTE: Sistemas automáticos de control de calidad y de selección en el proceso de producción de Quinua	Eficiencia	Velocidad de Trabajo	Granos / Segundo
		Volumen Resultado x Hora	- Kilogramos de quinua
		Precisión de resultados	Porcentual
	Accesibilidad	Costo de Implementación	- Alto - Medio - Bajo

Fuente: Elaboración propia

2.5. LIMITACIONES DE LA INVESTIGACIÓN

En esta investigación se conocen los siguientes alcances y limitaciones operativas:

- Se aplicara en el departamento de Puno.
- El estudio se contextualizara en el último cuatrimestre del 2013
- El proyecto solo será el análisis y diseño del sistema computacional para la interpretación colorimétrica y morfométrica de los granos de quinua.

CAPITULO III.

MÉTODO DE INVESTIGACIÓN

3.1. METODOLOGÍA DE LA INVESTIGACIÓN

En la presente investigación se utilizó el tipo de investigación analítico-experimental y para el proceso de desarrollo se utilizó el modelo de desarrollo RUP (Rational Unified Process) teniendo como soporte los lenguajes de modelado UML(Unified Modeling Language) y SysML(System Modeling Language).

3.2. MÉTODO DE RECOPILACIÓN DE DATOS

La principal fuente de recopilación de datos que se utilizará para la demostración de la hipótesis se realizará mediante pruebas experimentales, de tal manera que con la medición de cada dimensión propuesta podamos obtener el nivel de cumplimiento de la hipótesis.

3.3. ÁMBITO DEL ESTUDIO

El ámbito de estudio para la investigación está fijado en Puno pues esta investigación nace gracias al Proyecto FINCyT (Fondo Para la Innovación, Ciencia y Tecnología), localizado en:

Departamento : Puno

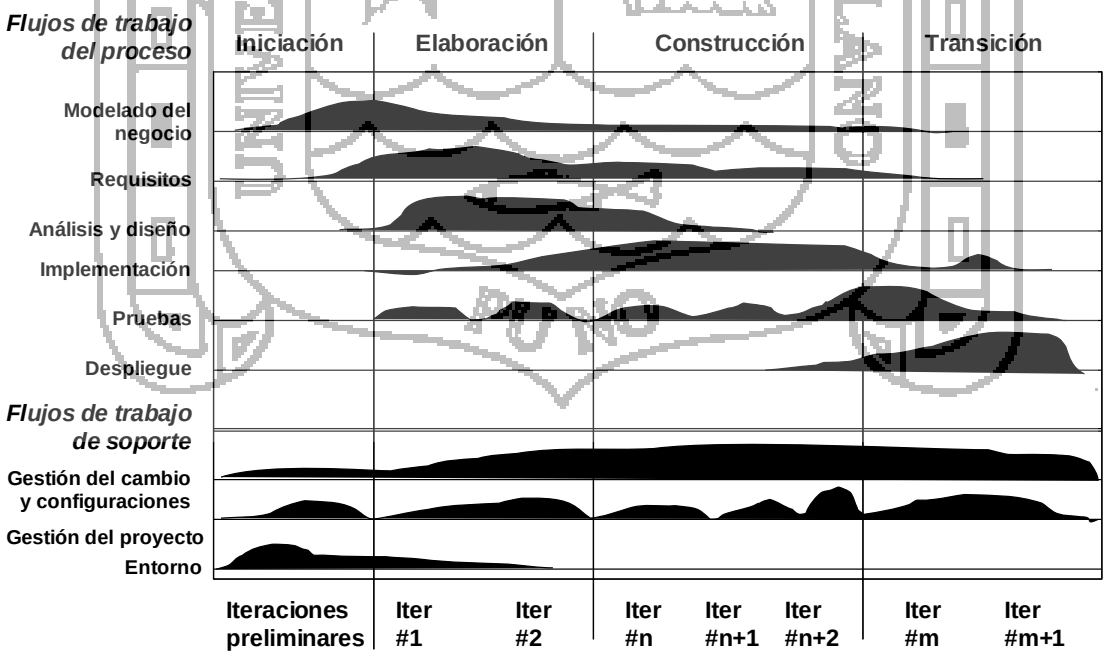
Provincia : Puno

Distrito : Puno

3.4. METODOLOGÍA DE DESARROLLO DEL SISTEMA

Modelo de desarrollo del Software: Para el desarrollo del sistema se aplicó el modelo Rational Unified Process (RUP) teniendo como soporte el lenguaje de modelado UML (Unified Modeling Language) Y SysML (System Modeling Language).

Ilustración 5: Modelo de desarrollo del Software



Fuente: (Alonso Amo, Martínez Normand, & Segovia Pérez, 2005)

3.5. ADMINISTRACIÓN DEL PROYECTO

3.5.1. Cronograma de Ejecución

Tabla 4: Cronograma de Ejecución

	ACTIVIDADES	SEPTIEMBRE				OCTUBRE				NOVIEMBRE				DICIEMBRE				ENERO	
		Semana				Semana				Semana				Semana					
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2
1	Elaboración del Plan de tesis	X	X																
2	Estudio de las variedades de quinua			X	X														
3	Estudio de algoritmos de procesamiento de imágenes y reconocimiento de patrones					X	X												
4	Diseño del sistema de control de calidad							X	X	X									
5	Implementación del software									X	X	X	X						
6	Realización de pruebas													X					
7	Elaboración del Informe final de la tesis				X	X	X	X	X	X	X	X	X	X	X	X			
8	Corrección al informe final de la tesis						X	X	X	X	X	X	X	X	X	X	X		
9	Presentación de la tesis																	X	
10	Sustentaciones																		X

3.5.2. Requerimiento de equipos, materiales y servicios para el desarrollo

3.5.2.1. Hardware

Mínimamente PC con procesador Intel Core I5, 4Gb de RAM, 1Gb de Video

- Placa de adquisición de video analógico y digital
- Unidades de almacenamiento USB, CD`S
- Cámara de video con comunicación con función de mando a distancia
- Impresora

3.5.2.2. Software

- Sistema Operativo Windows 7
- Visual Studio 2010 (Visual C++)
- SQL Server (Opcional) o MySQL
- Servidor Apache, PHP
- Herramientas para diseño de diagramas UML

3.5.2.3. Herramientas de servicio

- Procesador de texto
- Herramienta de planificación
- Herramientas de diseño de diagramas

3.5.2.4. Materiales de escritorio

- Papel continuo
- Papel A4 80g.
- Otros.

3.5.2.5. Servicios

- Energía Eléctrica.
- Acceso a Internet

3.5.3. Requerimientos de equipos y materiales para implantación

3.5.3.1. Hardware

- PC Intel Core i7-3970X @ 3.50GHz, Placa compatible, 4GB de tarjeta de video, 8Gb de RAM, Tarjeta de adquisición de video.
- Cámara fotográfica de alta velocidad, con comunicación a PC, SDK compatible con C++.
- Cable USB y HDMI.

3.5.3.2. Software

- SO Windows 7
- Drivers de cámara Fotográfica

3.5.3.3. Servicios

El lugar debe contar con servicio de Energía eléctrica

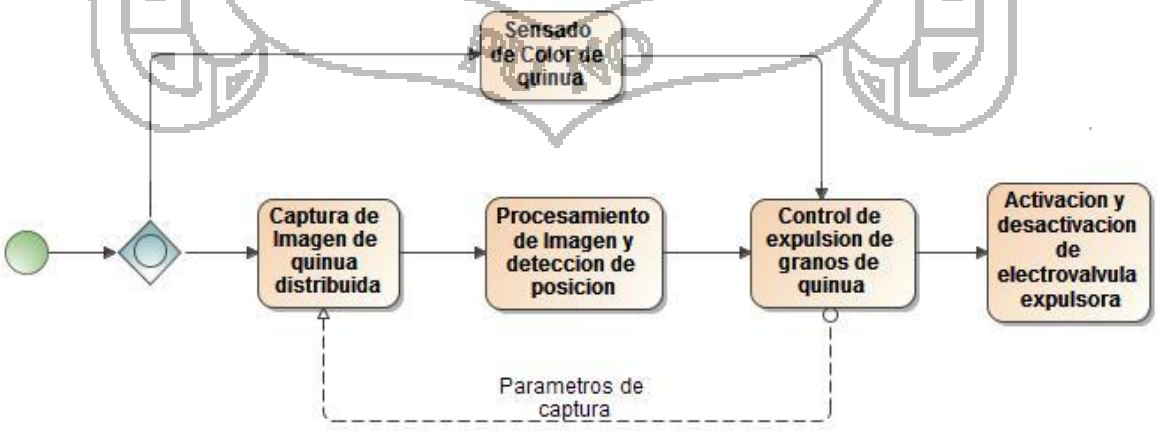
CAPITULO IV.

EXPOSICIÓN Y ANÁLISIS DE LOS RESULTADOS

4.1. ESTRUCTURA GENERAL DEL SISTEMA

A continuación se presentan los elementos que intervienen en el sistema de detección y procesamiento de granos de quinua y su flujo de proceso para su automatización:

Ilustración 6: Estructura General del sistema



Fuente: Elaboración Propia.

Para lo cual se tiene el siguiente flujo de procesos:

a) Captura de Video sin compresión de 1920x1080px de resolución a 60fps desde la cámara DSRL.
b) El video es transferido a la CPU por HDMI y almacenado en el buffer de video temporal.
c) El video en el buffer es captado como objetos de imagen en memoria mediante las librerías del SDK de la tarjeta de adquisición de video.
d) Las imágenes en memoria son filtradas y segmentadas para el análisis comparativo por segmento.
e) Los segmentos seleccionados se estructuran en una matriz de resultados donde se especifica mediante valores de tiempo el momento exacto a ser expulsados, esta matriz se transfiere a las unidades de control.
f) Cada placa de control interpretara los resultados preliminares de la CPU, y validará el resultado mediante el censado de su cámara y un procesamiento adicional de validación.

4.1.1. Estructura de hardware

Teniendo en cuenta la velocidad requerida de selección de granos de quinua, se utilizara como sensor de color una cámara digital la cual pueda transferir fotogramas a 30 o 60 veces por segundo, además una tarjeta capaz de adquirir el video a esa misma velocidad y almacenarlo en memoria a medida que se recibe.

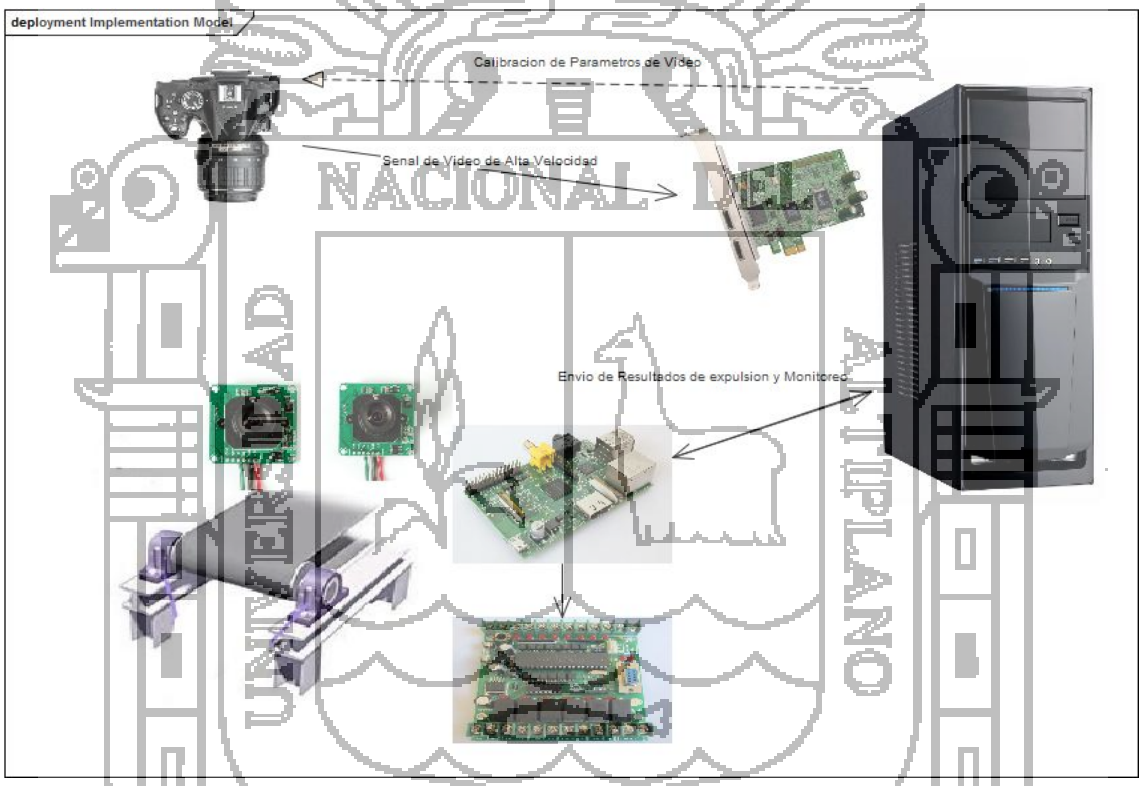
Por otro lado una vez obtenido la imagen digitalizada de granos de quinua deberá ser procesada por una CPU aplicando los algoritmos necesarios para la detección de posiciones, formas y colores de cada grano y según sus posiciones enviar los datos necesarios a la placa de control de la máquina para identificar los granos a expulsar según posición y tiempo.

Y finalmente según los datos proveídos por la unidad de procesamiento la placa se encargara de ordenar la expulsión o succión de los granos seleccionados.

Todas las transferencias de datos deberán ser realizadas con la mejor velocidad posible para lo cual se utilizaran conectores HDMI, USB o Cable de Red según sea necesario.

Para mejorar la calidad de selección se integrara además dos módulos Raspberry pi con su respectiva cámara, los cuales podrán validar la información proporcionada por la unidad de procesamiento, realizando análisis puntuales más detallados desde con los datos ya obtenidos. La Ilustración 4, nos muestra el diagrama general resultante.

Ilustración 7: Diagrama Estructural General Final



Fuente: Elaboración Propia

Una vez definido la estructura general se procederá a definir las características de cada elemento teniendo en cuenta la capacidad de procesamiento y transferencia de datos requeridos.

4.1.2. Componentes de Hardware

4.1.2.1. Cámara DSRL

El sistema a desarrollar requiere la adquisición de imágenes en tiempo real, por lo es necesario una cámara digital capaz de transferir 30 o más imágenes FullHD (1920x1080px de resolución) por segundo ó 50 o más imágenes HD (1280x720px de resolución) sin compresión (sin pérdida de calidad), puesto que la visión de esta cámara será de todo el área de censado cubriendo las 24 hileras de quinua en la faja trasportadora.

Además esta transferencia del conjunto de imágenes o video se realizará mediante HDMI para mantener compatibilidad con los dispositivos actuales.

La cámara deberá contar con un sistema de control desde la PC y las librerías necesarias para la personalización de comandos, por lo que es necesario contar con un SDK implementado por el fabricante.

A continuación se detallan los requerimientos mínimos de la cámara a utilizar.

- HDMI HD video output.
- HDMI out with support of uncompressed video (clean HDMI)
- Up to 1080p at 24, 25 and 50i, 30 and 60i, 720p at 50 or 60 frames per second.
- Live View shooting.
    - Software development kits (SDK).

De acuerdo a las especificaciones se eligió la Camara DSRL Nikon D5200, presentamos sus características en el Anexo 05.

4.1.2.2. Tarjeta de adquisición de Video

El sistema requiere una tarjeta de adquisición de imágenes o video que soporte la carga de transferencia de la cámara requerida, es decir deberá contar con una entrada HDMI la cual pueda recibir video sin compresión en FullHD a 30 frames por segundo o más.

La interfaz de la tarjeta deberá ser PCI Express para garantizar la velocidad requerida.

Además es necesario que cuente con un SDK con las librerías necesarias para la lectura de video almacenado en memoria (Buffered Video) para realizar el procesamiento de imágenes sin necesidad de que esta fuera almacenada en el disco duro.

A continuación se detallan los requerimientos mínimos para la tarjeta.

- Grabación de Video desde HDMI
- Captura en HD1080i
- Captura de video sin compresión
- Soporte para video con resolución 1920x1080@50i/60i
- Compatibilidad con Windows 7 /8 y Linux
- SDK con soporte para el lenguaje de Programación Visual C++ o C#
- PCIe1X

Según los requisitos del dispositivo, se eligió la Tarjeta Capturadora Avermedia Darkcrystal Hd Capture Sdk Hdmi C727 y sus características se presentan en el Anexo 04.

4.1.2.3. CPU

El sistema requiere también un procesador de imágenes de alta velocidad para lo cual se necesitará la adquisición de una CPU con la capacidad necesaria de procesamiento para realizar el trabajo de adquisición de imágenes, procesamiento (aplicación de filtros, separación de áreas, comparación de áreas o muestras, clasificación) y el envío de resultado al micro controlador mediante USB en el menor tiempo posible 3 a 20 milisegundos por captura.

Además la tarjeta de adquisición de video requiere la utilización de los microprocesadores más actuales para un buen desempeño según los *requerimientos mínimos del sistema* según los fabricantes de este tipo de tarjetas.

A continuación se detallan los requerimientos mínimos de la CPU a utilizar.

- Procesador Intel Corei7 de 4ta generación o el equivalente en AMD.
- Memoria RAM de 8GB DDR3 1600/1333 MHz.
- Interfaz PCI Express*3.0 (ó 2.0) slot.

- Memoria Video 2 GB DDR3/DDR5 de 128-bit/256-bit DirectX 11, OpenGL 4.2.

4.1.2.4. Módulos Raspberry PI con Cámara

Debido a la alta velocidad de transferencia de datos que se requiere para la entrega de resultados de procesamiento desde la CPU a la tarjeta de control de la maquina se vio por conveniente utilizar un módulo Raspberry Pi para la conexión punto a punto con la PC mediante el estándar Ethernet y el protocolo UDP.

Adicionalmente el módulo Raspberry se encargará de distribuir los resultados recibidos a los módulos de control de la etapa de expulsión.

Además se tiene previsto utilizar el módulo Raspberry para la validación del resultado de la unidad de procesamiento para lo cual se dispondrá de dos módulos Raspberry Pi con su respectiva cámara cada una las cuales estarán encargadas de analizar los granos a ser expulsados de acuerdo a los resultados iniciales, de tal forma que se obtendrá una mejor sincronización para la expulsión de los granos en movimiento los cuales tendrán diferentes posiciones en el espacio y el tiempo, y de lo cual se obtendrán resultados de expulsión más precisos y fiables.

Estas 2 cámaras estarán dispuestos cubriendo la mitad del área de censado equivalente a 12 hileras de quinua en paralelo.

4.1.3. Funcionamiento del sistema

A continuación se describen las interacciones de cada módulo del sistema:

4.1.3.1. Censado y Procesamiento

a) Captura de Video sin compresión de 1920x1080px de resolución a 60fps desde la cámara DSRL.
b) El video es transferido a la CPU por HDMI y almacenado en el buffer de video temporal.
c) El video en el buffer es desglosado en frames y procesado por la CPU en tiempo real.
d) La CPU genera una matriz de resultados donde se especifica mediante valores de tiempo el momento exacto a ser expulsados, esta matriz se transfiere a las unidades de control representados por las Placas Raspberry PI.
e) Cada placa de control Raspberry PI interpretara los resultados preliminares de la CPU, y validará el resultado mediante el censado de su cámara y un procesamiento adicional de validación.

4.1.3.2. Configuración

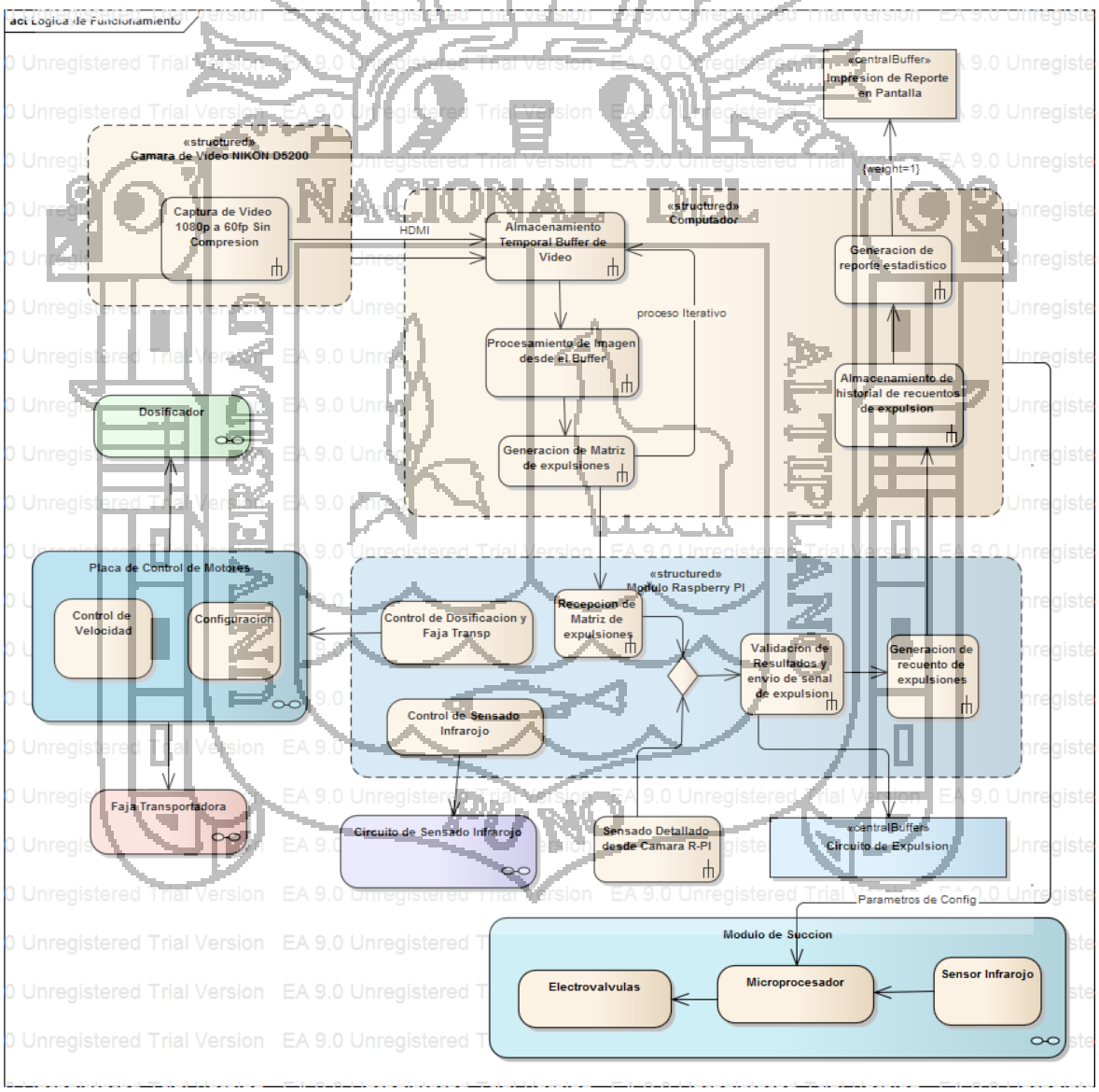
a) El usuario mediante la interfaz del computador establece las configuraciones siguientes:
   - Color de expulsión.
   - Margen de error de color.
   - Calibración de valores mínimos y máximos del Conversor Análogo Digital de la placa de control.
   - Configuración de valores de resolución y velocidad de video de la cámara.
b) Los datos establecidos por el usuario son transferidos a la Placa Raspbery PI mediante Cable de red vía Socket TCP y a la cámara mediante cable USB.
c) La placa Raspberry PI o la Cámara DSRL aplicarán los cambios necesarios según la configuración seleccionada.

4.1.3.3. Monitoreo

a) La placa Raspberry pi generará datos de recuento y estado, para enviarlos a la PC cada cierto tiempo.
b) La PC Mostrara estos valores en el monitor mediante una interfaz de usuario.

A continuación se presenta el diagrama general del proceso de funcionamiento del subsistema de procesamiento de imágenes en la Ilustración 8.

Ilustración 8: Proceso de funcionamiento del subsistema de procesamiento de imágenes



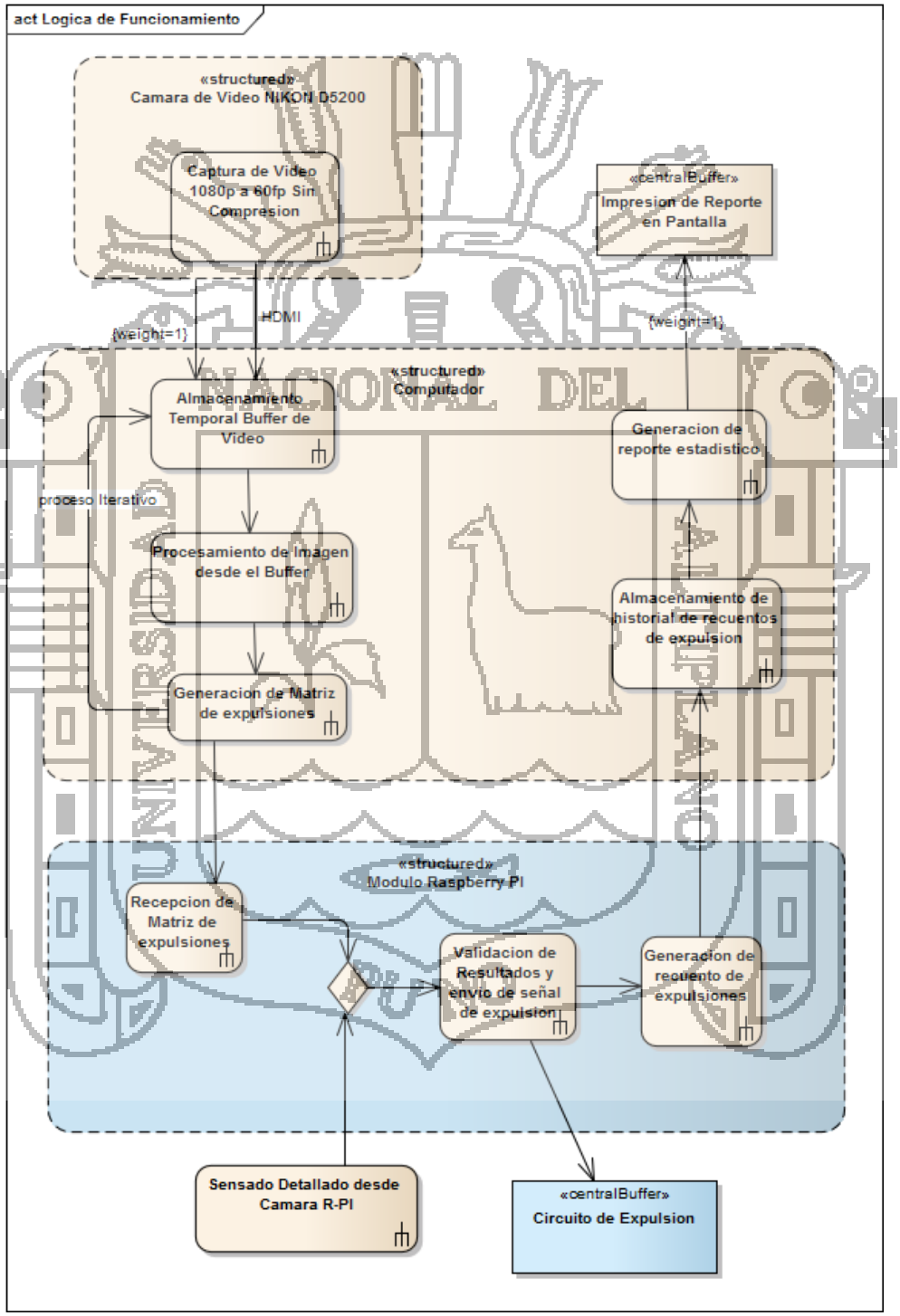
Fuente: Elaboración Propia

4.2. SUBSISTEMA DE PROCESAMIENTO DE IMÁGENES

4.2.1. Modelado del negocio

A continuación se muestra el diagrama de actividades del proceso general:

Ilustración 9: Diagrama de actividades del proceso general.



Fuente: Elaboración Propia

4.2.2. Requerimientos – Modelos de Casos de Uso

A continuación se definirán los requerimientos del software de captura, procesamiento de imágenes, aplicación de configuración de módulos y monitoreo de datos.

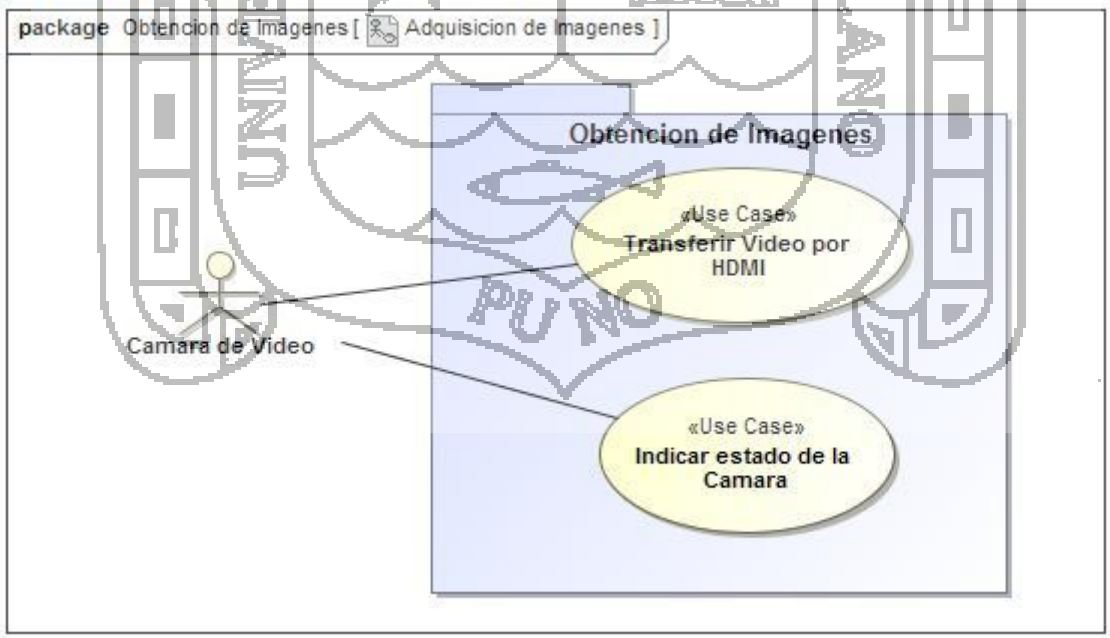
4.2.2.1. Actores

Ilustración 10: Diagrama de Actores

Fuente: Elaboración Propia

4.2.2.2. Adquisición de Imágenes

Ilustración 11: Adquisición de Imágenes



Fuente: Elaboración Propia

Tabla 5: Caso de uso - Indicar el estado de la maquina

Nombre Caso de Uso:	Indicar estado de la Cámara	ID:	
Actor Primario:	• Cámara de Video		
Relaciones			
Asociación:	• Actor Cámara de Video		
Narrativa			
Pre Condición:	La aplicación esta iniciada y la cámara están configuradas y en funcionamiento.		
Post Condición:	La información del estado de la cámara se muestra en la interfaz de la aplicación.		
Escenarios			
Flujo Básico de Eventos:	1. La aplicación envía una señal periódicamente a la cámara. 2. La cámara devuelve el estado actual de batería y parámetros de configuración. 3. La aplicación muestra en la interfaz los valores actuales recibidos.		
Flujo Alternativo de Eventos:			

Tabla 6: Caso de uso - Transferir video por HDMI

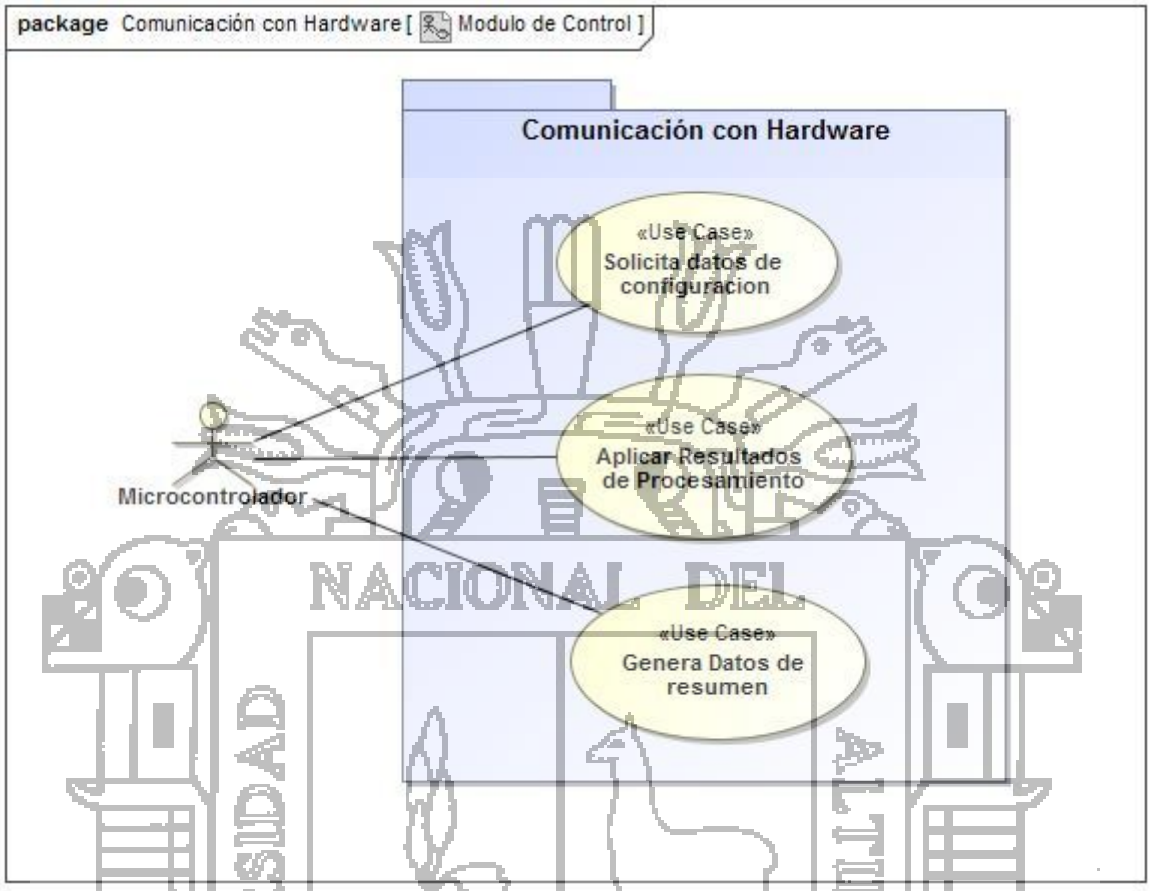
Nombre Caso de Uso:	Transferir Video por HDMI	ID:	
Actor Primario:	• Cámara de Video		

Relaciones	
Asociación:	• Actor Cámara de Video
Narrativa	
Pre Condición:	La cámara está conectada y encendida.
Post Condición:	El video es almacenado temporalmente en un buffer de video.
Escenarios	
Flujo Básico de Eventos:	1. La cámara realiza la petición del parámetro de velocidad. 2. La aplicación establece la configuración de los parámetros de captura de video establecidos por el Operador vía USB. 3. La cámara cada frame de video capturado a la tarjeta de adquisición.
Flujo Alternativo de Eventos:	

Fuente: Elaboración Propia

4.2.2.3. Módulo de Control

Ilustración 12: Módulo de Control



Fuente: Elaboración Propia

Tabla 7: Caso de uso – Aplicar resultados de procedimiento.

Nombre Caso de Uso:	Aplicar Resultados de Procesamiento	ID:
Actor Primario:	• Microcontrolador	
Relaciones		
Asociación:	• Actor Microcontrolador	

Narrativa	
Pre Condición:	El sistema recupera y procesa la imagen enviada por la cámara.
Post Condición:	El microcontrolador activa la electroválvula según el resultado recogido.
Escenarios	
Flujo Básico de Eventos:	1. El sistema recupera y procesa una imagen desde el buffer de entrada. 2. El sistema genera una matriz de resultados tomando en cuenta la posición de cada grano y la velocidad de la faja transportadora. 3. El sistema transfiere la matriz de resultados al microcontrolador por USB. 4. El microcontrolador interpreta los resultados y activa si es necesario la electroválvula en el tiempo preciso calculado.
Flujo Alternativo de Eventos:	

Fuente: Elaboración Propia

Tabla 8: Caso de uso - Genera datos de resumen

Nombre Caso de Uso:	Genera Datos de resumen	ID:	
Actor Primario:	• Microcontrolador		

Relaciones	
Asociación:	• Actor Microcontrolador
Narrativa	
Pre Condición:	La máquina se encuentra en funcionamiento.
Post Condición:	Se muestra un reporte estadístico en la pantalla de la aplicación.
Escenarios	
Flujo Básico de Eventos:	1. El microcontrolador almacena el recuento de granos expulsados y la cantidad clasificada por tamaño de grano. 2. El microcontrolador envía los valores almacenados periódicamente a la aplicación en la PC. 3. El sistema interpreta los valores y los almacena en una base de datos local. 4. El sistema muestra en la pantalla los valores estadísticos.
Flujo Alternativo de Eventos:	

Tabla 9: Caso de uso - Solicita datos de configuracin

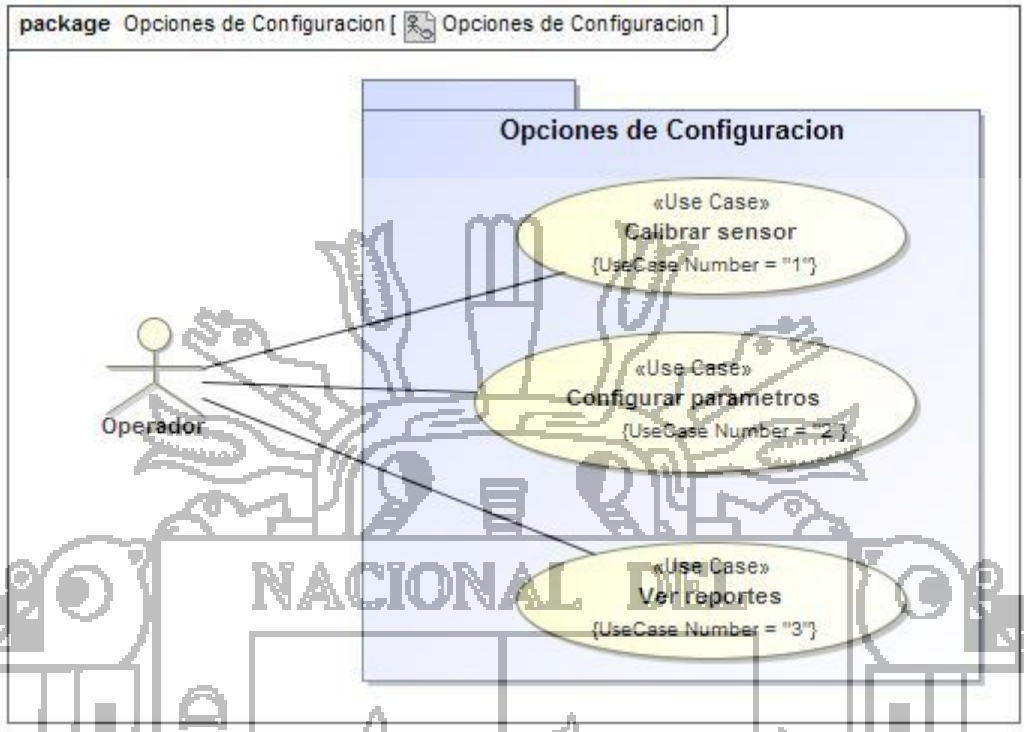
Nombre Caso de Uso:	Solicita datos de configuración	ID:	
Actor Primario:	• Microcontrolador		

Relaciones	
Asociación:	• Actor Microcontrolador
Narrativa	
Pre Condición:	La aplicación esta iniciada, la maquina encendida
Post Condición:	El microcontrolador aplica los parámetros establecidos.
Escenarios	
Flujo Básico de Eventos:	1. El microcontrolador envía una petición de parámetros de configuración vía USB. 2. El sistema recupera los datos de configuración almacenados. 3. El sistema responde la petición del microcontrolador con los valores recuperados. 4. El microcontrolador aplica los cambios al sistema mecánico-electrónico.
Flujo Alternativo de Eventos:	

Fuente: Elaboración Propia

4.2.2.4. Opciones de Configuración

Ilustración 13: Opciones de Configuración



Fuente: Elaboración Propia

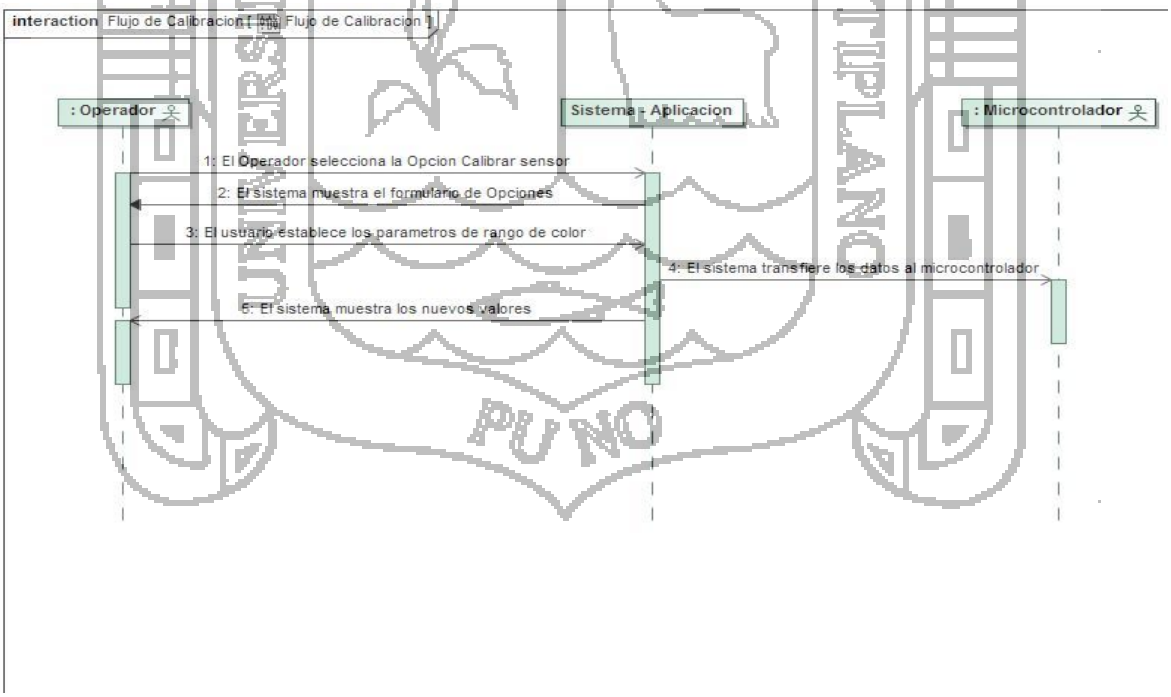
Tabla 10: Caso de uso - Calibrar sensor

Nombre Caso de Uso:	Calibrar sensor	ID:	1
Actor Primario:	• Operador		
Relaciones			
Asociación:	• Actor Operador		
Narrativa			
Pre Condición:	Haber iniciado la aplicación		
Post Condición:	Los parámetros de configuración son transferidos al microcontrolador		

Escenarios	
Flujo Básico de Eventos:	1. El usuario selecciona la opción calibrar sensor. 2. El sistema muestra el formulario de parámetros 3. El usuario establece los parámetros de rango de color 4. El sistema envía los parámetros establecidos al microcontrolador vía USB. 5. El sistema muestra los nuevos parámetros establecidos para el control del sensor.
Flujo Alternativo de Eventos:	

Fuente: Elaboración Propia

Ilustración 14: Diagrama de Interacción – caso de Uso Opciones de Configuración



Fuente: Elaboración Propia

Tabla 11: Caso de uso - Configurar parámetros

Nombre Caso de Uso:	Configurar parámetros	ID:	2
Actor Primario:	• Operador		
Relaciones			
Asociación:	• Actor Operador		
Narrativa			
Pre Condición:	Iniciar la aplicación		
Post Condición:	Los parámetros de velocidad, color de granos a expulsar y parámetros adicionales, serán establecidos.		
Escenarios			
Flujo Básico de Eventos:	1. El usuario Selecciona la opción establecer parámetros de trabajo. 2. La aplicación muestra el formulario de configuración. 3. El usuario ingresa los campos de velocidad, selecciona el color de quinua a expulsar y velocidad de muestreo de video. 4. El sistema transfiere los valores al microcontrolador. 5. El sistema muestra los datos aceptados.		
Flujo Alternativo de Eventos:	#3.1. El sistema muestra alertas de validación si los valores ingresados no son correctos o exceden los límites permitidos.		

Fuente: Elaboración Propia

Tabla 12: Caso de uso - Ver reportes de procesamiento

Nombre Caso de Uso:	Ver reportes de procesamiento	ID:	3
Actor Primario:	• Operador		
Relaciones			
Asociación:	• Actor Operador		
Narrativa			
Pre Condición:	El usuario se encuentra en la interfaz inicial de usuario		
Post Condición:	El usuario observa la información del recuento de procesamiento		
Escenarios			
Flujo Básico de Eventos:	El usuario inicial el proceso de selección. El usuario observa la información en tiempo real sobre el recuento de granos procesados y seleccionados.		
Flujo Alternativo de Eventos:			

Fuente: Elaboración Propia

4.2.3. Análisis del Procesamiento de Imágenes.

4.2.3.1. Filtrado de la Imagen

La siguiente imagen es una fotografía tomada de un conjunto de quinuas de diferentes colores y tamaños. Para poder explicar el análisis necesariamente debemos considerar lo siguiente:

Ilustración 14: Fotografía de una muestra de granos de quinua

Fuente: Elaboración Propia FINCyT

Dentro del procesamiento de la imagen anterior la cual consideramos como imagen origen (30x18cm, a 1920x1080px), necesitamos otra cuya sea más fácil su tratamiento, es decir la parte del filtrado será un subproceso para mejorar las características que nos posibilite efectuar las operaciones de ver su color, forma y su tamaño.

Los objetivos que perseguimos con el filtrado de la imagen origen son los siguientes:

- **Suavizar la Imagen**: Reduciremos las cantidades de variaciones de intensidad entre pixeles vecinos. Veamos en la siguiente figura:

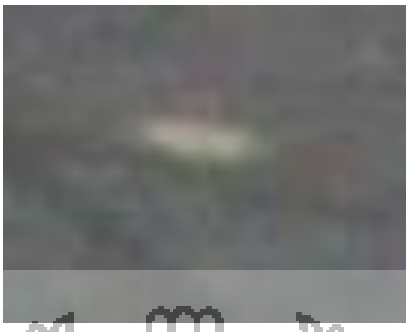
Ilustración 15: Imagen de una Quinua sin modificar

Fuente: Elaboración Propia FINCyT

Vemos una solo quinua la cual no tiene un borde definido solo pareciera haber una diferencia entre el fondo y exterior, pero mas no así en el contorno que os separa.

- **Eliminar Ruido:** Eliminar los pixeles e imágenes completas cuyo nivel de intensidad forma en conjunto es muy diferente al de sus vecinos y cuyo origen pueden estar tanto en proceso de la toma de la imagen, de un objeto que no debió de participar (como un cabello) o simplemente al momento de transmitir la imagen. Vemos en la imagen a continuación como una simple mancha puede ser confundida como una quinua pero sin embargo por el tamaño, la forma de su borde y forma, debe ser descartado posteriormente.

Ilustración 16: Imagen de un ruido que no representa una quinua



Fuente: Elaboración Propia FINCyT

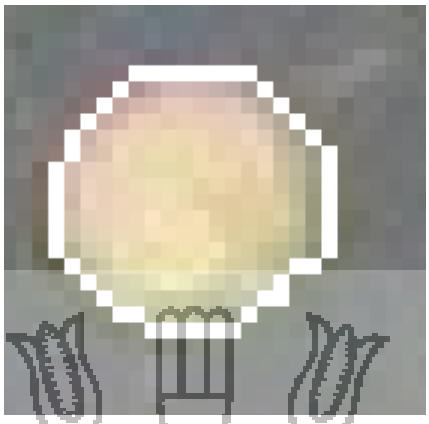
- **Realzar los Bordes**. Una vez vistos los dos inconvenientes anteriores, es necesario definir los contornos que son los bordes que separaran la imagen del fondo para luego poder determinar su tamaño y su forma, así una imagen sin borde seria de la siguiente forma.
-

Ilustración 17: Imagen de una Quinua mostrando sus Pixeles

Fuente: Elaboración Propia FINCyT

Y una vez realizado el proceso de realce de los bordes quedara de la siguiente forma la imagen anterior.

Ilustración 18: Imagen de una Quinua con los bordes realzados



Fuente: Elaboración Propia FINCyT

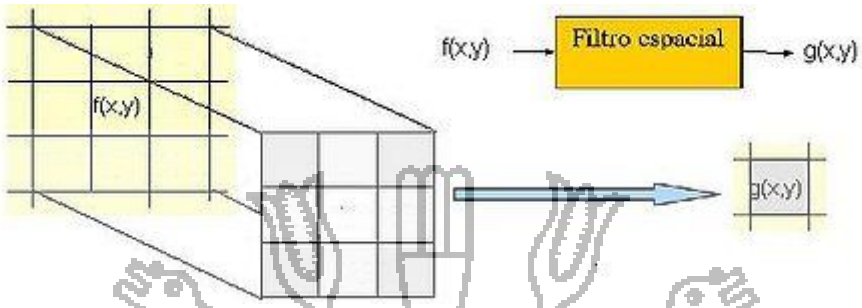
Es necesario destacar para la selección del color borde se ha utilizado el color blanco, debido a que el color del fondo tiene a ser negro, por contraste el color de la quinua tiende a ser claro, por eso la elección del color blanco.

El Filtrado se realizara mediante la técnica siguiente:

Filtrado en el dominio del espacio: Las operaciones tal como se vio en las imágenes anteriores se realizaran directamente sobre los pixeles de la imagen. En este proceso se relaciona para todos y cada uno delos puntos de la imagen, un conjunto de pixeles próximos al pixel objetivo para así obtener una información más útil, el filtro de dominio del a utilizarse por la característica de la imagen y por la característica de tratar de delimitar su contorno será un Filtro Lineal basado en un Kernel o mascara de convolución.

El concepto de kernel se entiende como una matriz de coeficientes donde el entorno del punto (x,y) que se considera en la imagen para obtener g(x,y) está determinado por el tamaño y forma del kernel seleccionado. Aunque la forma y tamaño de esta matriz es variable y queda a elección de cada usuario, es común el uso de kernelscuadrados nxn. Dependiendo de la implementación, en los límites de la imagen se aplica un tratamiento especial (se asume un marco exterior de ceros o se repiten los valores del borde) o no se aplica ninguno. Es por ello, que el tipo de filtrado queda establecido por el contenido de dicho kernel utilizado.

Ilustración 19: Explicación de un Método Kernel.



Fuente: (Wikipedia, 2013)

Forma de Aplicación del Kernel.

Para realizar un filtrado en el dominio del espacio se realiza una convolución (barrido) del kernel sobre la imagen. Para ello se sigue el Teorema de Convolución en el espacio: $g(x,y) = h(x,y) * f(x,y)$

Cada píxel de la nueva imagen se obtiene mediante el sumatorio de la multiplicación del kernel por los píxeles contiguos:

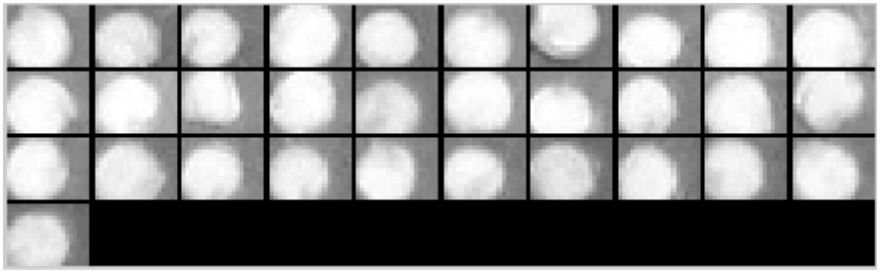
$g(x,y) = \sum\sum f(i,j)\, w(i,j)$

Generalmente se divide sobre cierto valor constante para normalizar que suele obtenerse de la suma de los valores del kernel empleado.

4.2.3.2. Segmentación de la Imagen.

Uno de los objetivos que perseguimos con la segmentación de la imagen es lograr dividir la imagen y que cada una de estas imágenes representen una quinua para poderla analizar más adelante en forma separada tal como se muestra en la figura siguiente:

Ilustración 20: Imagen Segmentada de las Quinuas.



Fuente: Elaboración Propia FINCyT

Los algoritmos de segmentación que utilizaremos se basan en los siguientes principios:

a). *Discontinuidades del nivel de Colores.* Debido a que en nuestro caso no estamos buscando un solo color si no esta será configurable de acuerdo a las especificaciones del usuario, segmentaremos la imagen a partir de los cambios grandes en los niveles de colores entre los píxeles. Las técnicas que utilizan las discontinuidades como base son la detección de líneas, de bordes, de puntos aislados, los cuales ya hemos explicado en los puntos anteriores.

b). *Similitud de niveles de Colores*. Como es un método contrario al anterior, se utilizara para el caso de que se quiera establecer el tipo de color de quinua que se está observando. Las divisiones de la imagen se realizaremos agrupando los píxeles que tienen unas características similares.

Método de Agrupamiento (Clustering)

El Algoritmo de las K-medias es una técnica iterativa que se utiliza para dividir una imagen en K clusters. El algoritmo básico es:

1. Escoger K centros de clusters, ya sea de forma aleatoria o basándose en algún método heurístico.
2. Asignar a cada píxel de la imagen el clúster que minimiza la varianza entre el pixel y el centro del cluster.

3. Recalcular los centros de los clusters haciendo la media de todos los pixeles del cluster.
4. Repetir los pasos 2 y 3 hasta que se consigue la convergencia (por ejemplo, los pixeles no cambian de clusters).

En este caso, la varianza es la diferencia absoluta entre un píxel y el centro del clúster. La diferencia se basa típicamente en color, la intensidad, la textura, y la localización del pixel, o una combinación ponderada de estos factores. El número K se puede seleccionar manualmente, aleatoriamente, o por una heurística. Este algoritmo garantiza la convergencia, pero puede devolver una solución que no sea óptima. La calidad de la solución depende de la serie inicial de clusters y del valor de K. En estadística y aprendizaje automático, el algoritmo de las k-medias es un algoritmo de agrupamiento para dividir objetos en k grupos, donde k <n. Es similar al algoritmo de maximización de expectativas para las mezclas de gaussianas ya que ambos pretenden encontrar los centros de agrupaciones naturales de los datos. El modelo requiere que los atributos del objeto correspondan a los elementos de un espacio vectorial. El objetivo es intentar alcanzar al mínima varianza total entre clusters, o, la función de error al cuadrado. El algoritmo de las k-medias fue inventado en 1956.

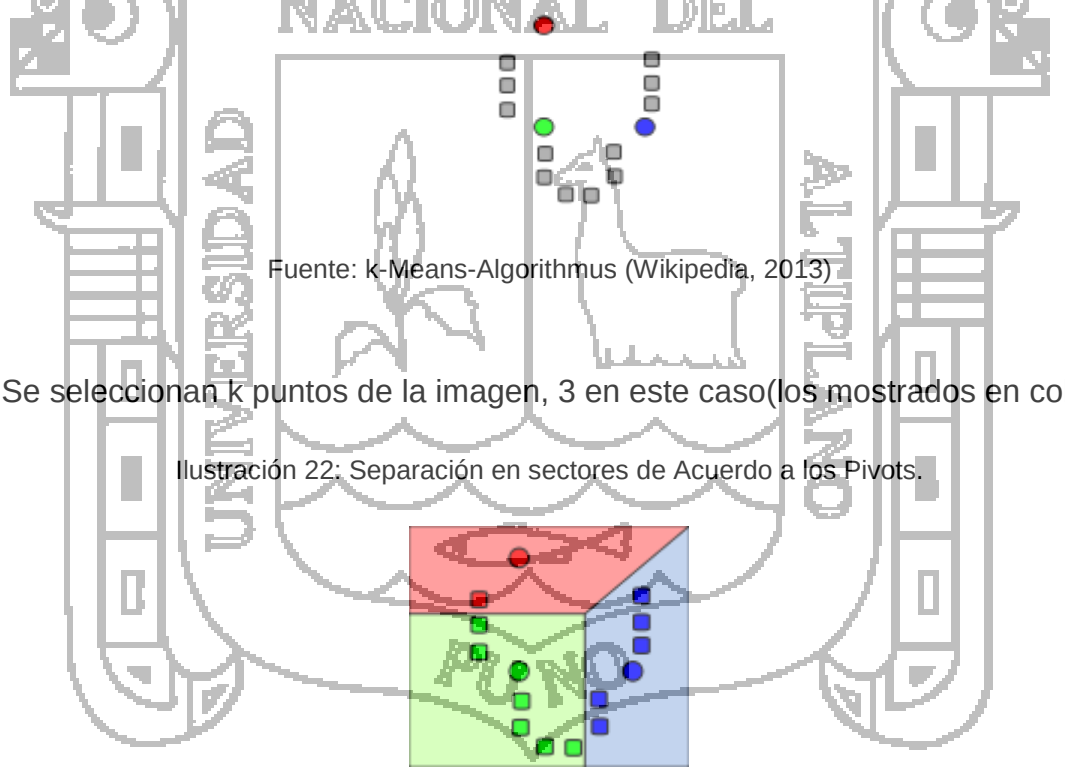
La forma más común del algoritmo usa una heurística de refinamiento conocido como el algoritmo de Lloyd. El algoritmo de Lloyd comienza dividiendo los puntos de entrada en k conjuntos iníciales, ya sea al azar o usando algunos datos heurísticos y a continuación, calcula el punto medio o centro de gravedad de cada conjunto. Se construye una nueva partición, asociando cada punto con el centro de gravedad más cercano.

Luego se recalculan los baricentros es para los nuevos clusters, y el algoritmo se repite alternando la aplicación de estos dos pasos hasta que la converja, que se obtiene cuando los puntos ya no cambiar de cluster (o los centros de gravedad ya no se modifican). Los algoritmos de Lloyd y de las K-medias a menudo se utilizan como sinónimos, pero en realidad el algoritmo de Lloyd es una heurística para resolver el problema de las K-medias, como ocurre con ciertas combinaciones de puntos de partida y baricentros, el algoritmo de Lloyd puede converger a una solución incorrecta.

Existen otras variantes, pero el algoritmo de Lloyd es el más popular, porque converge muy rápidamente. En cuanto al rendimiento, el algoritmo no garantiza que se devuelva un óptimo global. La calidad de la solución final depende en gran medida del conjunto inicial de clusters, y puede, en la práctica, ser mucho más pobre que el óptimo global. Dado que el algoritmo es extremadamente rápido, es un método común ejecutar el algoritmo varias veces y devolver las mejores agrupaciones obtenidas. Un inconveniente del algoritmo de las k-medias es que el número de clusters k es un parámetro de entrada. Una elección inadecuada de k puede dar malos resultados. El algoritmo también asume que la varianza es una medida adecuada de la dispersión del cluster. (Wikipedia, 2013)

Demostración del algoritmo estándar

Ilustración 21: Imagen con puntos Pivots.



Fuente: k-Means-Algorithmus (Wikipedia, 2013)

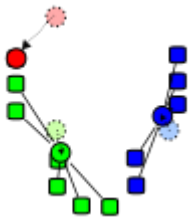
1) Se seleccionan k puntos de la imagen, 3 en este caso(los mostrados en color).

Ilustración 22: Separación en sectores de Acuerdo a los Pivots.

Fuente: k-Means-Algorithmus (Wikipedia, 2013)

2) Se crean k clusters a partir de los puntos anteriores.

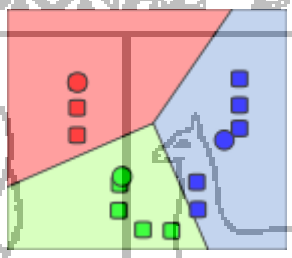
Ilustración 23: Creación de Clusters.



Fuente: k-Means-Algorithmus (Wikipedia, 2013)

3) El centro de gravedad de cada uno de los grupos k se convierte en el nuevo medio.

Ilustración 24: División de la Imagen Original



Fuente: k-Means-Algorithmus (Wikipedia, 2013)

4) Los pasos 2 y 3 se repiten hasta que se alcance la convergencia.

La forma de abordar o la técnica será la combinación de ambas de tal forma que se pueda lograr mucha más rapidez en el procesamiento de la imagen

Fíjese en la imagen a continuación, nótese que tenemos un sector determinando por 32 pixeles de alto y de ancho, la forma de acercarse a este borde es por el lado izquierdo superior, así en la imagen el primer punto a detectar es el 2, 13 .

A partir de ese punto determinamos hacia el eje x positivo el tamaño del segmento y en forma diagonal hacia la derecha el alto de la misma.

Ilustración 25: Imagen de una Quinua en Forma de Vector.

Fuente: Elaboración Propia FINCyT

Este método se implementó como una primera versión para la segmentación de granos, pero debido a la necesidad de identificar áreas y centros de posición de cada segmento de manera más rápida (de acuerdo a las pruebas realizadas), se optó por utilizar otro método, el medo de detección de bordes.

Método de Detección de Bordes

La detección de bordes es un campo bien desarrollado por sí mismo en el procesamiento de imágenes. Los límites de regiones y los bordes están estrechamente relacionados, ya que a menudo hay un fuerte ajuste en la intensidad en los límites de las regiones. Las técnicas de detección de bordes pueden ser usadas como otra técnica de segmentación más. Los bordes identificados por la detección de bordes en ocasiones están desconectados. Para segmentar un objeto a partir de una imagen sin embargo, es necesario que los bordes formen figuras cerradas.

Este algoritmo esta implementado en la librería OpenCV, por lo que su aplicación en nuestro proyecto se hace menos compleja.

La siguiente función es utilizada para nuestro propósito:

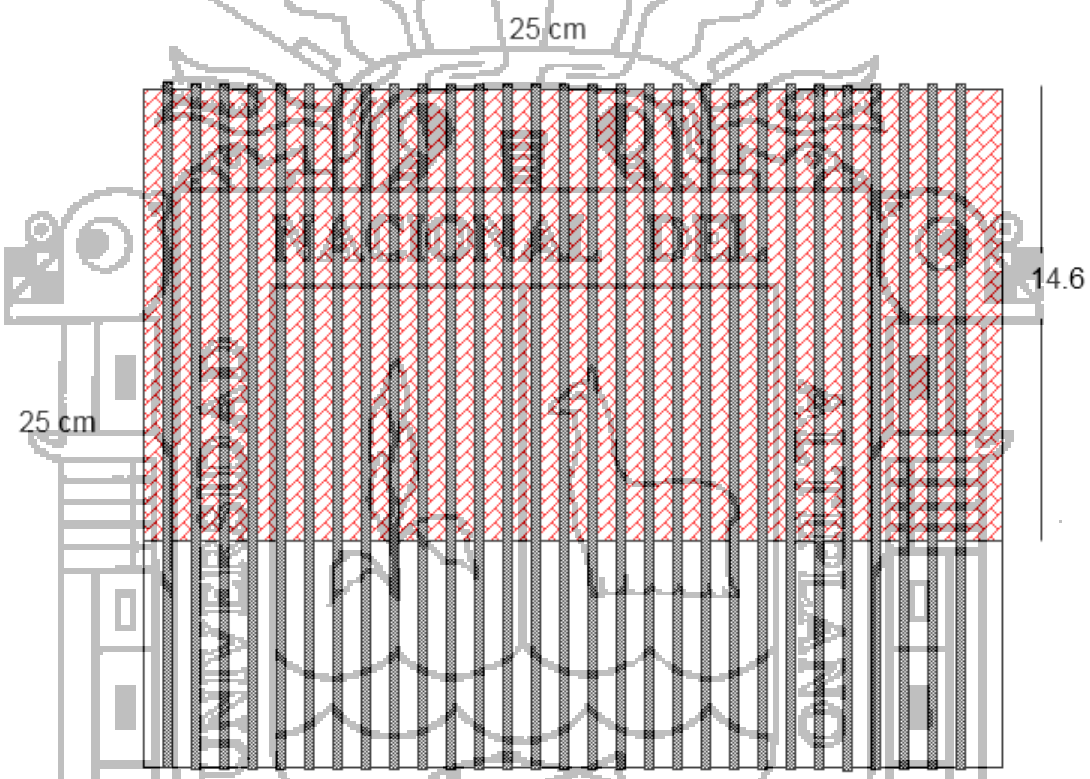
```
void findContours(InputOutputArray image, OutputArrayOfArrays contours,
OutputArray hierarchy, int mode, int method, Point offset=Point())
```

La función recupera los contornos de la imagen binaria utilizando el algoritmo [Suzuki85]. Los contornos son una herramienta útil para el análisis de la forma y la detección de objetos y el reconocimiento.

Consideraciones de Hardware:

Para la segmentación de la imagen y detección de posiciones, se tendrá que considerar la estructura mecánica en la cual se captura las imagenes, para establecer la densidad de pixeles necesarios por cm cuadrado que se requiere para un correcto procesamiento.

Ilustración 26: Tamaño del Panel a ser fotografiado



Fuente: Elaboración Propia FINCyT

Análisis obtenido de acuerdo al Tamaño del Panel y el tamaño de la Fotografía:

- La característica es que la Fotografía es de 1920 x 1080 pixeles
- El tamaño de la parte del panel que será fotografiada es de 250mm x

250 mm	1920 pixel
?	1080 pixel

Entonces =(250mm x 1080px)/ 1920px = 140.6mm

- El tamaño en mm de la imagen a evaluar es 4000 x 1875 mm
- Determinaremos ahora cuanto es un Pixel en milímetros

250 mm	1920	pixel
?	1	pixel

Entonces = > 1 Pixel = (250 x 1)/1920 = 0.1302 mm

- Debido a que un grano de quinua promedio tiene 2mm entonces esto será cubierto por la imagen de un grano de Quinua como **máximo 15.36 *pixeles*** (1.875 mm) lo cual es aceptable para el procesamiento de selección de color.

4.2.3.3. Algoritmos utilizados

Se implementaron los siguientes algoritmos para cada paso del proceso del procesamiento de las imágenes.

Algoritmo para la selección de la imagen

1. Lectura de la Imagen (1920 x 1080 pixels)
2. Se toma una imagen de 1920 pixels de ancho x 15 pixels de alto empezando de la parte superior.
3. SI se verifica si la imagen contiene exactamente a un grano es decir
   a. Se Anota la coordenada del punto de inicio.
4. SI NO
   a. Se continúa con la búsqueda hacia abajo y luego hacia la derecha de la imagen.

Algoritmo para verificar si la imagen contiene un grano

Ilustración 27: Algoritmo para verificar si la imagen contiene un grano.

1. INICIO
2. IMGSEL = Lectura de la imagen
3. V_IMGSEL= Crear Vector de 25 x 1
4. Para a=1 hasta 25
   a. V_IMGSEL[a]=0
5. Para a=1 hasta 25
   a. Se Recorre a la imagen a
   b. Si la Imagen a es contenida_perfectamente ENTONCES V_IMGSEL[a]=0
6. FIN

Algoritmo para el recorrido a la imagen n

1. INICIO

2. Ejex = 0;
3. V_IMGSEL= Crear Vector de 25 x 1
4. Para a=1 hasta 25
    a. V_IMGSEL[a]=0
5. Para a=1 hasta 25
    a. Se Recorre a la imagen a
    b. Si la Imagen a es contenida_perfectamente ENTONCES
        V_IMGSEL[a]=0
6. FIN

Algoritmo para el recorrido a la imagen en caso que el análisis sea por canaletas.

1. INICIO
2. LARGO = 1912 - 33; % Largo de la Imagen
3. ALTO = 1068 - 33; % Alto de la Imagen
4. Alto_Quinua = 15;
5. Ancho_Quinua = 20;
6. V_IMGSEL= Crear Vector de 25 x 1
7. PARA x = 1 : Ancho_Quinua

SI (A(SA,SL+x,1)>Color) ,Q_Largo = Q_Largo +2^(Ancho_Quinua-x);

end

8. SI Largo >AltoImagen

CONTA_IMAG_SEL = CONTA_IMAG_SEL + 1;

IMAG_SEL (CONTA_IMAG_SEL,:) = [SA-7,SL];

9. FIN

Algoritmo para saber si una imagen escontenida_perfectamente

1. Inicio
2. Para y = 1 : (CONTA_IMAG_SEL)
3. Si ((SA>=IMAG_SEL(y,1)) & (SA<=IMAG_SEL(y,1)+Ancho_Quinua)) & ((SL>=IMAG_SEL(y,2)) & (SL<=IMAG_SEL(y,2)+Ancho_Quinua))

4. SW = 1;
5. Fin

Nota: El Tamaño de la Quinua ya sea en ancho o alto en diámetro o son especificados por el usuario.

Algoritmo del Recorrido.

A continuación la parte principal del código fuente del algoritmo del recorrido en Matlab.

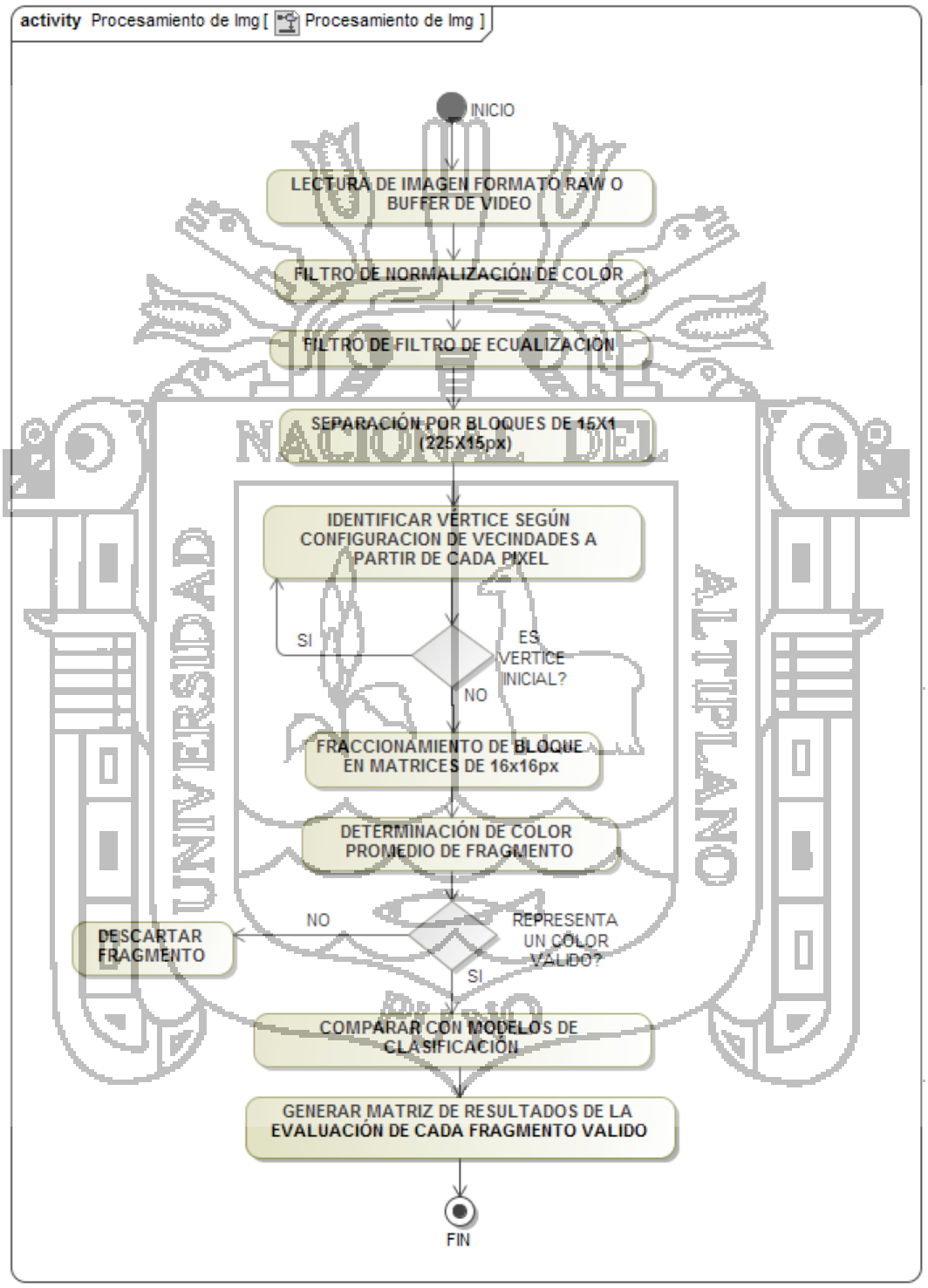
```
clearall;

[A,map,alpha] = imread('prueba.png');

LARGO  = 1912 - 33;  % Largo de la Imagen
ALTO   = 1068 - 33;  % Alto de la Imagen

CONTA_SEL = 1;
clearSECTOR;
CONTA_IMAG_SEL = 1;
IMAG_SEL(1,:)=[2000,2000];
II=200;
Alto_Quinua = 15;
Ancho_Quinua = 20;
ForSL=1: (LARGO )
%Analizando el Sector de Izquierda a Derecha
%SECTOR = A(45:78,20:53,1);
ForSA=1: ALTO
if A(SA,SL,1)>II
%Evaluando si el punto se encuentra ya registrado
                EX=0;
for y = 1 : (CONTA_IMAG_SEL)
%Y,X
If ((SA>=IMAG_SEL(y,1)) & (SA<=IMAG_SEL(y,1)+Ancho_Quinua)) &
((SL>=IMAG_SEL(y,2)) & (SL<=IMAG_SEL(y,2)+Ancho_Quinua))
    EX = 1;
break
end
end
%Fin de Evaluando
if EX==0
Q_Largo = 0;
for x = 1 : Ancho_Quinua
if (A(SA,SL+x,1)>II) ,Q_Largo = Q_Largo +2^(Ancho_Quinua-x);, end
end
ifQ_Largo> 1048320
```

```
CONTA_IMAG_SEL = CONTA_IMAG_SEL + 1;
IMAG_SEL (CONTA_IMAG_SEL,:) = [SA-7,SL];

end
end
end
end
end
AnchoIS  = 10;
AltoIS   = 10;
XAnchoIS = 1;
YAnchoIS = 1;
XAltoIS  = 1;
YAltoIS  = 1;
for a=2:CONTA_IMAG_SEL
V_Imag_Sel(YAltoIS:YAltoIS+14,XAnchoIS:XAnchoIS+Ancho_Quinua-1)=
A(IMAG_SEL(a,1):IMAG_SEL(a,1)+14,IMAG_SEL(a,2):IMAG_SEL(a,2)+Ancho_Quinua-1,1);

XAnchoIS = XAnchoIS + Ancho_Quinua+1;
ifXAnchoIS>Ancho_Quinua*10 ,XAnchoIS=1;YAltoIS = YAltoIS + 16; ,end
end
imshow(V_Imag_Sel);
```

4.2.3.4. Diagrama General del Algoritmo

A continuación se presenta el diagrama general del algoritmo inicial.

Ilustración 28: Algoritmo General Inicial



Fuente: Elaboración Propia

4.2.3.5. Resultados Iniciales

Los algoritmos presentados fueron fueron probados unitariamente en matlab y c++ donde se observó la falta de velocidad en el procesamiento de dichos algoritmos, y en base a estos, se determinó mejorar el algoritmo para que pueda ejecutarse más rápidamente, ya que en el proyecto requiere de una elevada velocidad de procesamiento la cual se verá reflejada en la eficiencia de la maquina a implementar.

Para lo cual se optó utilizar el método de segmentación por detección de bordes y obtención de momentos (De acuerdo al teorema de Green) a partir de dichos bordes, para finalmente obtener el área y el punto medio de cada curva cerrada que representa a cada grano de quinua en el espacio.

Para poder implementar este algoritmo se hace uso de la librería OpenCV la cual se explica en el siguiente punto.

4.2.4. Análisis y Diseño del Sistema

4.2.4.1. Captura de Video

Para la captura de frames de la secuencia de video se utilizaran las funciones de OpenCV:

Inicialización de captura desde cámara:

*CvCapture capture = cvCaptureF romCAM(0); //*Capturando el video desde el dispositivo 0:

Captura de Frame:

cvGrabF rame(capture);

Obtención de frame capturado:

img = cvRetrieveF rame(capture);

Liberación de la fuente de captura:

cvReleaseCapture(&capture);

Obtención de propiedades del dispositivo de captura:

cvQueryF rame(capture);

La siguiente sentencia es necesaria para obtener las propiedades correctas de captura

cvGetCaptureProperty(capture; property id; value);

Identificadores de propiedad:

- CV_CAP_PROP_ FRAME_HEIGHT
- CV_CAP_PROP_FRAME_WIDTH
- CV_CAP_PROP_FPS
- CV_CAP_PROP_FRAME_COUNT

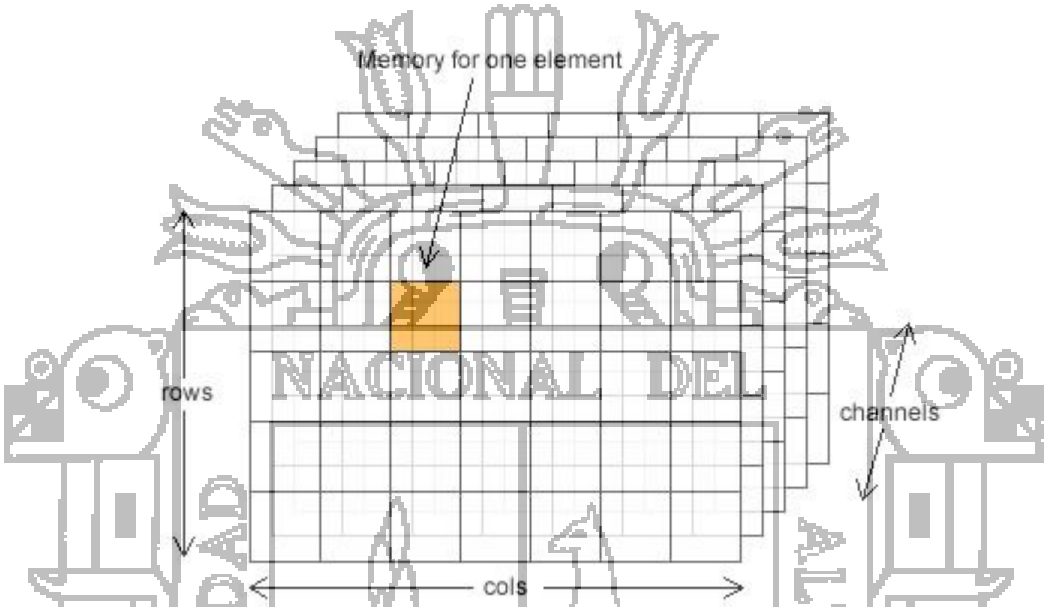
La captura se realizará cada X milisegundos, donde X es aproximadamente 50 y varía de acuerdo a la calibración de velocidad del hardware.

4.2.4.2. Estructura de almacenamiento de una imagen:

Dependiendo del tipo de color y calidad necesitemos manejar se tienen una variedad estructuras disponibles de almacenamiento predefinidas en OpenCV.

La fórmula general es: CV_<profundidad_bits>(S|U|F)C<numero_canales>

Ilustración 29: Estructura de almacenamiento de imágenes en OpenCV



Fuente: (Bradski Gary, 2008)

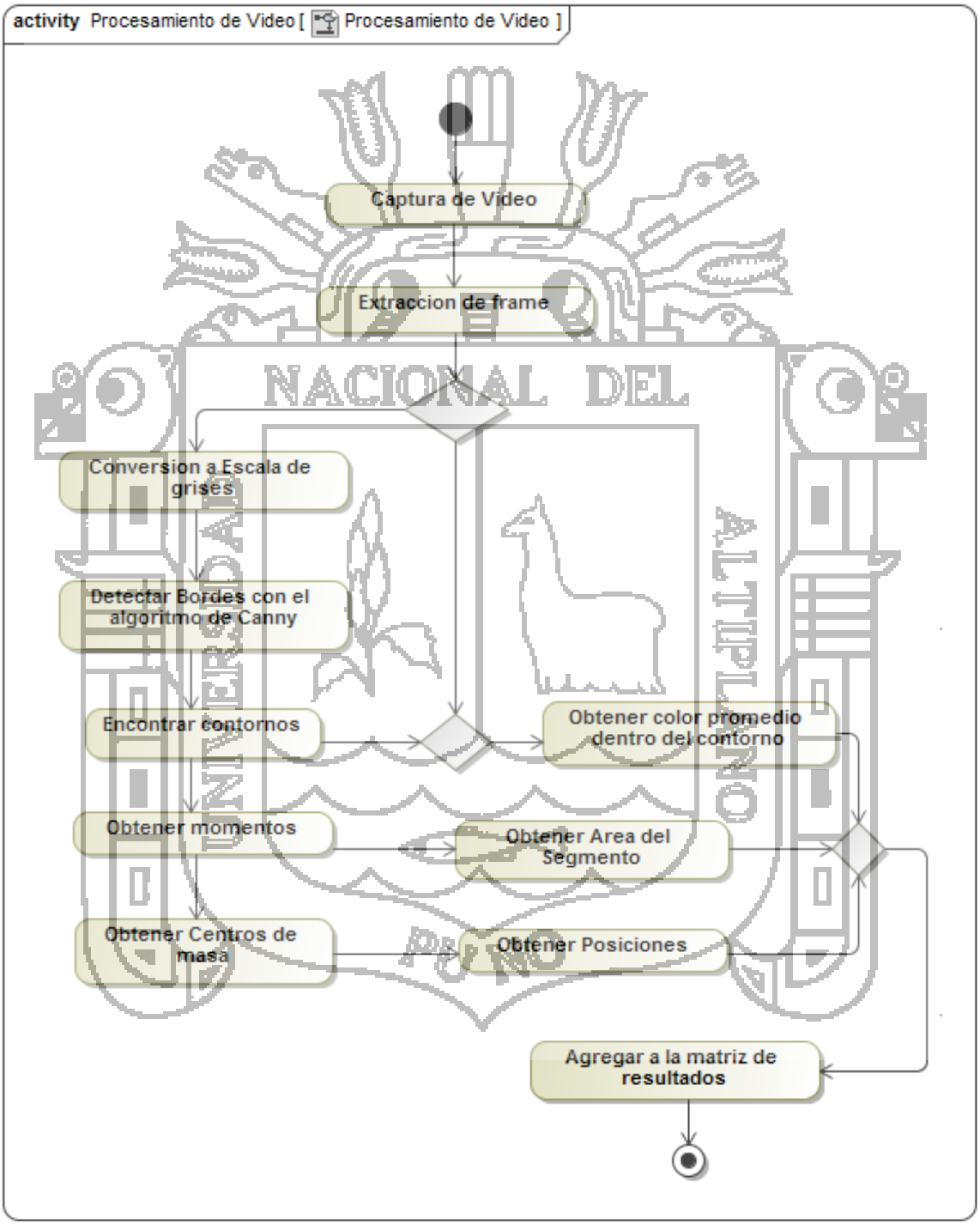
- La profundidad de bits puede ser 8, 16, 32 o 64 bits. Esta es la cantidad de memoria para el celular de color naranja solo en la imagen de arriba.
- S significa entero, U significa entero sin signo y F significa coma flotante (valores decimales)
- Podemos disponer de cualquier número de Canales.

El tipo más utilizado en el proyecto será CV_8UC3 matrices de enteros sin signo de 8 bits con 3 canales.

4.2.4.3. Algoritmo de procesamiento de Imagen

A continuación se presenta el algoritmo general final del procesamiento de un frame del video capturado:

Ilustración 30: Algoritmo del procesamiento de un frame



Fuente: Elaboración Propia.

En el Anexo 02 se presentan el código fuente de las funciones principales para el procesamiento de imágenes.

A continuación se presentan los resultados en cada uno de los pasos del algoritmo

En la ilustración 32 se muestra el proceso del procesamiento de un frame capturado.

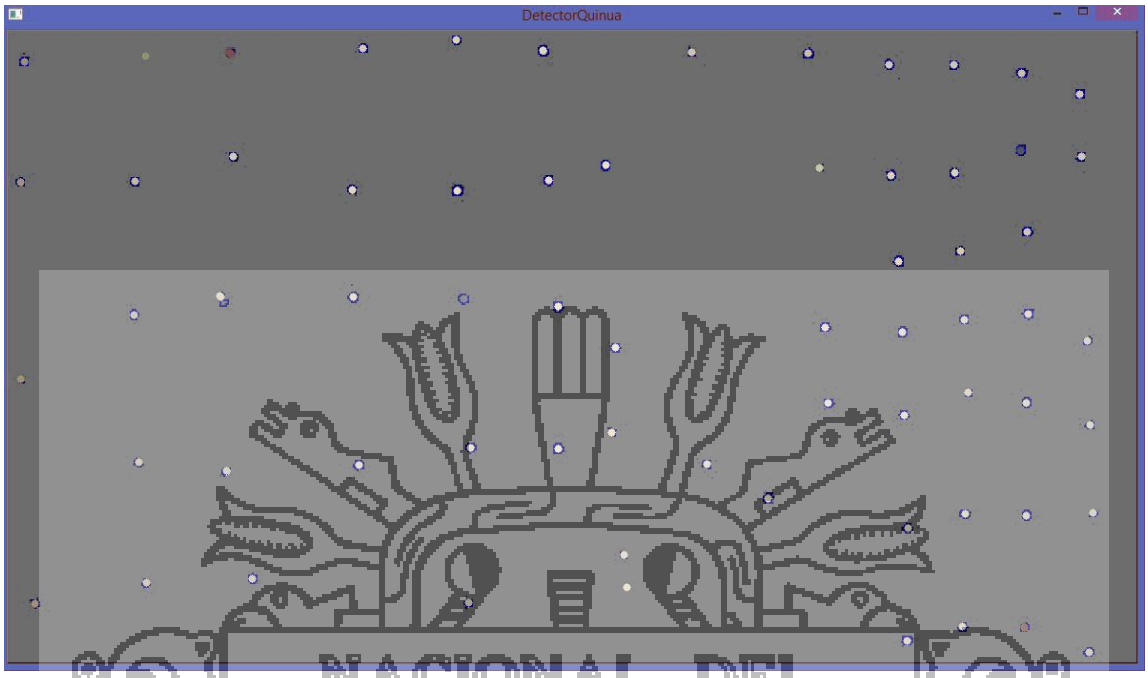
Captura y separación de 1 frame:

Ilustración 31: Frame Inicial

Fuente: Elaboración propia.

El resultado del filtrado, detección de contornos, momentos y centros de masa se presenta en la ilustración 33.

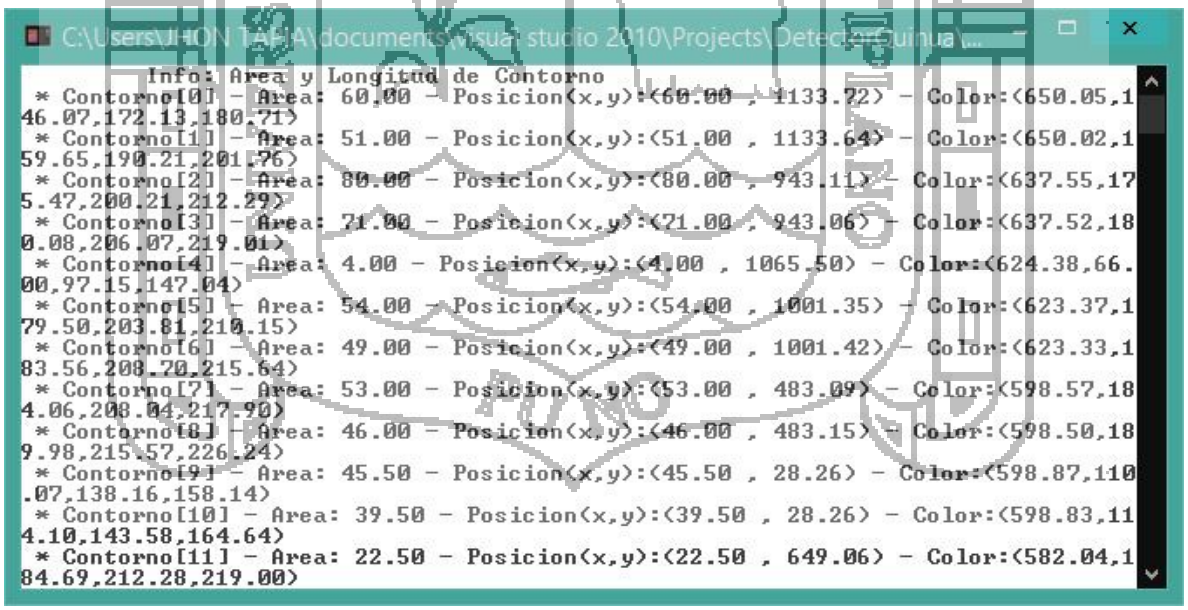
Ilustración 32: Frame con contornos y colores detectados



Fuente: Elaboración propia.

Matriz de resultados del procesamiento.

Ilustración 33: Resultados del procesamiento de Imágenes



Fuente: Elaboración propia.

4.2.5. Transferencia de Resultados

Para la transferencia de datos de la PC al Raspberry mediante el estándar Ethernet, se implementaron los programas para la comunicación por Sockets usando el protocolo UDP.

Los datos procesados por la PC devuelven como resultado una matriz de datos en memoria en tiempo de ejecución del programa de procesamiento de imágenes en C++, esta matriz es transmitida cada cierto tiempo (tiempo definido por calibración) mediante sockets al módulo Raspberry PI.

Debido a que el programa en la PC es la que controla el periodo de envío este se comporta como un cliente socket y por lo tanto el servidor socket seria el módulo Raspberry Pi.

En el Anexo 03 se muestra el código fuente de la parte de transferencia del lado servidor de datos del CPU con Windows y La placa Raspberry con Linux.

4.2.6. Interpretación de datos y expulsión

Una vez obtenido los datos mediante el procesamiento de imágenes en la PC, estos son transferidos al módulo Raspberry PI mediante una conexión punto a punto (cable UTP) por sockets.

Según los datos proveídos por la unidad de procesamiento (punto de tiempo de captura, posición y decisión de expulsión de cada grano) el módulo Raspberry Pi se encargará de mandar la señal de expulsión a la electroválvula o compuerta de selección de los granos indicados mediante la habilitación de los pines correspondientes a cada hilera en un bucle sincronizado de captura y expulsión.

Para este proceso se utilizará el puerto Ethernet RJ45 del módulo Raspberry Pi y el puerto GPIO.

Una vez transferida la matriz de resultados del procesamiento de imágenes al módulo Raspberry PI, esta es almacenada e interpretada de la siguiente manera:

La matriz principal está conformada por un conjunto de 24 matrices (1 matriz por hilera) de dimensión 3xL donde L puede variar de 0 a 15, que representa la cantidad de granos a expulsar en cada hilera en el bloque actual. Cada sub-matriz tiene como componente el valor de la posición en la hilera, la calificación de expulsión y el tiempo en el que fue capturado.

El módulo Raspberry inicia un bucle de censado repetitivo al pin del puerto GPIO conectado al sensor de paso correspondiente a cada hilera. Una vez que ocurra una variación o interrupción en dicho sensor se realiza la lectura de la matriz correspondiente al bloque e hilera obteniendo el valor de la decisión de expulsión o no. Si el valor indica expulsión el módulo Raspberry generará una señal de activación de la compuerta de la hilera correspondiente.

Para el uso del puerto GPIO será necesario adicionar un buffer de Entrada y Salida que además servirá para elevar el nivel de tensión de 3.3V a 5V, de esa forma las señales de expulsión y lectura del sensor estarán estandarizados a 5V.

4.2.7. Pruebas del Procesamiento de Imágenes

Todas las pruebas fueron realizadas de manera unitaria como vimos en los capítulos anteriores, teniendo en cuenta que aún no se tenía la faja transportadora en funcionamiento por lo cual se optó por las pruebas con secuencias de imagen estáticas.

Se observó en dichas prueba que el algoritmo de procesamiento de imágenes es eficiente determinando el color valido de los granos de quinua en un 95% (debido a la posición e iluminación) y la selección de granos a expulsar en un 90% lo cual corresponde al complemento del margen de error establecido para la decisión, es decir si el grano de quinua es gris en un 52% existe la probabilidad de que este no sea expulsado ya que para tener la certeza de la expulsión este debería tener como mínimo un 55% de gris.

Para la siguiente etapa se presentaran las pruebas finales en movimiento una vez terminada la etapa de la parte mecánica de dosificación y transporte de quinua mediante la faja transportadora.

4.2.8. Diseño del Módulo de Configuración

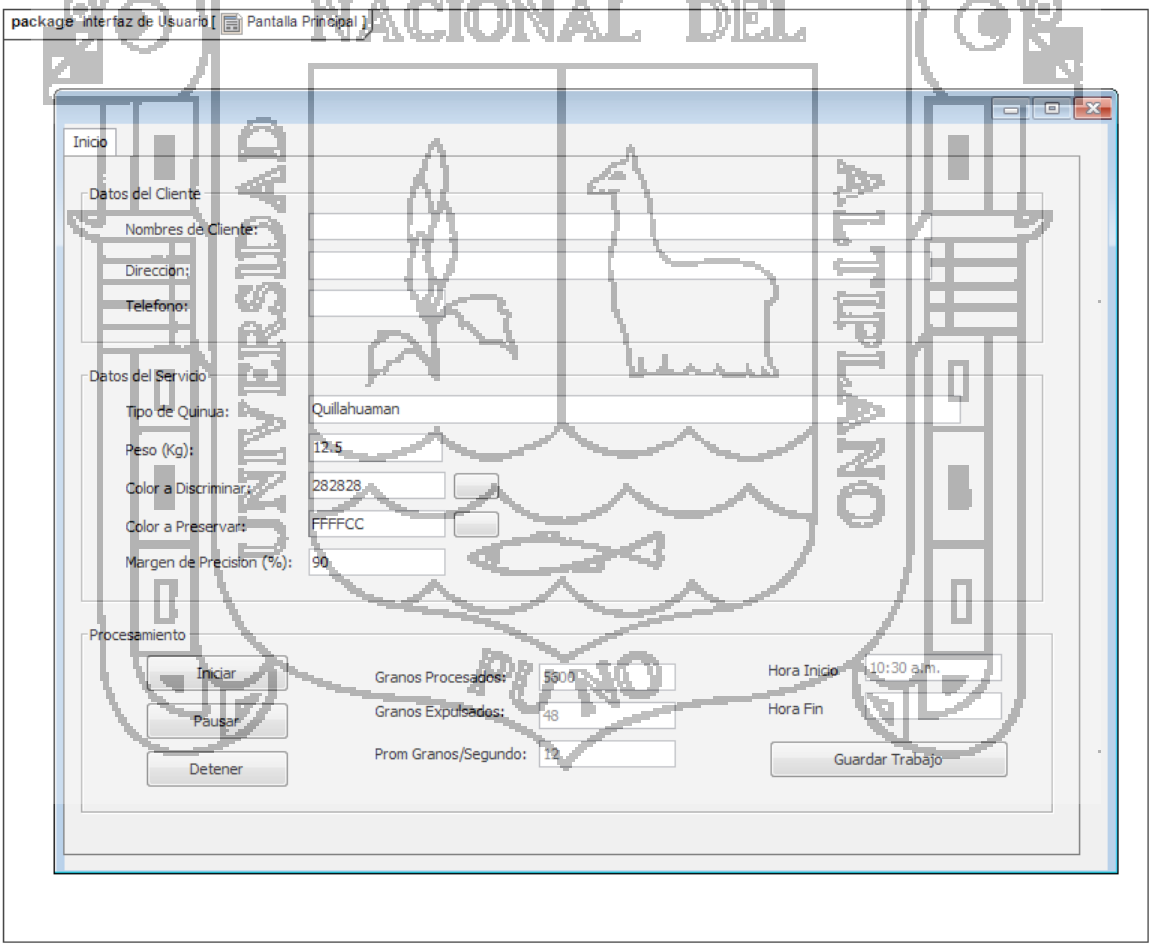
El desarrollo de un aplicación de configuración y monitoreo de datos y estado de la máquina, para lo cual se desarrolló una aplicación en Visual C++, la cual Transfiere los datos seleccionados por el usuario al módulo de control mediante comunicación por sockets.

4.2.9. Interfaz de usuario

A continuación se muestra la interfaz del usuario:

Interfaz de Inicio:

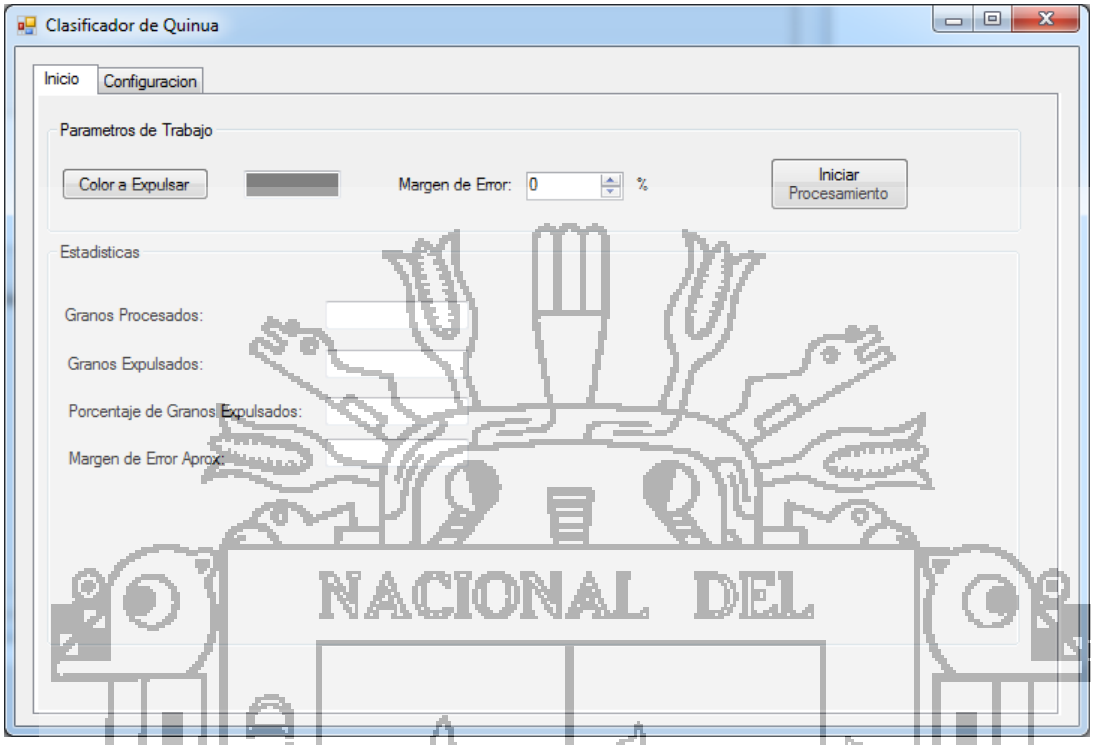
Ilustración 34: Diseño de la Interfaz de Usuario



Fuente: Elaboración propia.

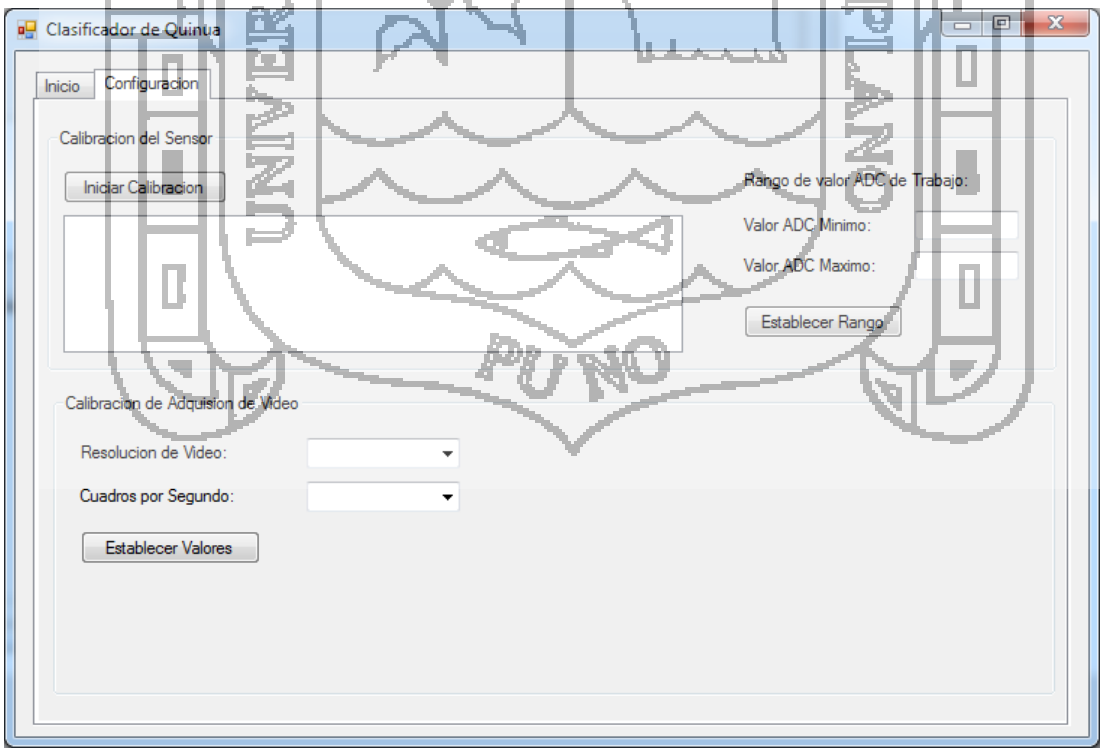
Prototipo Inicial de la Pantalla de Inicio y configuración, implementado en VC++.

Ilustración 35: Prototipo de la Pantalla de Inicio VC++



Fuente: Elaboración Propia

Ilustración 36: Interfaz de Configuración VC++



Fuente: Elaboración Propia

La aplicación nos ofrece la posibilidad de configurar los siguientes parámetros:

- Color del grano de quinua a expulsar.
- Margen de error de color.
- Calibración de valores mínimos y máximos del Conversor Análogo Digital de la placa de control.
- Configuración de valores de resolución y velocidad de video de la cámara.

4.2.10. Pruebas de Configuración

Las pruebas de configuración de módulos se desarrollaron separadamente por cada módulo, teniendo en cuenta que estas todavía estaban en proceso de implementación por lo que se presentaron cambios durante el proceso de desarrollo de la aplicación opciones de configuración y monitoreo.

Esta aplicación deberá estar terminada una vez finalizada la implantación de todos los módulos externos (dosificación, faja trasportadora, placa de control circuitos de censado y los circuitos de expulsión), pues esta organiza la parametrización de todos estos módulos los que están en etapa de implementación y pruebas.

CONCLUSIONES

1. El desarrollo del sistema de interpretación de las características colorimétricas y morfométricas del grano de quinua, se presenta como una alternativa frente a los costosos sistemas de selección óptica, aprovechando dispositivos de uso diario y de fácil acceso.
2. El sistema diseñado en la presente investigación nos muestra la posibilidad de incluirla en sistemas más completos, como por ejemplo una máquina de selección óptica de quinua o una máquina de control de calidad de quinua.
3. El software desarrollado tendrá un efecto relativo en la eficiencia y la eficacia de la maquina clasificadora pues estos índices también dependerán de la velocidad del hardware y principalmente en el módulo de expulsión mecánico.
4. La implementación del software va de la mano con la implementación del hardware, pues si se presentan cambios en el hardware el software desarrollado para la PC sufrirá cambios, y esta finalizara una vez que los demás módulos terminen de implementarse.

RECOMENDACIONES

Para la implementación de sistemas finales en base al sistema desarrollado en la presente investigación, se recomienda optimizar los recursos usados en el módulo de procesamiento de datos haciendo uso de las nuevas alternativas de hardware para la captura de imágenes y su procesamiento tales como los dispositivos embebidos con Linux como por ejemplo Raspberry pi, BeagleBone, PCDuino, etc. Los cuáles debido a su costo reducirán significativamente los costos generales.

Para el procesamiento continuo de granos de quinua distribuidos en una faja transportadora para su implementación en un producto final, se sugiere adicionar opciones de configuración en el software, tales como calibración de velocidad de transporte, frecuencia de dosificación, nivel de iluminación, etc. con el fin de obtener una calibración más exacta y adaptable al sistema final.

BIBLIOGRAFÍA

Food and Agriculture Organization. (Julio de 2011). *La Quinua: Cultivo milenario para contribuir a la seguridad alimentaria mundial.* Obtenido de http://www.fao.org/fileadmin/templates/aiq2013/res/es/cultivo_quinua_es.pdf

Alonso Amo, F., Martínez Normand, L., & Segovia Pérez, F. J. (2005). *Introducción a la ingeniería del software.* Delta Publicaciones.

Apaza, S. (2008). *Competitividad de la quinua: una aplicación del modelo de Michael Porter.* Obtenido de http://www.fao.org/agronoticias/agro-noticias/detalle/es

Bradski Gary, K. A. (2008). *Learning OpenCV* (Primera ed.). O'Reilly Media, Inc.

Caracterización de Variables Continuas y Discretas del Grano de Quinua. (15 de Diciembre de 2013). Obtenido de http://www.bensoninstitute.org/Publication/RELAN/V15/Caracterizacion.htm

Castro, M. M. (2011). *Base de datos*. Recuperado el 20 de Abril de 2011, de http://bibliotecavirtualut.suagm.edu

Chenopodium Quinoa. (10 de Octubre de 2013). Obtenido de http://es.wikipedia.org/wiki/Chenopodium_quinoa

Etxeberria, Jon Aristondo. (2010). *Algoritmo de reconocimiento de forma y color para una plataforma robótica.* Obtenido de http://www.ehu.es/documents/1545039/1570316/10jaristondo.pdf

Marca Vilca, S., Chaucha Jove, W., Quispe Quispe, J. C., & Mamani Centón, V. (s.f.). *Comportamiento Actual De Los Agentes De La Cadena Productiva De Quinua En La Región Puno.* Dirección Regional Agraria Puno.

NASA Technical paper 3422. (1983). *Quinoa: An Emerging «New» crop with potential of CELSS. Porcentaje de proteína con referencia a sus componentes.*

OpenCV Developers Team. (2014). Recuperado el 18 de Julio de 2014, de OpenCV 2.4.9.0 documentation: http://docs.opencv.org/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html?highlight=moments#moments

Procesamiento digital de imágenes. (Diciembre de 2013). Obtenido de http://es.wikipedia.org/wiki/Procesamiento_digital_de_im%C3%A1genes

Rojas, W. (1998). *Análisis de la diversidad genética del germoplasma de quinua.*

Stevens, P. y. (2002). *Utilización de UML Ingeniería de Software con Objetos y Componentes.* Madrid España: Addison Wesley.

Tapia, M. E. (2000). *Cultivos Andinos Subexplotados y su Aporte a la Alimentación.*

Valenzuela Riaño, J. (2007). *Caracterización morfométrica y calorimétrica del grano pulido de arroz, Utilizando reconocimiento de Patrones y Análisis Fractal de Imágenes Digitales.*

Villacresp, E., Peralta, E., Egas, L., & Mazon, N. (2011). *Potencial agroindustrial de la quinua.* Voletin Divulgativo 146.

ANEXOS

Anexo 01

Variación y/o combinaciones de color de pericarpio, espisperma y perisperma.

Código	Pericarpio		Color del episperma	Apariencia del perisperma	Nº Accesión	%
	Aspecto	Color				
1	Cenizo	Habano	Transparente	Vítreo	12	0.51
2	Cenizo	Crema suave	Transparente	Vítreo	4	0.17
3	Cenizo	Habano	Transparente	Opaco	155	6.57
4	Cenizo	Crema suave	Transparente	Opaco	268	11.35
5	Cenizo	Habano	Canela	Opaco	28	1.19
6	Cenizo	Habano	Café	Opaco	5	0.21
7	Cenizo	Habano	Café oscuro	Opaco	22	0.93
8	Cenizo	Habano	Negro	Opaco	17	0.72
9	Cenizo	Crema suave	Transparente	Opaco	16	0.68
10	Cenizo	Amarillo azufrado	Transparente	Opaco	18	0.76
11	Cenizo	Anaranjado	Transparente	Opaco	19	0.80
12	Cenizo	Rosado	Transparente	Opaco	23	0.97
13	Cenizo	Rosado oscuro	Transparente	Opaco	6	0.25
14	Cenizo	Rojo	Transparente	Opaco	1	0.04
15	Cenizo	Púrpura pálido	Transparente	Opaco	2	0.08
16	Cenizo	Café rojizo	Transparente	Opaco	9	0.38
17	Sucroso	Habano	Transparente	Vítreo	16	0.68
18	Sucroso	Crema suave	Transparente	Vítreo	23	0.97
19	Sucroso	Pajizo	Transparente	Vítreo	4	0.17
20	Sucroso	Anaranjado	Transparente	Vítreo	5	0.21
21	Sucroso	Rosado	Transparente	Vítreo	5	0.21
22	Sucroso	Rosado oscuro	Transparente	Vítreo	2	0.08
23	Sucroso	Púrpura	Transparente	Vítreo	3	0.13
24	Sucroso	Café oscuro	Transparente	Vítreo	1	0.04
25	Sucroso	Habano	Transparente	Opaco	24	1.02
26	Sucroso	Crema suave	Transparente	Opaco	891	37.75
27	Sucroso	Habano	Transp. + canela	Opaco	1	0.04
28	Sucroso	Habano	Café claro	Opaco	1	0.04
29	Sucroso	Habano	Canela	Opaco	3	0.13
30	Sucroso	Habano	Café	Opaco	10	0.42
31	Sucroso	Habano	Café oscuro	Opaco	2	0.08
32	Sucroso	Pajizo	Transparente	Opaco	4	0.17
33	Sucroso	Crema suave	Transparente	Opaco	11	0.47
34	Sucroso	Amarillo claro	Transparente	Opaco	45	1.91
35	Sucroso	Amarillo azufrado	Transparente	Opaco	68	2.88
36	Sucroso	Anaranjado	Transparente	Opaco	139	5.89
37	Sucroso	Rosado	Transparente	Opaco	58	2.46
38	Sucroso	Rosado oscuro	Transparente	Opaco	74	3.14
39	Sucroso	Rojo	Transparente	Opaco	42	1.78
40	Sucroso	Café oscuro	Transparente	Opaco	5	0.21
41	Sucroso	Café rojizo	Transparente	Opaco	22	0.93
42	Sucroso	Café rojizo	Canela	Opaco	17	0.72
43	Sucroso	Café rojizo	Café oscuro	Opaco	13	0.55
44	Sucroso	Café rojizo	Negro	Opaco	6	0.25
45	Sucroso	Púrpura pálido	Transparente	Opaco	25	1.06
46	Sucroso	Púrpura pálido	Canela	Opaco	5	0.21
47	Sucroso	Púrpura pálido	Café oscuro	Opaco	9	0.38
48	Sucroso	Púrpura pálido	Negro	Opaco	3	0.13
49	Sucroso	Púrpura	Transparente	Opaco	11	0.47
50	Sucroso	Púrpura	Canela	Opaco	6	0.25
51	Sucroso	Púrpura	Café oscuro	Opaco	2	0.08
52	Sucroso	Púrpura	Negro	Opaco	4	0.17
53	Sucroso	Negro	Negro	Opaco	44	1.86
54	Sucroso	Amarillo claro	Canela	Opaco	7	0.30
55	Sucroso	Amarillo claro	Café oscuro	Opaco	8	0.34
56	Sucroso	Anaranjado	Café	Opaco	8	0.34
57	Sucroso	Anaranjado	Negro	Opaco	32	1.36
58	Sucroso	Café	Café	Opaco	3	0.13
59	Sucroso	Café claro	Negro	Opaco	8	0.34
60	Sucroso	Café casi verde	Negro	Opaco	21	0.89
61	Sucroso	Café oscuro	Canela	Opaco	1	0.04
62	Sucroso	Café oscuro	Café oscuro	Opaco	5	0.21
63	Sucroso	Café oscuro	Negro	Opaco	30	1.27
64	Sucroso	Negro	Café	Opaco	3	0.13
65	Sucroso	Negro	Café oscuro	Opaco	4	0.17
66	Sucroso	Mixtura	Especificar	Opaco	21	0.89
TOTAL					**2360**	**99.96**

Anexo 02

Código fuente del Algoritmo de procesamiento de un Frame:

```
Mat src; Mat src_gray;

int thresh = 100;
int max_thresh = 255;
RNG rng(12345);

/// Function header
void thresh_callback(int, void*);


IplImage* GetThresholdedImage(IplImage* img)
{
    // Convert the image into an HSV image
    IplImage* imgHSV = cvCreateImage(cvGetSize(img), 8, 3);
    cvCvtColor(img, imgHSV, CV_BGR2HSV);

    IplImage* imgThreshed = cvCreateImage(cvGetSize(img), 8, 1);

    // Values 20,100,100 to 30,255,255 working perfect for yellow at around 6pm
    cvInRangeS(imgHSV, cvScalar(112, 100, 100), cvScalar(124, 255, 255),
imgThreshed);

    cvReleaseImage(&imgHSV);

    return imgThreshed;
}


int main(int argc, char* argv[])
{
    VideoCapture cap(0); // open the video camera no. 0

    if (!cap.isOpened())  // if not success, exit program
    {
        cout << "Cannot open the video cam" << endl;
        return -1;
    }

    double dWidth = cap.get(CV_CAP_PROP_FRAME_WIDTH); //get the width of frames
of the video
    double dHeight = cap.get(CV_CAP_PROP_FRAME_HEIGHT); //get the height of
frames of the video

    cout << "Frame size : " << dWidth << " x " << dHeight << endl;

    namedWindow("MyVideo", CV_WINDOW_AUTOSIZE); //create a window called
"MyVideo"

    while (1)
    {
        Mat frame;

        bool bSuccess = cap.read(frame); // read a new frame from video
```

```
        if (!bSuccess) //if not success, break loop
        {
            cout << "Cannot read a frame from video stream" << endl;
            break;
        }

        //-------------------------------

        src = frame;
        cvtColor(src, src_gray, CV_BGR2GRAY);
        blur(src_gray, src_gray, Size(3, 3));

        thresh_callback(0, 0);

        //-------------------------------

        imshow("MyVideo", src_gray); //show the frame in "MyVideo" window

        if (waitKey(30) == 27) //wait for 'esc' key press for 30ms. If 'esc'
key is pressed, break loop
        {
            cout << "esc key is pressed by user" << endl;
            break;
        }
    }
    return 0;

}

/** @function thresh_callback */
void thresh_callback(int, void*)
{
    Mat canny_output;
    vector<vector<Point> > contours;
    vector<Vec4i> hierarchy;

    /// Detectar bordes(edges) utilizando canny
    Canny(src_gray, canny_output, thresh, thresh * 2, 3);

    /// Encontrar contornos
    findContours(canny_output, contours, hierarchy, CV_RETR_TREE,
CV_CHAIN_APPROX_SIMPLE, Point(0, 0));

    /// Obtener momentos
    vector<Moments> mu(contours.size());
    for (int i = 0; i < contours.size(); i++)
    {
        mu[i] = moments(contours[i], false);    //para obtener las areas
    }

    /// Obtener centros de masa:
    vector<Point2f> mc(contours.size());
    for (int i = 0; i < contours.size(); i++)
    {
        mc[i] = Point2f(mu[i].m10 / mu[i].m00, mu[i].m01 / mu[i].m00);  //para
obtener las posiciones
    }

    /// Dibujar contornos
```

```
    Mat drawing = Mat::zeros(canny_output.size(), CV_8UC3);


    cv::Mat labels = cv::Mat::zeros(src.size(), CV_8UC1);
    std::vector<Scalar> cont_avgs(contours.size(), 0.f); // This contains the
averages of each contour

    /// ver en ventana
    for (size_t i = 0; i < contours.size(); ++i)
    {
        //Mat mask = Mat::zeros(src.rows, src.cols, CV_8U); and then
drawContours(mask, contours, idx, color, CV_FILLED, 8);
        cv::drawContours(labels, contours, i, cv::Scalar(i), CV_FILLED);
        cv::Rect roi = cv::boundingRect(contours[i]);
        cv::Scalar mean = cv::mean(src(roi), labels(roi) == i);
        cont_avgs[i] = mean;    //para los colores
    }
    //-------------------------------

    //namedWindow( "DetectorQuinua", CV_WINDOW_AUTOSIZE );
    //imshow( "DetectorQuinua", drawing );

    /// Calculate the area with the moments 00 and compare with the result of
the OpenCV function
    printf("\t Info: Area y Longitud de Contorno \n");
    for (int i = 0; i< contours.size(); i++)
    {
        //printf(" * Contorno[%d] - Area (M_00) = %.2f - Area OpenCV: %.2f -
Length: %.2f \n", i, mu[i].m00, contourArea(contours[i]), arcLength(
contours[i], true ) );
        printf(" * Contorno[%d] - Area: %.2f - Posicion(x,y):(%.2f , %.2f) -
Color:(%.2f,%.2f,%.2f,%.2f)\n", i, mu[i].m00, contourArea(contours[i]),
mc[i].x, mc[i].y, cont_avgs[i][0], cont_avgs[i][1], cont_avgs[i][2],
cont_avgs[i][3]);
        //Scalar color = Scalar( rng.uniform(0, 255), rng.uniform(0,255),
rng.uniform(0,255) );
        drawContours(drawing, contours, i, (255, 255, 255), 1, 8, hierarchy, 0,
Point());
        circle(drawing, mc[i], 4, cont_avgs[i], -1, 8, 0);
    }

    namedWindow("DetectorQuinua", CV_WINDOW_AUTOSIZE);
    imshow("DetectorQuinua", drawing);

}
```

Anexo 03

Código fuente del Algoritmo de Comunicación por Sockets.

```
void sendData( int sockfd, int x ) {
  int n;

  char buffer[32];
  sprintf( buffer, "%d\n", x );
  if ( (n = write( sockfd, buffer, strlen(buffer) ) ) < 0 )
    error( const_cast<char *>( "ERROR al escribir en socket") );
  buffer[n] = '\0';
}

int getData( int sockfd ) {
  char buffer[32];
  int n;

  if ( (n = read(sockfd,buffer,31) ) < 0 )
    error( const_cast<char *>( "ERROR leer desde socket") );
  buffer[n] = '\0';
  return atoi( buffer );
}

int monitoreo(){
     int sockfd, newsockfd, portno = 51717, clilen;
     char buffer[256];
     struct sockaddr_inserv_addr, cli_addr;
     int n;
     int data;

     printf( "usando el puerto #%d\n", portno );

     sockfd = socket(AF_INET, SOCK_STREAM, 0);
     if (sockfd < 0)
         error( const_cast<char *>("ERROR al abrir socket") );
     bzero((char *) &serv_addr, sizeof(serv_addr));

     serv_addr.sin_family = AF_INET;
     serv_addr.sin_addr.s_addr = INADDR_ANY;
     serv_addr.sin_port = htons( portno );
     if (bind(sockfd, (struct sockaddr *) &serv_addr,
              sizeof(serv_addr)) < 0)
       error( const_cast<char *>( "ERROR al enmascarar" ) );
     listen(sockfd,5);
     clilen = sizeof(cli_addr);

     while ( 1 ) {
        printf( "Esperandocliente...\n" );
        if ( ( newsockfd = accept( sockfd, (struct sockaddr *) &cli_addr,
(socklen_t*) &clilen) ) < 0 )
            error( const_cast<char *>("ERROR al aceptar") );
        printf( "Comunicaci□ierta..\n" );
        while ( 1 ) {
             data = getData( newsockfd );
             printf( "got %d\n", data );
             if ( data < 0 )
                break;
```

```
            data = func( data );

            printf( "devolviendo %d\n", data );
            sendData( newsockfd, data );
         }
         close( newsockfd );

         if ( data == -2 )
           break;
      }
      return 0;
}
```

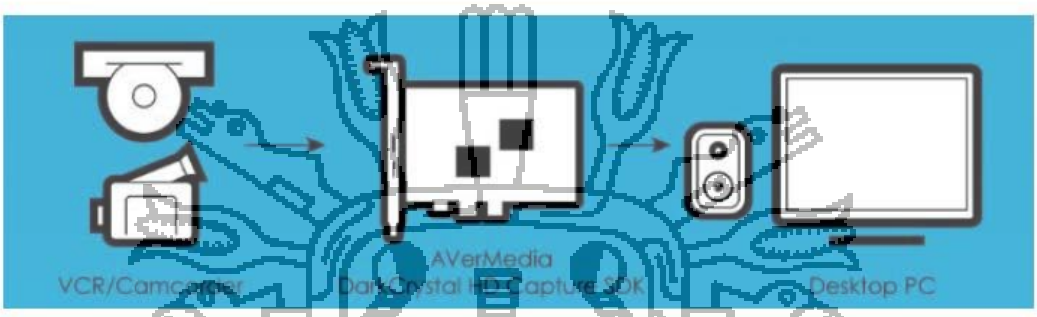
Anexo 04

Tarjeta Capturadora Avermedia Darkcrystal Hd Capture Sdk Hdmi C727

AVerMedia

Digital Entertainment Everywhere

DarkCrystal HD Capture SDK



Work with Popular Video Software

Compatible with **DirectShow**, AVerMedia DarkCrystal HD Capture SDK supports various popular professional video editing software ! The **Adobe Premiere Pro CS4** support also makes it a best solution for professionals. You can capture/edit your own video directly on your familiar software without learning.

Package Included

- AVerMedia DarkCrystal HD Capture SDK (Weight: 45.8g)
- Installation CD (Driver & Applications)
- Quick Installation Guide
- Dongle Cable
- Low Profile Bracket

Specifications

Input Signal :

- S-Video
- Composite (RCA)
- Component (YPbPr, 1080i/720p)
- HDMI (HDCP support*, 1080i/720p)
- Analog Audio L/R

*Preview HDCP content on AVer MediaCenter

System Requirements

- For SD Video Capturing
  - Intel® Pentium® 4 3.0GHz
  - AMD Athlon™ 64 3200+
- For HD Video Real-time Capturing (MPEG-2 Format)
  - Intel® Core™2 Duo 2.4GHz
  - AMD Athlon™ 64x2 Dual Core 2.8GHz
- VGA card with DirectX9.0c or above*
- 512 MB RAM**
- PCI-E Slot
- Sound Card
- Windows® 7/Vista™/XP (32/64-Bit)

*Standalone graphics card is recommended for HD video capturing.

**2GB RAM is recommended for HD video capturing.

※ Please visit AVerMedia official website for more detail about SDK and the latest driver at http://www.avermedia.com

※ Please seek customer service in the country / region of purchase if needed.

AVerMedia's Quality Insistence

AVerMedia's products are not only commit to RoHS/WEEE Directive, but conduct EMI and safety countermeasure to ensure consumers' health and environment protection. All products are available with lead-free finishes and flame retardant.

RoHS CE AVerMedia

Anexo 05

Especificaciones CAMARA Nikon D5200

Nikon Digital SLR Camera D5200 Specifications

Type of camera	Single-lens reflex digital camera
Lens mount	Nikon F mount (with AF contacts)
Effective angle of view	Nikon DX format; focal length equivalent to approx. 1.5x that of lenses with FX-format angle of view
Effective pixels	24.1 million
Image sensor	23.5 × 15.6 mm CMOS sensor
Total pixels	24.71 million
Dust-reduction System	Image Sensor Cleaning, Airflow Control System, Image Dust Off reference data (optional Capture NX 2 software required)
Image size (pixels)	6000 × 4000 (L), 4496 × 3000 (M), 2992 × 2000 (S)
File format	• NEF (RAW): 14 bit, compressed • JPEG: JPEG-Baseline compliant with fine (approx. 1:4), normal (approx. 1:8) or basic (approx. 1:16) compression • NEF (RAW) + JPEG: Single photograph recorded in both NEF (RAW) and JPEG formats
Picture Control System	Standard, Neutral, Vivid, Monochrome, Portrait, Landscape; selected Picture Control can be modified; storage for custom Picture Controls
Storage media	SD (Secure Digital) and UHS-I compliant SDHC and SDXC memory cards
File system	DCF (Design Rule for Camera File System) 2.0; DPOF (Digital Print Order Format); Exif (Exchangeable Image File Format for Digital Still Cameras) 2.3; PictBridge
Viewfinder	Eye-level pentamirror single-lens reflex viewfinder
Frame coverage	Approx. 95% horizontal and 95% vertical
Magnification	Approx. 0.78× (50 mm f/1.4 lens at infinity, -1.0 m⁻¹)
Eyepoint	17.9 mm (-1.0 m⁻¹ from center surface of viewfinder eyepiece lens)
Diopter adjustment	-1.7 to + 0.7 m⁻¹
Focusing screen	Type B BriteView Clear Matte Mark VII screen
Reflex mirror	Quick return
Lens aperture	Instant return, electronically controlled
Compatible lenses	Autofocus is available with AF-S and AF-I lenses; autofocus is not available with other type G and D lenses. AF lenses (IX NIKKOR and lenses for the F3AF) are not supported, and AI-P lenses; non-CPU lenses can be used in mode M, but the camera exposure meter will not function. The electronic rangefinder can be used with lenses that have a maximum aperture of f/5.6 or faster
Shutter type	Electronically controlled vertical-travel focal-plane shutter
Shutter speed	1/4000 to 30 s in steps of 1/3 or 1/2 EV, bulb, time (requires optional ML-L3 Remote Control)
Flash sync speed	X=1/200 s; synchronizes with shutter at 1/200 s or slower
Release modes	□ (single frame), □L (continuous low speed), □H (continuous high speed), ⏲ (self-timer), ⏲ 2s (delayed remote), ML-L3, ⏲ (quick-response remote; ML-L3), ⏲ (quiet shutter release), interval timer photography supported
Frame advance rate	Up to 3 fps (□L) or 5 fps (□H)
Self-timer	2 s, 5 s, 10 s, 20 s; 1 to 9 exposures
Exposure metering mode	TTL exposure metering using 2016-pixel RGB sensor
Metering method	• Matrix: 3D color matrix metering II (type G and D lenses); color matrix metering II (other CPU lenses) • Center-weighted: Weight of 75% given to 8-mm circle in center of frame • Spot: Meters 3.5-mm circle (about 2.5% of frame) centered on selected focus point
Metering range (ISO 100, f/1.4 lens, 20°C/68°F)	• Matrix or center-weighted metering: 0 to 20 EV • Spot metering: 2 to 20 EV
Exposure meter coupling	CPU
Exposure modes	Auto modes (📷 auto; 📷 auto (flash off)); programmed auto with flexible program (P); shutter-priority auto (S); aperture-priority auto (A); manual (M); scene modes (🧍 portrait; 🏞 landscape; 👶 child; 🏃 sports; 🌷 close up; 🌙 night portrait; 🌃 night landscape; 🎉 party/indoor; 🏖 beach/snow; 🌅 sunset; 🌆 dusk/dawn; 🐕 pet portrait; 🕯 candlelight; 🌸 blossom; 🍁 autumn colors; 🍽 food); special effects modes (🌃 night vision; 🎨 color sketch; 🏙 miniature effect; 🎯 selective color; 👤 silhouette; ▣ high key; ▣ low key)
Exposure compensation	Can be adjusted by -5 to +5 EV in increments of 1/3 or 1/2 EV in P, S, A and M modes
Exposure bracketing	3 shots in steps of 1/3 or 1/2 EV
Exposure lock	Luminosity locked at detected value with AE-L/AF-L button
ISO sensitivity (Recommended Exposure Index)	ISO 100 to 6400 in steps of 1/3 EV; can also be set to approx. 0.3, 0.7, 1 or 2 EV (ISO 25600 equivalent) above ISO 6400; auto ISO sensitivity control available
Active D-Lighting	Auto, extra high, high, normal, low, off
ADL bracketing	2 shots
Autofocus	Nikon Multi-CAM 4800DX autofocus sensor module with TTL phase detection, 39 focus points (including 9 cross-type sensors), and AF-assist illuminator (range approx. 0.5 to 3 m/1 ft 8 in. to 9 ft 10 in.)
Detection range	-1 to +19 EV (ISO 100, 20°C/68°F)
Lens servo	• Autofocus (AF): Single-servo AF (AF-S), continuous-servo AF (AF-C), or auto AF-S/AF-C selection (AF-A); predictive focus tracking activated automatically according to subject status • Manual focus (MF): Electronic rangefinder can be used
Focus point	Can be selected from 39 or 11 focus points
AF-area modes	Single-point AF, 9-, 21-, or 39-point dynamic-area AF, 3D-tracking, auto-area AF
Focus lock	Focus can be locked by pressing shutter-release button halfway (single-servo AF) or by pressing AE-L/AF-L button
Built-in flash	📷, 🧍, 🏞, 🌷, 🌃, 🎉, 🏃, 🌙: Auto flash with auto pop-up; P, S, A, M, ⚡: Manual pop-up with button release
Guide number	Approx. 12/39, 13/43 with manual flash (m/ft, ISO 100, 20°C/68°F)
Flash control	TTL: i-TTL flash control using 2016-pixel RGB sensor is available with built-in flash and SB-910, SB-900, SB-800, SB-700, SB-600 or SB-400; i-TTL balanced fill-flash for digital SLR is used with matrix and center-weighted metering, standard i-TTL flash for digital SLR with spot metering
Flash modes	Auto, auto with red-eye reduction, auto slow sync, auto slow sync with red-eye reduction, fill-flash, red-eye reduction, slow sync, slow sync with red-eye reduction, rear-curtain with slow sync, rear-curtain sync, off
Flash compensation	-3 to +1 EV in increments of 1/3 or 1/2 EV
Flash-ready indicator	Lights when built-in flash or optional flash unit is fully charged; flashes after flash is fired at full output
Accessory shoe	ISO 518 hot-shoe with sync and data contacts and safety lock
Nikon Creative Lighting System (CLS)	Advanced Wireless Lighting supported with SB-910, SB-900, SB-800 or SB-700 as a master flash or SU-800 as commander; Flash Color Information Communication supported with all CLS-compatible flash units
Sync terminal	AS-15 Sync Terminal Adapter (available separately)
White balance	Auto, incandescent, fluorescent (7 types), direct sunlight, flash, cloudy, shade, preset manual, all except preset manual with fine-tuning
White balance bracketing	3 shots in steps of 1
Live view lens servo	• Autofocus (AF): Single-servo AF (AF-S); full-time-servo AF (AF-F) • Manual focus (MF)
AF-area modes	Face-priority AF, wide-area AF, normal-area AF, subject-tracking AF
Autofocus	Contrast-detect AF anywhere in frame (camera selects focus point automatically when face-priority AF or subject-tracking AF is selected)
Automatic scene selection	Available in 📷 and 🅂 modes
Movie metering	TTL exposure metering using main image sensor
Metering method	Matrix
Frame size (pixels) and frame rate	• 1920 × 1080, 60i (50 fields/s)*, 50i (50 fields/s)*, ★ high/normal • 1920 × 1080, 30p (progressive)/25p/24p, ★ high/normal • 1280 × 720, 60p/50p, ★ high/normal • 640 × 424, 30p/25p, ★ high/normal Frame rates of 30p (actual frame rate 29.97 fps), 60i, and 60p (actual frame rate 59.94 fps) are available when NTSC is selected for video mode; 25p, 50i, and 50p are available when PAL is selected for video mode; actual frame rate when 24p is selected is 23.976 fps *Sensor output is about 60 or 50 fps Note: A smaller crop is used for movies with a frame size/frame rate of 1920 × 1080 60i or 50i
File format	MOV
Video compression	H.264/MPEG-4 Advanced Video Coding
Audio recording format	Linear PCM
Audio recording device	Built-in or external stereo microphone; sensitivity adjustable
Maximum length	29 min. 59 s (3 min. in miniature effect mode)
ISO sensitivity	ISO 100 to 6400; can also be set to approx. 0.3, 0.7, 1, or 2 EV (ISO 25600 equivalent) above ISO 6400
Monitor	7.5-cm (3-in.), approx. 921k-dot (VGA), vari-angle TFT monitor with 170° viewing angle, approx. 100% frame coverage, and brightness adjustment
Playback	Full-frame and thumbnail (4, 9, or 72 images or calendar) playback with playback zoom, movie playback, photo and/or movie slide shows, histogram display, highlights, auto image rotation, and image comment (up to 36 characters)
USB	Hi-Speed USB
Video output	NTSC, PAL
HDMI output	Type C mini-pin HDMI connector
Accessory terminal	Wireless remote controller: WR-R10 (available separately) Remote cord: MC-DC2 (available separately) GPS unit: GP-1 (available separately)
Audio input	Stereo mini-pin jack (3.5-mm diameter)
Supported languages	Arabic, Chinese (Simplified and Traditional), Czech, Danish, Dutch, English, Finnish, French, German, Greek, Hindi, Hungarian, Indonesian, Italian, Japanese, Korean, Norwegian, Polish, Portuguese (Portugal and Brazil), Romanian, Russian, Spanish, Swedish, Thai, Turkish, Ukrainian
Battery	One EN-EL14 Rechargeable Li-ion Battery
AC adapter	EH-5b AC Adapter, requires EP-5A Power Connector (available separately)
Tripod socket	1/4 in. (ISO 1222)
Dimensions (W × H × D)	Approx. 129 × 98 × 78 mm/5.1 × 3.9 × 3.1 in.
Weight	Approx. 555 g/1 lb 3.6 oz with battery and memory card but without body cap; approx. 505 g/1 lb 1.8 oz (camera body only)
Operating environment	Temperature: 0 to 40°C/32 to 104°F; humidity: 85% or less (no condensation)
Supported accessories (may differ by country or area)	EN-EL14 Rechargeable Li-ion Battery, MH-24 Battery Charger, DK-5 Eyepiece Cap, DK-20 Rubber Eyecup, UC-E17 USB Cable, EG-CP16 Audio Video Cable, AN-DC3 Camera Strap, BF-1B Body Cap, BS-1 Accessory Shoe Cover, ViewNX 2 CD-ROM

• The SD, SDHC and SDXC logos are trademark of the SD Card Association. • PictBridge is a trademark. • HDMI, the HDMI logo and High-Definition Multimedia Interface are trademarks or registered trademarks of HDMI Licensing, LLC. • Google and Android™ are registered trademarks or trademarks of Google Inc. • Products and brand names are trademarks or registered trademarks of their respective companies. • Images in viewfinders or LCDs and monitors shown in this brochure are simulated.

Specifications and equipment are subject to change without any notice or obligation on the part of the manufacturer. December 2012

WARNING **TO ENSURE CORRECT USAGE, READ MANUALS CAREFULLY BEFORE USING YOUR EQUIPMENT. SOME DOCUMENTATION IS SUPPLIED ON CD-ROM ONLY.**