Universidad de Castilla-La Mancha

Escuela Superior de Informática

Departamento de Tecnologías y Sistemas de Información
Programa Oficial de Postgrado en Tecnologías Informáticas Avanzadas

Trabajo Fin de Máster

Título:

Framework para el Desarrollo de Interfaces Naturales de Usuario basadas en Visión por Computador y Realidad Aumentada

Septiembre de 2015

Alumno: Manuel Hervás Ortega

Director: Dr. Carlos González Morcillo

Trabajo Fin de Máster
(Máster en Tecnologías Informáticas Avanzadas)

Framework para el Desarrollo de Interfaces Naturales de Usuario basadas en Visión por Computador y Realidad Aumentada

Septiembre de 2015

Autor
Manuel Hervás Ortega
(Graduado en Ingeniería Informática)

Director
Dr. Carlos González Morcillo

Este documento fue maquetado con LaTeX. Imágenes compuestas con *Gimp* y *LibreOffice*.

Resumen

Las Interfaces Naturales de Usuario surgen como consecuencia de la evolución en la forma que tenemos para interaccionar con un computador. Su objetivo es lograr una tecnología lo más natural e intuitiva posible, de modo que el propio uso del ordenador pase desapercibido.

El potencial de la Realidad Aumentada para construir interfaces naturales es evidente. Su uso va más allá de juegos y entretenimiento; las grandes compañías informáticas están presentando novedosos sistemas que explotan sus características. Aunque actualmente la mayoría de aplicaciones utilizan pantallas o gafas como dispositivos de representación, la utilización de proyectores permite desarrollar innovadoras «*ventanas de información*» sobre cualquier superficie o dispositivo.

La aplicación de técnicas de Realidad Aumentada en el ámbito de los Interfaces Naturales permite modelar entornos interactivos que proporcionen una sensación inmersiva al usuario sobre elementos existentes del entorno físico.

El presente Trabajo Fin de Máster surge como parte del *Proyecto ARgos*, cuyo objetivo es el uso de técnicas de Realidad Aumentada como ayuda a la gestión de documentos impresos. El proyecto plantea la construcción de un dispositivo embebido que dé soporte a la detección de documentos y su posterior tratamiento empleando técnicas de realidad aumentada mediante un entorno natural e interactivo.

El framework se encarga de la detección e identificación de documentos, así como del cálculo de su posicionamiento dentro del espacio 3D. El sistema emplea el paradigma de «*pantalla táctil*», utilizando el propio documento impreso como superficie interactiva. Se ha elaborado una herramienta genérica que permite tanto el calibrado de cámaras y proyectores en el espacio 3D, como la representación de información aumentada directamente sobre el espacio físico. Además se ha construido un prototipo hardware sobre el que se han definido diferentes escenarios de explotación por la Asociación ASPRONA (Asociación para la Atención a Personas con Discapacidad Intelectual y sus Familias de la Provincia de Albacete).

La arquitectura implementada y los resultados obtenidos se describen en el presente documento.

Abstract

Natural User Interfaces arise from changes in the way that we have to interact with a computer. Its aim is to achieve the most natural and intuitive technology possible, so that the use of the computer itself goes unnoticed.

The potential of Augmented Reality to build natural interfaces is evident. Its use goes beyond gaming and entertainment; main computer companies are introducing new systems that exploit its features. Although most applications currently used screens or glass display devices, the use of projectors allows developing innovative «information windows» on any surface or device.

Application of Augmented Reality techniques in the field of Natural Interfaces allows modeling interactive environments that provide an immersive sensation to the user on existing elements of the physical environment.

This Master's thesis comes as part of the *ARgos Project* whose goal is the use of Augmented Reality techniques as an aid to document management. The project involves the construction of an embedded device that supports document image recognition and its management using augmented reality techniques through a natural and interactive environment.

The framework is responsible for the detection and identification of documents and calculating their position within the 3D space. The system uses the paradigm of «touch screen», using the printed document as an interactive surface. It has developed a generic tool that allows calibration of cameras and projectors in 3D space, such as increased representation of information directly on the physical space. It has also built a prototype hardware on which they have defined different operational scenarios for ASPRONA Association (Association for the Care of Persons with Intellectual Disabilities and their Families of the Province of Albacete).

Índice general

Índice de figuras

1

Introducción

1.1. Planteamiento general

Desde la introducción del ordenador personal en la década de los años 80, los esfuerzos por naturalizar la experiencia de la interacción ha estado limitada por el factor de adaptación de una persona a los distintos entornos informáticos. A día de hoy, todavía conservamos el mismo esquema de utilización de un ordenador de sobremesa de principios de los 80, con computadores que tienen prácticamente la misma configuración de monitor, teclado y ratón. Realmente no hay nada de natural en este tipo de experiencia de usuario, y en todo caso es lo más alejado de ser una función humana. Para apoyar aún más lo lejos que nos hemos desviado de esta naturaleza, el estudio de la ergonomía surgió como una manera de minimizar en nuestros cuerpos el riesgo de una lesión, ya que tratamos de adaptarlos a este entorno no natural.

No es hasta finales de los años 2000, cuando podemos decir que empieza la ***«era post-PC»***. La generalización de los dispositivos multi-touch y la informática móvil marca uno de los mayores hitos en las interfaces persona-computador. Los dispositivos móviles se diseñan para ser ligeros, portátiles y encajar con nuestro estilo de vida personal. De repente, estamos en medio de una de las más importantes revoluciones tecnológicas, cuando nuestros dispositivos informáticos comienzan a adaptarse a nuestras funciones naturales humanas.

Apenas estamos comenzando a descubrir las posibilidades de un mundo en el que los dispositivos se adaptan a nuestras conductas y la tecnología apoya y amplifica nuestras funciones naturales. Estaremos en el camino de lo que podemos llamar ***«computación humana»***, cuando el hardware desaparezca y nos pase desapercibido que realmente estamos interaccionado con un ordenador. Esto ocurre cuando la tecnología se integra discretamente en objetos cotidianos o en funciones naturales. La tecnología será la que se adapte a nosotros, en lugar de ser nosotros quienes se adapten a la computación y la máquina aprenda a reconocer e interpretar los patrones humanos para producir una salida basada en un contexto familiar.

Interfaz natural de usuario es un término genérico para una variedad de tecnologías que permiten a los usuarios interactuar con los ordenadores en términos humanos. Algunas de estas tecnologías son las basadas en visión por computador y que son capaces desde interpretar expresiones naturales como gestos hasta proporcionar información contextual que se proyecta dentro del campo de visión del usuario, como pretenden las HoloLens.

La **realidad aumentada** se podría considerar como la aplicación de distintas técnicas de visión por computador, mediante la cual la percepción del mundo real se complementa con información adicional generada por ordenador en tiempo real. Esta información adicional puede ser desde etiquetas virtuales, representaciones de modelos tridimensionales, o incluso cambios de iluminación.

Hoy en día, la mayoría de las aplicaciones de realidad aumentada se centran en la movilidad. De este modo, las pantallas de dispositivos portátiles y los smartphones se han convertido en la opción dominantes para la visualización. Sin embargo, el aumento de las prestaciones y la reducción de los costes hacen que los proyectores se hayan popularizado y establecido como herramientas habituales para la visualización. La capacidad de generar imágenes mucho mayores que el propio dispositivo prácticamente en cualquier lugar es una característica interesante para muchas aplicaciones que no pueden ser mostradas en pantallas convencionales.

Los **enfoques basados en proyectores** combinan las ventajas de la realidad virtual y la realidad aumentada proporcionando sensaciones inmersivas que se pueden realizar sobre entornos cotidianos, sin la necesidad de pantallas de proyección especiales y configuraciones de pantalla dedicadas. Para muchas aplicaciones, esto requiere la perdida de la movilidad, pero no necesariamente de la portabilidad. Otras aplicaciones, sin embargo, no requieren movilidad y más bien se benefician de las propiedades aumentadas que proporciona la proyección. Los ejemplos van desde entretenimiento educativo en los museos, con proyecciones sobre paredes o sobre las propias obras de arte, hasta proyecciones en fachadas de edificios históricos para conseguir efectos de movimiento ó 3D, dando lugar a un espectáculo artístico conocido como ***projection mapping***.

1.2. Objetivo del Trabajo de Fin de Máster

El objetivo del presente Trabajo Fin de Máster es el desarrollo de un framework para desarrollar interfaces naturales de usuario basado en el uso de Realidad Aumentada y Visión Artificial.

El objetivo general es construir sistema que utilice una cámara de bajo coste como entrada al módulo de visión por computador y un cañón de proyección portátil para mostrar información visual, directamente alineada sobre un documento del mundo físico. Responderá a las peticiones que el usuario realice sobre el espacio físico, ampliando información relacionada que sea relevante a la acción que quiera realizar. El documento podrá moverse dentro de una región del escritorio y la amplificación deberá quedar perfectamente alineada en el espacio físico. Como soporte hardware, se utilizará un computador en placa *Raspberry Pi* con arquitectura *ARM*.

Deberemos proveer al sistema de un módulo para obtener las imágenes y aplicarle el procesado previo necesario, como puede ser el escalado, umbralización, detección de bordes o detección de características [24] [4]. Otra tarea a realizar es calcular la distorsión debida a la proyección en perspectiva mediante los parámetros extrínsecos e intrínsecos de la cámara.

Contará con un sistema de identificación rápida empleando algoritmos de recuperación de imágenes y comparará el documento que está siendo analizado con una base de datos de documentos conocidos por el sistema.

Para el correcto alineado de la información mostrada, el módulo de tracking y registro contará con funciones de cálculo de *pose* (rotación y translación del objeto en el espacio 3D) en tiempo real y algoritmos para la estimación y descripción del movimiento como *Optical Flow* [19].

El usuario podrá interactuar directamente en el espacio físico sin utilizar sistemas de mando o dispositivos de entrada tradicionales como sería un ratón, teclado, etc. siendo sustituidos por funciones más naturales como el uso de movimientos gestuales con las manos.

Contará con diferentes modos de amplificación de la información del mundo real. Por un lado, la información visual se amplificará empleando el cañón de proyección que mostrará información relevante al contexto directamente sobre el espacio del papel, así como otras fuentes de información visual adicionales.

Para facilitar la implantación real en el entorno de trabajo, deberá funcionar con componentes de bajo coste, incorporando mecanismos de corrección de distorsión y registro 3D totalmente software.

El desarrollo se realizará siguiendo estándares, tecnologías y bibliotecas libres multiplataforma, con el objetivo de que pueda ser utilizado en el mayor número de plataformas posibles tanto hardware (x86, x86-64 y ARM) como software (GNU/Linux, Windows y Mac).

1.3. Marco de trabajo: el Proyecto ARGOS

El presente proyecto se enmarca dentro de la Cátedra Indra-UCLM, en el proyecto ***«ARgos: Sistema de Ayuda a la Gestión de Documentos Impresos basado en Visión por Computador y Realidad Aumentada»*** que tiene como objetivo la construcción de un sistema de ayuda a la gestión de documentos, basado principalmente en visión por computador y síntesis visual y auditiva en el espacio físico, empleando técnicas de realidad aumentada.

Todos los países de la Unión Europea aceptan las recomendaciones generales de la Organización Mundial de la Salud así como las directrices y programas de las Naciones Unidas relativas a las personas con necesidades especiales y que proponen expresamente *«su participación plena en la vida social, con oportunidades iguales a las del resto»*. El proyecto esta pensado para facilitar la integración laboral, sean cuales sean las necesidades especiales de las personas que tenga que gestionar documentación impresa.

1.4. Estructura del documento

El presente documento se estructura de la siguiente forma:

- **Capítulo 2:** Presenta una visión general del estado de las disciplinas y tecnologías que ha sido necesario investigar para el desarrollo de este trabajo.

  Dentro de dichas disciplinas se encuentra la geometría de formación de imágenes, diferentes técnicas de Realidad Aumentada, detección de marcadores y puntos de interés y *optical flow*, o los algoritmos para la identificiación de imágenes de documentos mediante el uso de visión por computador.

- **Capítulo 3:** Describe la arquitectura propuesta para el desarrollo de intefaces naturales de usuario basadas en realidad aumentada espacial.

- **Capítulo 4:** Muestra un apartado de conclusiones evaluando el aprovechamiento de la investigación y las líneas de trabajo futuro propuestas. Muestra un análisis del trabajo realizado y los objetivos conseguidos. Incluye una descripción de las líneas de trabajo futuro sobre el proyecto y una valoración personal del trabajo realizado.

- **Anexo A:** Resumen de todas las asignaturas cursadas por el alumno durante el máster, junto a una conlusión personal sobre qué han aportado cada una a la labor investigadora y a la experiencia del alumno.

- **Anexo B:** *Currículum Vítae* en español actualizado a Septiembre de 2015 del alumno Manuel Hervás Ortega.

2

Estado del arte

En este capítulo se introducirán los campos y las tecnologías relacionadas con este proyecto, realizando una revisión de las mismas. Se realizará un estudio del estado del arte de los sistemas existentes.

2.1. Geometría de la formación de imágenes

La formación de las imágenes consiste en una representación bidimensional del mundo 3D, perdiéndose la información de profundidad.

La óptica geométrica clásica se basa en modelos de lentes para modelar el proceso de formación de las imágenes. Sin embargo, podemos simplificar este proceso suponiendo que todos los rayos que llegan a la cámara atraviesan un único punto y se proyectan en un plano. Este modelo se conoce como modelo ***pinhole***.

Debido a que las lentes no tienen un comportamiento ideal, habrá que añadir al modelo unos parámetros de distorsión que permitan corregirlo y aproximarlo al comportamiento real de la cámara.

2.1.1. Estructura del modelo *pinhole*

El modelo ***pinhole*** [13] permite modelar el proceso de formación de las imágenes mediante una proyección central, en la cual, de cada punto del espacio tridimensional sale un rayo de luz que pasa por un punto fijo del espacio y intersecta en un plano dando lugar a la imagen.

Los elementos que forman este modelo se definen de la siguiente forma:

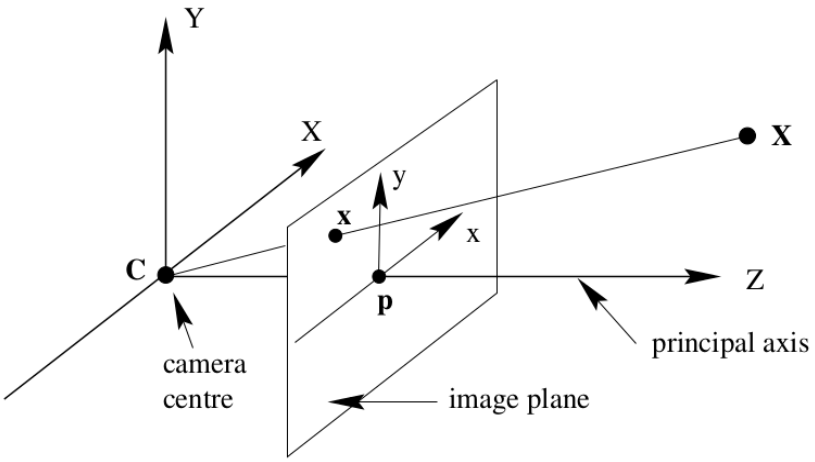


Figura 2.1: Modelo de cámara *pinhole*. La imagen de un punto 3D se forma mediante proyección de perspectiva. Un punto X es proyectado a un punto x en el plano imagen

- El **centro óptico** es el punto fijo del espacio por donde pasan todos los rayos de luz. Se corresponde con el centro de la cámara y es donde se fija el sistema de referencia de la cámara.
- El **plano imagen** o plano focal. De cada punto del espacio parte un rayo de luz que pasa por el centro de proyección e intersecta con este plano formando la imagen. Como se puede ver en la figura, el plano focal se ha situado delante del centro óptico. Si éste estuviese detrás, las imágenes estarían invertidas.
- **Distancia focal.** Se define como la distancia entre el centro de proyección y el plano imagen.
- **Eje principal** Es la línea que pasa por el centro de proyección y es perpendicular al plano imagen.
- **Punto principal.** Es el punto de intersección del eje principal con el plano imagen. Coincide con el centro de la imagen.
- **Plano principal.** Es el plano paralelo al plano imagen y que contiene al centro de proyección. Además, este plano está formado por todos los puntos cuyas proyecciones se corresponden con puntos en el infinito en la imagen.

2.1.2. Matriz de proyección

El modelo *pinhole* [13] fija un sistema de coordenadas proyectivas en el centro óptico. El eje Z de este sistema coincide con el eje principal de la cámara. A partir de ahora nos referiremos a este sistema de coordenadas como sistema de referencia de la cámara. Además, el plano imagen se fija en el plano $Z = f$.

Utilizaremos $[X, Y, Z]$, para representar vectores fila. Por tanto, su traspuesta, $[X, Y, Z]^T$, será un vector columna.

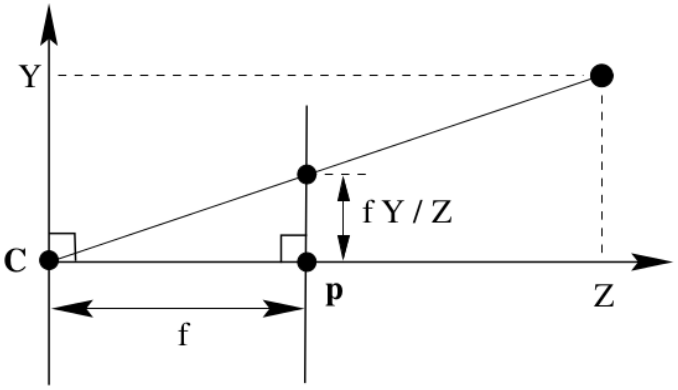


Figura 2.2: Modelo de cámara *pinhole* visto desde el eje X

Por convenio con la notación anterior, las coordenadas de un punto cualquiera se van a representar por un vector columna, definiendo los siguiente elementos a considerar:

- $[X_c, Y_c, Z_c]^T$ Coordenadas de un punto del espacio tridimensional respecto al sistema de referencia de la cámara.
- $[X_w, Y_w, Z_w]^T$ Coordenadas del mismo punto del espacio tridimensional respecto a un sistema de referencia asociado al modelo del objeto. Este nuevo sistema de referencia es distinto al de la cámara.
- $[u, v]^T$ Coordenadas del punto proyectado en el plano imagen.
- f Distancia focal

Mediante relación de semejanza de triángulos, de la figura 2.2:

$$\frac{u}{f} = \frac{X}{Z} \qquad \frac{v}{f} = \frac{Y}{Z} \tag{2.1}$$

podemos determinar la correspondencia entre un punto cualquiera del espacio y su proyección en el plano imagen:

$$[X_c, Y_c, Z_c]^T \Longrightarrow [u, v]^T = [f\frac{X_c}{Z}, f\frac{Y_c}{Z}]^T \tag{2.2}$$

Las **coordenadas homogéneas** son un instrumento empleado para describir un punto en el espacio proyectivo. Consiste en ampliar el plano euclídeo (en el caso bidimensional) al plano proyectivo, es decir, incluirle los puntos impropios o del infinito. De forma que, un punto de dimensiones $[x, y, z]$, se representa por el cuaternión: $[x/w,\ y/w,\ z/w, w]$ con $w = 1$.

Este sistema de coordenadas tiene la particularidad de que permite pasar fácilmente coordenadas de un número de dimensiones a otro. Para ello, almacena las coordenadas con una dimensión adicional, de tal forma que para un espacio de 3D, utilizaríamos 4 coordenadas. El valor de la coordenada adicional indica entre otras cosas, si el punto se encuentra en el infinito, $w = 0$, o es un punto cualquiera, $w \neq 0$. En este sistema, si dos coordenadas son proporcionales, se refieren al mismo punto.

Utilizándolas, podemos expresar un punto del espacio tridimensional respecto al sistema de referencia de la cámara de forma matricial:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} \Longrightarrow \begin{bmatrix} fX_c \\ fY_c \\ Z_c \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{2.3}$$

y la expresión de relación de correspondencia, queda de la siguiente forma:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{2.4}$$

Y de forma abreviada:

$$q = PQ \tag{2.5}$$

donde:

- $P = diag(f, f, 1)[I|0]$
- $diag(f, f, 1)$ es una matriz diagonal.
- $[I|0]$ representa una matriz identidad de 3x3 concatenada con un vector columna nulo de dimensión 3.
- q es el vector columna, de dimensión 3, que representan las coordenadas homogéneas del punto de la imagen.
- Q es el vector columna, de dimensión 4, que representa las coordenadas homogéneas del punto del espacio respecto al sistema de referencia de la cámara.
- A la matriz P de se la denomina **matriz homogénea de proyección de la cámara**.

2.1.3. Parámetros intrínsecos de la cámara

La matriz de proyección P de la ecuación 2.5 permite transformar las coordenadas de un punto 3D del mundo real en píxeles de una imagen. Se construye a partir de una matriz K y un vector de valores nulos:

$$P = [K|0] \tag{2.6}$$

donde la K es la **matriz de la cámara**, la cuál está formada por una serie de parámetros, denominados **parámetros intrínsecos**:

$$K = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.7}$$

Los **parámetros intrínsecos** son aquellos que describen las características ópticas y geométricas de una cámara y son constantes. Entre estos parámetros se encuentran la distancia focal, el centro óptico y el punto principal. En la figura 2.3 se pueden visualizar estos tres parámetros.

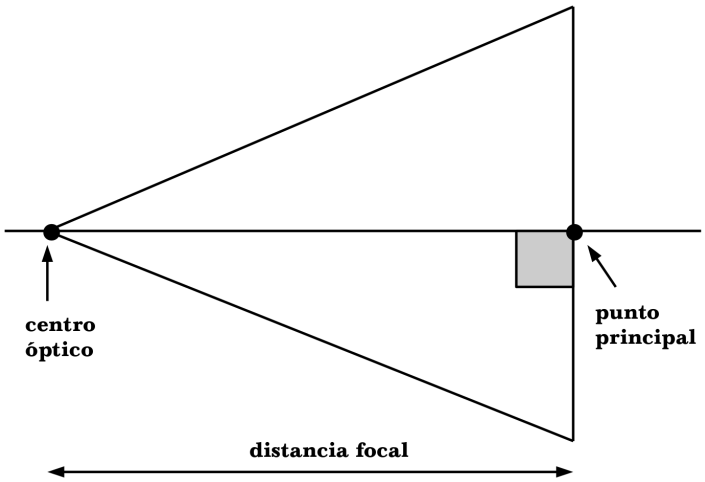


Figura 2.3: Parámetros intrínsecos de una cámara.

- El **centro óptico** o centro de proyección, es el punto de la cámara desde el cual parten todos los rayos que son proyectados en el plano de imagen de la cámara.
- El **punto principal**, denotado como ***c***, es la proyección ortogonal del centro óptico sobre el plano de cámara. c_x, c_y indican el desplazamiento del centro de coordenadas del plano imagen, respecto al punto principal. Será nulo sólo si el eje óptico coincide con el centro del sensor de la cámara, pero el eje óptico no siempre atraviesa el centro de la imagen generada.
- El **factor** γ **(*skew factor*)** determina el grado de perpendicularidad de las paredes de los píxeles del sensor. Es inversamente proporcional a la tangente del ángulo que forman los ejes X e Y, por lo que γ tendrá un valor nulo si los píxeles son rectangulares. Esto suele ser así en casi todos los sensores utilizados hoy en día.
- La **distancia focal**, es la distancia existente entre el centro óptico y el punto principal. f_x y f_y son dos distancias focales en píxeles. Son proporcionales a la longitud focal f considerada en las ecuaciones (2.1) y (2.2), según:

$$f_x = fS_x \qquad f_y = fS_y \tag{2.8}$$

  donde f es la longitud focal física de la lente, en unidades de longitud (milímetros, micras, etc.). S_x y S_y son el número de píxeles por unidad de longitud del sensor, a largo del eje X y del eje Y respectivamente. Como es obvio, si el sensor tiene el mismo número de píxeles por unidad de longitud en todas sus dimensiones, las dos focales f_x y f_y tendrán el mismo valor.

2.1.4. Distorsión

El uso de lentes facilita la entrada de luz, un enfoque adecuado y una mayor versatilidad, pero también introduce deformaciones en las imágenes que se forman en el sensor. Para el cálculo de coeficientes de distorsión, se tienen en cuenta factores radiales y tangenciales.

La **distorsión radial**, consiste en el desplazamiento de los píxeles de de la imagen, de tal modo que las líneas situadas en los extremos del encuadre aparentarán salir hacia el exterior o el interior.

Para la corrección de la distorsión radial de las coordenadas de un píxel de la imagen, se utiliza la siguiente fórmula:

$$\begin{aligned} x_{corrected} &= x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \\ y_{corrected} &= y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \end{aligned} \tag{2.9}$$

La **distorsión tangencial** se produce porque la lente no se encuentra perfectamente paralela al plano de imagen. Se puede corregir a través de las siguientes fórmulas:

$$\begin{aligned} x_{corrected} &= x + [2p_1 xy + p_2(r^2 + 2x^2)] \\ y_{corrected} &= y + [p_1(r^2 + 2y^2) + 2p_2 xy] \end{aligned} \tag{2.10}$$

De tal forma que tenemos cinco parámetros de distorsión que son representados como una matriz fila con 5 columnas:

$$Distortion_{coefficients} = [k_1 \quad k_2 \quad p_1 \quad p_2 \quad k_3] \tag{2.11}$$

2.1.5. Parámetros extrínsecos de la cámara

En la sección 2.1.2 se asumió el hecho de que el centro óptico era el origen de coordenadas del mundo. Esto era así a efectos del cálculo de parámetros intrínsecos, con lo cual el modelo era válido siempre que el sistema no fuera movido de su posición inicial.

Por otro lado, en la mayor parte de las aplicaciones prácticas es necesario que la cámara se mueva o se gire, para captar adecuadamente la escena. Por ello, para poder modelar el sistema con independencia de que su posición haya sido alterada o de que un objeto se pueda referenciar respecto al origen de coordenadas de la cámara, es necesario modificar la ecuación 2.5 introduciendo un matriz de transformación $[R|t]$ que contiene los **parámetros extrínsecos** de la cámara.

$$q = K[R|t]Q \tag{2.12}$$

$$[R|t] = \begin{bmatrix} R_{1,1} & R_{1,2} & R_{1,3} & T_1 \\ R_{2,1} & R_{2,2} & R_{2,3} & T_2 \\ R_{3,1} & R_{3,2} & R_{3,3} & T_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.13}$$

Donde:

- R es una **matriz de rotación**, que representa un giro de la cámara (o de un objeto respecto de ella). Tendrá una forma distinta dependiendo de respecto a que eje (X, Y, Z) se haga la rotación:

$$R_x(\Psi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\Psi & \sin\Psi \\ 0 & -\sin\Psi & \cos\Psi \end{bmatrix} \tag{2.14}$$

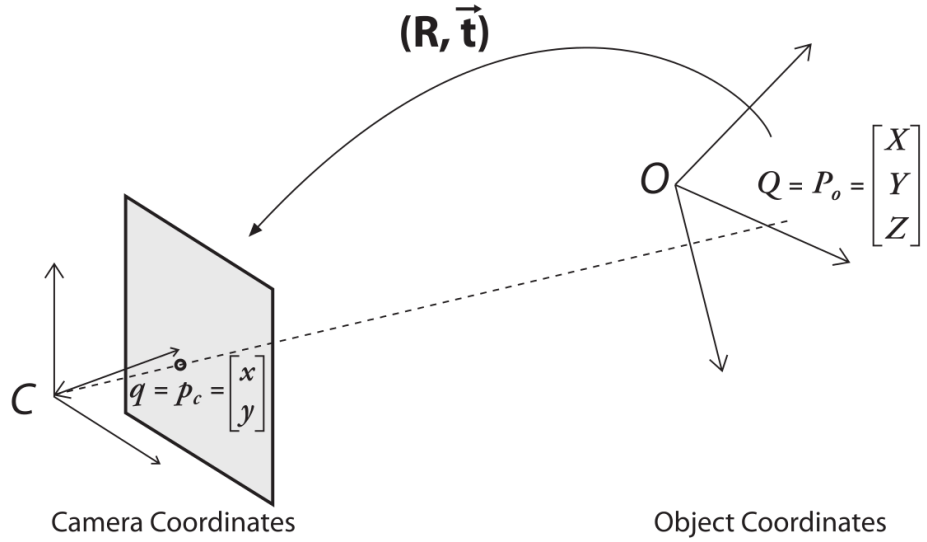


Figura 2.4: Conversión del sistemas de coordenadas del objeto al sistema de la cámara. El punto p_c está relacionado con el punto P_0 mediante la aplicación de una la matriz de rotación y el vector de traslación. (Bradski y Kaehler)

$$R_y(\varphi) = \begin{bmatrix} \cos\varphi & 0 & -\sin\varphi \\ 0 & 1 & 0 \\ \sin\varphi & 0 & \cos\varphi \end{bmatrix} \tag{2.15}$$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.16}$$

Si el giro se realiza respecto al eje Z, tal y como se muestra en la figura 2.5, las nuevas coordenadas quedarán de la siguiente forma:

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \Rightarrow \begin{array}{c} X' = X\cos\theta + Y\sin\theta \\ Y' = -X\sin\theta + Y\cos\theta \\ Z' = Z \end{array} \tag{2.17}$$

- t es un **vector de translación** que representa el cambio de un sistema de coordenadas a otro cuyo origen se desplaza a otra ubicación; en otras palabras, el vector de traslación es el desplazamiento desde el origen del sistema de coordenadas inicial hasta el origen del sistema de coordenadas final. En nuestro caso, como podemos ver en la figura 2.4, el sistema inicial pertenecería al del objeto, y el final, al sistema de coordenadas de la cámara.

$$t = origen_{objeto} - origen_{camara} \tag{2.18}$$

El cálculo aproximado de la matriz de transformación $[R|t]$ a partir de la información visual capturada se denomina ***pose estimation***. La solución para la estimación de la *pose* ha sido propuesta, entre otros métodos, mediante transformada lineal directa (DLT) ó algoritmos Pnp (perspective-n-point).

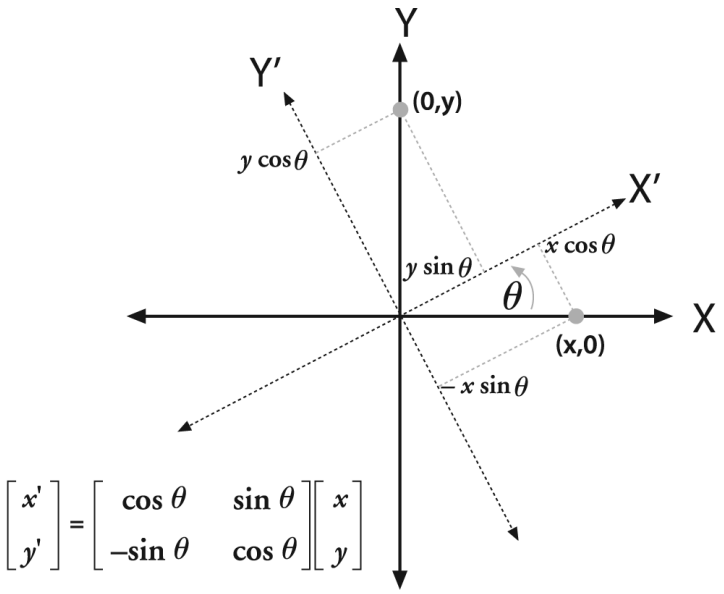


Figura 2.5: Rotación de puntos alrededor del eje Z (Bradski y Kaehler)

2.2. Realidad Aumentada

La Realidad Aumentada es el **conjunto de técnicas que permiten integrar en tiempo real contenido digital a la percepción del mundo real**. El término fue acuñado en 1990 por Tom Caudell durante el desarrollo de un sistema de Boeing para ayudar a los trabajadores en el ensamblaje de aviones mediante la ayuda de pantallas que mostraban información del montaje.

Uno de los principales problemas a resolver es denominado **registro**. Los objetos del mundo real y virtual deben estar correctamente alineados o la sensación de integración se verá seriamente afectada. Sin un registro preciso, la realidad aumentada no podría ser soportada por muchas aplicaciones, por ejemplo, en medicina para el guiado de una aguja en la realización de una biopsia.

Para distinguir que podemos llamar realidad aumentada y que no, Ronald T. Azuma [3], define una serie de características que deben cumplir las aplicaciones:

- **Combinar el mundo real con el virtual.** El resultado final debe mostrar la información sintética sobre las imágenes percibidas del mundo real.
- **Debe ser interactivo en tiempo real.** La integración debe ser realizada *en el momento*, por lo que el cálculo necesario debe realizarse n el menor tiempo posible.
- **La alineación de los elementos virtuales debe realizarse en 3D.** Los objetos sintéticos deben de estar en el espacio tridimensional.

A la hora de construir un sistema de realidad aumentada, podemos observar que depende de cuatro componentes físicos:

Figura 2.6: Representación conceptual de la realidad aumentada (James Provost)

- **Elementos de entrada y sensores de orientación.** Proporciona al sistema la información visual, la entrada del usuario y, potencialmente, ayudar a la orientación.
- **Fuente de datos.** Proporciona información aplicable al medio ambiente para que el sistema aumente la visión del usuario.
- **Periféricos de retroalimentación de los usuarios.** Principalmente en la forma de producción visual, pero también pueden incluir audio y otras interfaces de usuario.
- **Unidad de proceso.** Combina los datos de los sensores de entrada para determinar la orientación y aumentar la visión del usuario con información de la fuente de datos. Envía el resultado a los periféricos de retroalimentación del usuario.

2.3. Detección

En el contexto de la realidad aumentada, la detección es el proceso de localizar un objeto en una imagen capturada y calcular la posición y orientación de la cámara *(camera pose)* respecto a ese objeto.

Los enfoques que se han dado en las distintas técnicas de detección se pueden clasificar en dos tipos: La detección mediante marcas de referencia o mediante puntos de interés naturales.

2.3.1. Detección basada en marcas de referencia *(fiducial markers)*

Una marca de referencia es un objeto que se coloca en el campo de visión de la imagen a procesar y proporciona un punto de referencia y unas dimensiones conocidas de antemano, facilitando el proceso de detección y el calculo de la *pose*.

Mediante este método Kato presentó en 1999 un paper que utilizaba marcas cuadradas con un borde negro para calcular la *pose* de la cámara en tiempo real[15]. El resultado fue la biblioteca ARToolKit que ha popularizado la realidad aumentada.

Otros papers en la misma línea, como el de Stricker [33], en el que describe un método para encontrar las coordenadas 3D de las 4 esquinas de un marcador cuadrado, mientras que Park [26] describe un algoritmo para el cálculo de la *pose* de la cámara a partir de características conocidas.

Hasta el año 2002, las técnicas mediante marcas se habían estudiado ampliamente. Zhang [35] realizo un estudio recopilando y comparando varios de los principales enfoques que existían. A partir de esta fecha no se han presentado nuevos sistemas basados en marcadores.

2.3.2. Detección basada en puntos de interés naturales

El aspecto más importante de la detección de puntos de interés es hacerlo lo más coherente posible. Los mismos puntos de interés que se detectan en un frame, debe detectarse de nuevo en el siguiente cuadro, dado que todavía siguen presentes en la imagen.

Si tenemos un rectángulo sin bordes en movimiento delante de un fondo del mismo color, sería imposible de rastrear porque el sistema no podría distinguir dos puntos distintos en la imagen. Si tenemos un punto negro sobre un fondo blanco, y esté empieza a moverse, es muy fácil de seguir, ya que el sistema sólo tendría que encontrar ese punto en los siguientes frames. Este punto puede ser visto como una discontinuidad o un punto en el que se produce un cambio busco en la intensidad de la imagen.

Por tanto, el requisito deseable para considerar a un punto de interés es que debe ser una discontinuidad. Si tuviéramos muchos más de estos puntos también sería imposible diferenciar entre ellos y tendríamos el mismo problema que con el rectángulo del mismo color del fondo. Sólo podremos distinguirlos si la región que rodee al punto de interés es diferente, al menos en cierto grado, de la región local que rodea a todos los demás. Por tanto, el segundo requisito de un punto de interés es que él y la región que lo rodea debe ser único.

Prácticamente en todos los trabajos sobre detección mediante puntos de interés se pueden establecer las siguientes fases:

- **Extracción:** El proceso de extracción consiste en la búsqueda de zonas en la imagen con diferente apariencia que las que están a su alrededor, denominadas características *(features)*. Normalmente las características suelen ser bordes, esquinas o zonas más brillantes u oscuras en función del algoritmo utilizado en particular. A esta fase también se la denomina detección.

  Existen muchos algoritmos de detección, que obtienen distintas tipos de características, por ejemplo, el detector de esquinas de Harris[12] o el algoritmo FAST [29] que

devuelve los píxeles con valores máximos y mínimos en función de sus vecinos mediante técnicas de *Machine Learning*.

- **Descripción:** Se calcula el vector que describe la característica de un punto significativo para la posterior comparación entre otros puntos de interés. Los enfoques basados en el uso de descripción locales han sido ampliamente investigados y se dividen en dos tipos: el histograma de gradientes y la prueba binaria.

  El histograma de gradientes se calcula a partir de la cuantificación de los gradientes dentro de una área local. En SIFT [18], una zona se divide en subregiones y se calcula el histograma de gradiente en cada una de ellas. Este enfoque es utilizado y mejorado en los trabajos de Ambai [1], Bay (SURF) [4], y Wagner [34]

  Una prueba binaria es una comparación de la intensidad de dos píxeles y produce un resultado binario que representa qué píxel es más brillante. Se realizan cientos de pruebas para calcular un vector de características, ya que solo una de ellas no es lo suficientemente discriminatoria. El tema de investigación principal de este enfoque es el de muestreo eficiente entre dos píxeles y es utilizado en los métodos BRIEF [5], BRISK [17], y ORB [30].

- **Búsqueda de Coincidencias:** Los vectores de características se almacenan en una base de datos. Cuando se busca una coincidencia, se accede a la base datos con los datos del vector de consulta y se devuelve el vector almacenado más similar.

  Si un vector de características es de grandes dimensiones como en SIFT [18], la búsqueda completa no se podría realizar para tareas en tiempo real. En lugar de ello, se utilizan técnicas de árboles de búsqueda basado en la aproximación al vecino más cercano [2] [22]. El costo de la búsqueda de este enfoque depende del número de características almacenados en la base de datos. Otro tipo de búsqueda consiste en mapear el vector de características a un índice de tipo entero [9] y almacenarlo en una tabla hash. Este enfoque es teóricamente rápido con $O(1)$, independientemente del tamaño de la base de datos, pero es sensible a errores. Si un vector de características se describe como cadena binaria, se realizará una búsqueda completa en toda la tabla [5]. Para intentar corregir esto, se han planteado la utilización de árboles aleatorios [16] y estructuras no jerárquicas [25].

Hay muchas clases de detectores de características, pero los más utilizados son los detectores de esquinas (*corners*), de bordes (*edges*) y de regiones (*blobs*).

2.3.2.1. Detector de esquinas de Harris

El ejemplo más obvio de un punto de interés es una esquina, o la intersección de dos bordes. A lo largo de los años se han desarrollado una serie de algoritmos de detección de esquinas. La mayoría de los algoritmos de detección de puntos de interés calculan una función C de respuesta, ya sea para todos los píxeles, o sólo algunos píxeles seleccionados.

Probablemente el más famoso detector de esquinas se conoce como ***detector de esquinas de Harris*** desarrollado por Harris [12], que fue creado para la interpretación 3D de secuencias de imágenes. Al igual que con el detector de bordes de Canny [6], el cálculo de la segunda derivada parcial de una imagen en una dirección específica indicará regiones con cambios bruscos de intensidad de la imagen (discontinuidades) en esa dirección.

Dada una imagen en escala de grises en dos dimensiones, I, de modo que la intensidad de la imagen en un píxel específico se de por $I(x, y)$, Harris determinará C_{Harris} para un punto específico.

2.3.2.2. Shi-Tomasi

Shi y Tomasi [32] mejoraron el detector de Harris y encontraron que se obtenía un buen resultado en la detección de esquinas, siempre y cuando el autovalor más pequeño era mayor que un cierto umbral. $C_{ShiTomasi} = min(\lambda_1, \lambda_2)$. La función $C_{ShiTomasi}$ tiene el inconveniente de necesitar el cálculo explícito de los autovalores, que es costoso computacionalmente.

Los resultados muestran que Shi-Tomasi tiene mejor criterio para determinar a un punto de interés, devolviendo más puntos que Harris para un mismo umbral. Los resultados también muestran que Shi-Tomasi es más lento que el algoritmo de Harris para el mismo umbral, posiblemente debido al cálculo de los autovalores.

2.3.2.3. FAST (Features from Accelerated Segment Test)

FAST es algoritmo de detección de esquinas presentado por Rosten y Drummond [29]. En contraste con los métodos de detección de puntos de interés, como Harris, FAST tiene un enfoque mucho más cualitativo, y mucho menos costoso, para determinar si un píxel representa una esquina.

Para cada píxel, p, en una imagen en escala de grises, FAST examina 16 píxeles en un círculo de radio 3 alrededor de p, como se muestra en la figura 2.7. La intensidad (valor de gris) de un píxel está dado por $I(x, y)$. FAST simplemente afirma que p puede ser clasificado como un punto de interés si las intensidades de al menos 12 puntos contiguos en el círculo de prueba son más claros o más oscuros que $I(x_p, y_p)$ (la intensidad de p) para un cierto umbral, t.

Un punto de interés puede ser categorizado como «positivo» (brillante) si por lo menos 12 puntos circulares contiguos tienen intensidades mayores que $I(x_p, y_p)+t$, o «negativo» (oscuro) si sus intensidades son más pequeños que $I(x_p, y_p)-t$. Esta partición puede ser útil para que los puntos positivos no sean comparado con puntos negativos en etapas posteriores del *tracking*.

2.3.2.4. Detección de bordes

Muchos de los algoritmos de *tracking* basado en modelos buscan coincidencia entre los bordes de un objeto del mundo real con los bordes de un modelo conocido para determinar la *pose*.

La base de estos métodos es el cálculo de una característica geométrica a partir de los bordes. En el caso del trabajo de Hagbi [11], propone un método para reconocer formas planas y estimar la *pose* de la cámara a partir del contorno de las formas. Para cada concavidad del contorno, se extraen de las líneas bitangentes, puntos clave invariantes de la proyección. La *pose* inicial se estima a partir de estos puntos y son refinados mediante la minimización del error de reproyección.

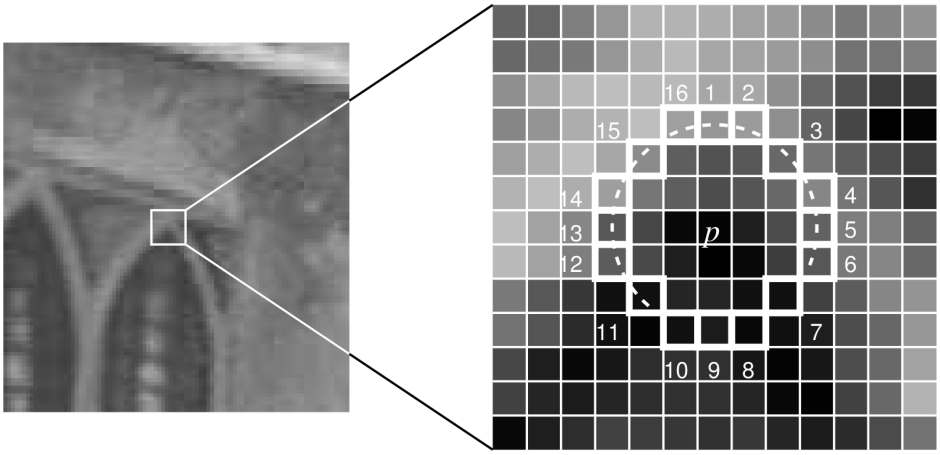


Figura 2.7: FAST: Círculo de análisis alrededor del píxel p. (Rosten y Drummond)

Existen gran variedad de algoritmos de detección de bordes, pero uno de los más populares y de mayor éxito es un método perfeccionado por Canny [6]. El detector devuelve un conjunto de curvas conectadas. El *tracking* de contornos relacionados con la estructura de un objeto de la imagen en lugar realizar un seguimiento de los píxeles individuales reduce significativamente la cantidad de datos a procesar, que es exactamente el propósito de la detección de características.

2.3.2.5. Detección de Regiones

Podemos definir una región *(blob)* como una zona de la imagen que tiene propiedades distintas, como el brillo o el color, comparada con otras zonas que la rodean.

La investigación en este área se centra en algoritmos que podemos agrupar en dos clases principales de detectores de regiones: métodos diferenciales y métodos basados en extremos locales.

Donoser [10] presenta un enfoque para extraer contornos que utiliza **MSER** (Maximally Stable Extremal Regions) para definir regiones de interes *(blob)*.

El **detector de regiones extremas MSER** (Maximally Stable Extremal Regions) selecciona aquellas regiones cuyos píxeles son más brillantes o más oscuros que todos los píxeles de su alrededor [20]. De este modo, las regiones quedan definidas por una propiedad extrema de la función de intensidad en la región y su límite exterior.

El algoritmo ordena todos los píxeles de la imagen según su intensidad con un coste computacional de $O(n)$ si el rango de los valores de la imagen S es pequeño, por ejemplo los valores de la imagen en escala de grises $\{0, \ldots, 255\}$, ya que la ordenación se implementa mediante un algoritmo BINSORT. Después de la ordenación, los píxeles se colocan en la imagen (con orden ascendente o descendente) y se van creando regiones de componentes conectados que van creciendo y fusionándose (utilizando un algoritmo de tipo *union-find*, con una complejidad temporal de $O(n \log(\log n))$) hasta que todos los píxeles han sido seleccionados.

Como resultado final se obtienen diferentes funciones de densidad que describen estructuras de datos que representan a cada una de las áreas de los componentes conectados que contiene la imagen.

Las regiones extremas se calculan buscando aquellos componentes conectados que permanecen constantes durante un número determinado de iteraciones. Para ello, se seleccionan como umbrales de las regiones extremas para las diferentes iteraciones los niveles de intensidad que son mínimos locales del rango de cambio de la función del área. Finalmente, el algoritmo transforma en elipses las regiones extremas, que inicialmente pueden tener cualquier forma.

2.3.3. Descriptores de Características

La búsqueda de correspondencias de un punto es una tarea importante para llevar a acabo con éxito la triangulación y el cálculo de la *pose* de la cámara. Con el fin de encontrar características correspondientes más fotogramas de vídeo que debe ser posible detectar características como se describe en la sección anterior , pero también es importante para identificar las características y relacionarlas entre fotogramas. Llamamos a esta descripción de la función y se trata de extraer información de la característica.

Al igual que con la detección de características , una amplia variedad de algoritmos de descripción de funciones se han presentado en los últimos años . Una buena descripción del carácter debe exhibir estas tres características :

- **Reproducible** Los descriptores de características deben ser fiables, encontrando los mismos puntos de interés bajo diferentes condiciones de visualización. Debe tener una alta precisión y una tasa baja de falsos positivos. También debe ser invariante a cambios en la rotación, traslación y escala.
- **Robusto** El descriptor debe ser capaz de identificar el mismo punto entre los distintos frames, incluso si hay cambios en la iluminación, se presente ruido en la imagen o existan pequeños cambios en el punto de vista .
- **Rápido** Debe ser capaz de extraer información de los puntos característicos y compararla con una gran base de datos en el menor tiempo posible, preferiblemente en tiempo real.

Lowe [18] presenta un método para la extracción de características de la imagen llamado SIFT *(Scale Invariant Feature Transform)*. Este proceso es invariante a cambios en la escala de imagen, traslación y rotación, así también, al menos parcialmente invariante, a cambios en la iluminación y transformaciones proyectivas 3D. Este enfoque transforma una imagen en una gran colección de vectores de características locales llamados «*SIFT Keys*» que se utilizan para su identificación.

2.3.3.1. SURF

El descriptor SURF, *Speeded-Up Robust Features*, fue desarrollado por Herbert Bay[4] como un detector de puntos de interés y descriptores robusto. El descriptor SURF guarda cierta similitud con la filosofía del descriptor SIFT [18], si bien presenta notables diferencias que quedarán patentes con la siguiente exposición sobre su desarrollo. Los autores afirman

Figura 2.8: Estimación de la orientación sobre puntos de interés. (Bay)

sin embargo que este detector y descriptor presentan principalmente 2 mejoras resumidas en los siguientes conceptos:

- Velocidad de cálculo considerablemente superior sin ocasionar perdida del rendimiento.
- Mayor robustez ante posibles transformaciones de la imagen.

Estas mejoras se consiguen mediante la reducción de la dimensionalidad y complejidad en el cálculo de los vectores de características de los puntos de interés obtenidos, mientras continúan siendo suficientemente característicos e igualmente repetitivos.

Las diferencias más originales respecto del descriptor SIFT:

- La normalización o longitud de los vectores de características de los puntos de interés es considerablemente menor, concretamente se trata de vectores con una dimensionalidad de 64, lo que supone una reducción de la mitad de la longitud del descriptor SIFT.
- El descriptor SURF utiliza siempre la misma imagen, la original.
- Utiliza el determinante de la matriz Hessiana para calcular tanto la posición como la escala de los puntos de interés

2.4. *Tracking*

En contraste con la detección, que estima la *pose* de la cámara en una imagen, el *tracking* es el seguimiento del objeto (y la estimación de la *pose* de la cámara respecto a ese objeto) en una secuencia de fotogramas.

El procedimiento a seguir es la de identificar «puntos clave» visualmente significativos en un frame que podamos encontrar de forma fiable de nuevo en el siguiente. El procedimiento de *tracking* se basa en encontrar una cantidad suficiente de estas correspondencias de puntos entre los frames.

El *tracking*, es un problema complejo debido a la pérdida de información causada por la proyección del mundo 3D en una imagen 2D, la calidad de la imágenes obtenidas, fondos difíciles de segmentar, oclusiones totales o parciales, cambios en la iluminación y la exigencia de para trabajar en tiempo real.

Para construir un buen sistema de *tracking*, es deseable que cumpla con los siguientes requisitos:

- **Robusto.** Incluso en condiciones complicadas, como fondos difíciles de segmentar, cambios de iluminación, oclusiones o movimientos complejos, un algoritmo de *tracking* debe ser capaz de seguir al objeto de interés.
- **Adaptable.** Adicionalmente a los cambios de entornos que se puedan producir, el objeto en sí también puede sufrir cambios. Esto requiere que el algoritmo tenga algún mecanismo de adaptación para el seguimiento del objeto según la apariencia que tenga en cada momento.
- **Cómputo en tiempo real.** Para obtener una sensación fluida y que el ojo humano no perciba retrasos en la imagen, al menos debemos trabajar y procesar 15 imágenes por segundo. Por tanto, es necesario que el algoritmo sea rápido y esté optimizado.

Desde el punto de la información obtenida para el cálculo de la *pose* de la cámara, las técnicas de *tracking* se pueden se dividir en dos tipos:

***Tracking* por detección o búsqueda (by matching).** El cálculo de la posición y orientación de la cámara *(camera pose)* se realiza en cada frame por correspondencia entre la imagen de entrada y otra de referencia mediante una técnica de detección. La información de anteriores «*poses*» de la cámara no son tenidas en cuenta para la estimación de la nueva *pose*. El calculo de la *pose* mediante arboles aleatorios [16] y estructuras no jerárquicas [25] son ejemplos de este tipo de enfoque.

***Tracking* mediante *tracking*.** Para el calculo, la *pose* previa es utilizada como *pose* inicial para el cálculo de la posición y orientación actual. Una vez que se detectado un objeto, se hace un seguimiento de los puntos claves del objetos en el siguiente fotograma [34], o minimizan la diferencia entre dos imágenes consecutivas [28] y analizan el cambio no lineal de iluminación producida [8]. La mayoría de estos enfoques se basan en la minimización del desplazamiento de la cámara entre dos imágenes sucesivas.

Otra clasificación de los métodos de *tracking* disponibles puede dividirse en dos clases: basado en detección de características y basados en detección de modelos.

2.4.1. *Tracking* por detección de características *(feature-based)*

En el mundo de la visión artificial una característica (feature) es una zona de la imagen que un algoritmo de *tracking* puede detectar y seguir a lo largo de múltiples frames. Normalmente las características suelen ser bordes, esquinas, zonas más brillantes u oscuras en función del algoritmo de *tracking* en particular.

En lugar de utilizar marcadores de referencia, la estimación de la *pose* de la cámara se puede realizar mediante la extracción de características naturales, como puntos, líneas, bordes o texturas. Esta línea de investigación también ha sido ampliamente estudiado. Park [27] presenta un método en el que utiliza las características naturales como una extensión en el *tracking* de características artificiales. Después de realizar el cálculo de la estimación de la *pose* de la cámara mediante características visuales conocidos, el sistema sistema adquiere dinámicamente características naturales adicionales y los utiliza para la actualización continua de la estimación de la nueva *pose*. De esta manera proporciona un seguimiento robusto, incluso cuando las marcas originales originales ya no están a la vista.

2.4.1.1. *Optical Flow*

El *Optical Flow* o flujo óptico juega un papel importante en la estimación y descripción del movimiento, por lo cual es comúnmente utilizado en tareas de detección, segmentación y seguimiento de objetos móviles en una escena a partir de un conjunto de imágenes.

El flujo óptico puede ser definido como el movimiento aparente de los patrones de intensidad en una imagen. La palabra aparente indica que el movimiento espacial de los objetos (campo de movimiento) puede coincidir o no con el flujo estimado. No obstante, en situaciones en las cuales el movimiento de los objetos implica un movimiento de sus patrones de intensidad en el plano imagen, el flujo óptico puede ser directamente relacionado con el movimiento de los objetos en la escena. La mayoría de las técnicas existentes para la estimación del flujo óptico se puede clasificar en 4 categorías: las basadas en gradientes espacio-temporales, las basadas en comparación de regiones, las basadas en fase y las basadas en energía.

En todas las estrategias de estimación de flujo óptico se parte de la hipótesis de que los niveles de gris permanecen constantes ante movimientos espaciales en un tiempo dado. Dicha hipótesis da lugar a la ecuación general de flujo óptico, donde $I(x, y, t)$ corresponde a la intensidad en niveles de gris del píxel (x, y) de la imagen I en el tiempo t.

$$I(x, y, t) = I(x + dx, y + dy, t + dt) \tag{2.19}$$

Expandiendo la ecuación anterior en series de Taylor sobre el punto (x, y, t)

$$I(x, y, t) = I(x, y, t) + dx\frac{\partial I}{\partial x} + dy\frac{\partial I}{\partial y} + dt\frac{\partial I}{\partial t} + \epsilon \tag{2.20}$$

donde ϵ contiene la información de las derivadas de orden superior. Si se asume ϵ despreciable, la ecuación de flujo óptico puede reescribirse como

$$I_x u + I_y v + I_t = 0 \tag{2.21}$$

donde (u, v), con $u = dx/dt$ y $v = dy/dt$, corresponde al vector de flujo óptico y, I_x y I_y son las derivadas parciales horizontal y vertical de la imagen, respectivamente. Para cada píxel (x, y) de la imagen, en el tiempo t, puede plantearse la ecuación anterior, sin embargo no existe una única solución para esta ecuación.

Pueden emplearse diferentes restricciones para estimar el flujo óptico en la imagen. Horn y Shunck en [14] restringen el flujo óptico en la imagen a variar suavemente, por lo que la ecuación es minimizada junto a un término de regularización que penaliza los cambios

Figura 2.9: Lucas-Kanade: Seguimiento de puntos. (David Stavens)

abruptos del flujo. Lucas y Kanade (LK) [19] proponen un método alternativo que se describe a continuación el cual puede ser implementado de forma más eficiente que el propuesto en [14]. Las técnicas propuestas en [14] y [19] se basan en gradientes espacio-temporales, pues minimizan la ecuación anterior.

2.4.2. *Tracking* por detección modelos *(model-based tracking)*

La tendencia más reciente en las técnicas de *tracking* es el basados en modelos. Estas técnicas utilizan explícitamente un modelo de las características de los objetos rastreados, como por ejemplo, un modelo CAD ó un patrón del objeto basado en sus características distinguibles. El primer trabajo basado en modelos fue obra de Comport [7] que en 2003 que utilizó esta aproximación utilizando las características geométricas de líneas, círculos, cilindros y esferas del modelo para el cálculo de la *pose* de la cámara.

3

Descripción de la propuesta

En este capítulo se describe la arquitectura propuesta para el desarrollo de un framework para la construcción de interfaces naturales de usuario basado en el uso de Realidad Aumentada y Visión Artificial del proyecto ARgos.

El framework se ha diseñado como una arquitectura con 6 subsistemas (Figura 3.1) y una herramienta externa para calibrar cámaras y proyectores (Figura 3.3). A continuación se indica el cometido de cada uno de los módulos:

- *Sistema de Calibrado (calibrationToolbox):* Implementado como una aplicación externa, su función es obtener los parámetros intrínsecos y extrínsecos de la cámara y el proyector.
- *Sistema de Captura:* Es el encargado de la capturar y proveer de imágenes al sistema por medio de la cámara de la Raspberry Pi ó a través de cámaras USB.
- *Sistema de Tracking y Registro:* Su misión es detectar y calcular la *pose* de los documentos mostrados al sistema.
- *Sistema de Identificación de Documentos:* Este módulo asume la tarea de identificar los documentos en base a su contenido.
- *Sistema de Interacción Natural de Usuario:* Proporciona los mecanismos para implementar el paradigma de «pantalla táctil» como interfaz del sistema.
- *Sistema de Modelos Matemáticos:* Consiste en la implementación de los modelos matemáticos, de los que se sirven el resto de módulos para realizar sus cálculos.
- *Sistema de Soporte y Utilidades:* Es un conjunto de utilidades internas. Incorpora el *log* del sistema, funciones para dibujar y un gestor de configuración.

En los siguientes apartados se explicará en detalle los sistemas sin tener en cuenta la distinción entre la parte del cliente y la del servidor, ya que es totalmente transparente al sistema y no corresponde al alcance del proyecto.

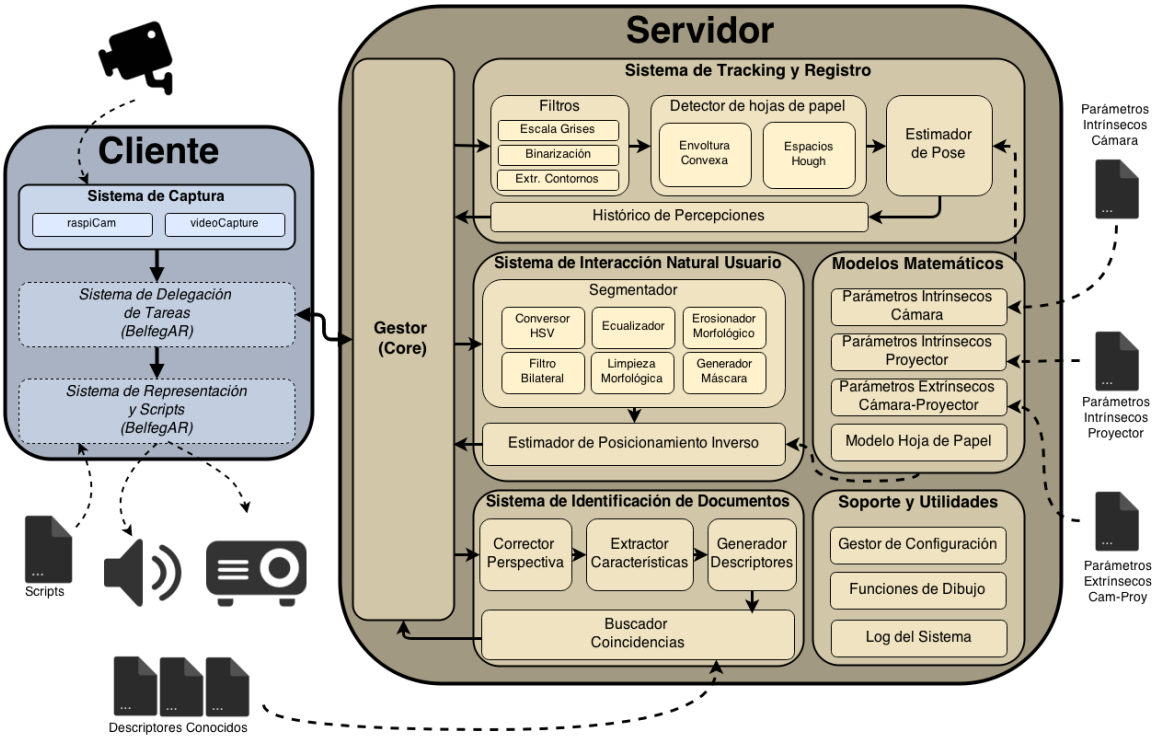


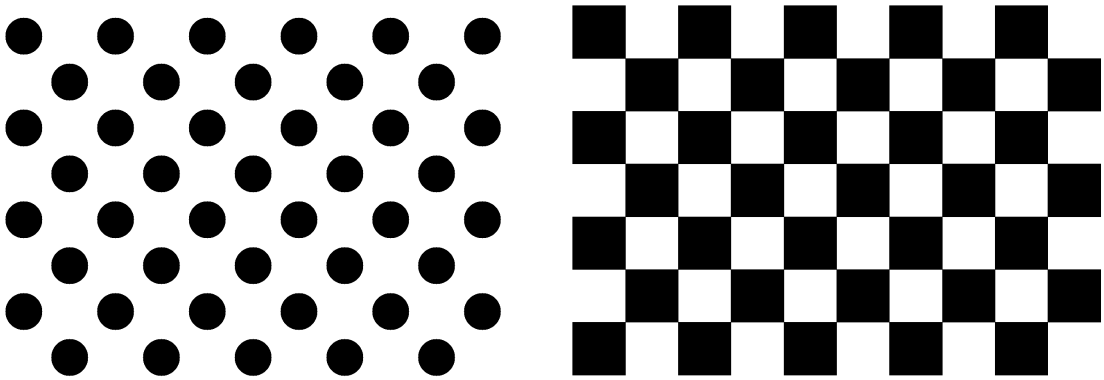
Figura 3.1: Diagrama de la estructura del framework en el contexto de *ARgos*

3.1. Módulo externo de calibración (calibrationToolbox)

Como ya se explicó en el capitulo 2, los objetivos de realizar el proceso de calibración son la estimación de los parámetros intrínsecos y extrínsecos de la cámara. Los parámetros intrínsecos se refieren a las características internas de la cámara, como por ejemplo, su distancia focal, distorsión, y el centro de la imagen. Los parámetros extrínsecos describen su posición y orientación dentro de un espacio de referencia. Conocer los parámetros intrínsecos es un primer paso esencial, ya que permite calcular la estructura de la escena en el espacio euclídeo y elimina la distorsión de lentes, la cual afecta a la precisión.

Para ubicar objetos en el mundo real, establecemos un sistema de referencia, denominado sistema de referencia global. Un objeto en una imagen es medido en términos de coordenadas de píxeles, los cuales están en el sistema de referencia de la imagen. El sólo conocer la distancia en píxeles entre puntos en una imagen, no nos permite determinar la distancia correspondiente a los mismos puntos en el mundo real. Por lo tanto, es necesario establecer las ecuaciones que unan el sistema de referencia global con el sistema de referencia de la imagen, de manera de establecer la relación entre los puntos en coordenadas en el espacio 3D y los puntos en coordenadas de imagen 2D.

Desafortunadamente, no se puede establecer esta relación directamente, haciéndose necesario establecer un sistema de referencia intermedio, llamado sistema de referencia de la cámara. Por lo tanto, se deben encontrar las ecuaciones que unan el sistema de referencia de la cámara con el sistema de referencia de la imagen, y las ecuaciones que unan el siste-



(a) Patrón de Círculos Asimétricos (b) Patrón Tipo Tablero de Ajedrez

Figura 3.2: Tipos de patrones de calibración

ma de referencia del global con el sistema de referencia de la cámara. Al resolver el sistema de ecuaciones generado se obtiene la relación buscada.

Básicamente, el proceso consiste en obtener una serie de imágenes en los que se encuentre visible un patrón plano (de dimensiones conocidas), con distintas orientaciones y distancias de la cámara. De cada patrón encontrado en las imágenes obtenemos una ecuación de homografía que establece la relación entre los puntos en coordenadas en el espacio 3D y los puntos en coordenadas de imagen 2D. Aunque en teoría con dos imágenes sería suficiente para resolverlo mediante un sistema lineal de ecuaciones, el objetivo es obtener el mayor número de ellas, ya que en la práctica existe gran cantidad de ruido en las imágenes adquiridas. Se recomienda por tanto, para obtener buenos resultados, al menos 10 imágenes correctas del patrón en diferentes posiciones.

En principio, cualquier objeto caracterizado apropiadamente podría ser utilizado como patrón para la calibración. Existen otros métodos que basan sus referencias en objetos tridimensionales o que requieren de patrones de calibración consistentes, en al menos dos planos ortogonales.

La principal ventaja de la utilización de patrones planos frente a otras técnicas es su flexibilidad. No necesita de una preparación exhaustiva de la escena, ni es necesario conocer las posiciones de los mismos. También resulta mucho más complicada la construcción y distribución de objetos 3D precisos para realizar una calibración.

El proceso de calibrado de la cámara esta basado esencialmente por el enfoque de Zhang [35]. Se utiliza un patrón tipo tablero de ajedrez, en la que se alternan cuadrados blancos y negros, de dimensiones conocidas. El patrón se imprime y se pega sobre una superficie plana rígida. A continuación se obtiene una serie de imágenes en los que se encuentre visible el patrón desde varias posiciones.

Se realiza el cálculo de las homografías entre el patrón y sus imágenes. Estas transformaciones proyectivas 2D producen un sistema de ecuaciones lineales que al resolverse obtiene los parámetros de la cámara. Esta fase generalmente es seguida por una etapa de refinamiento no lineal, basado en la minimización del error total de reproyección.

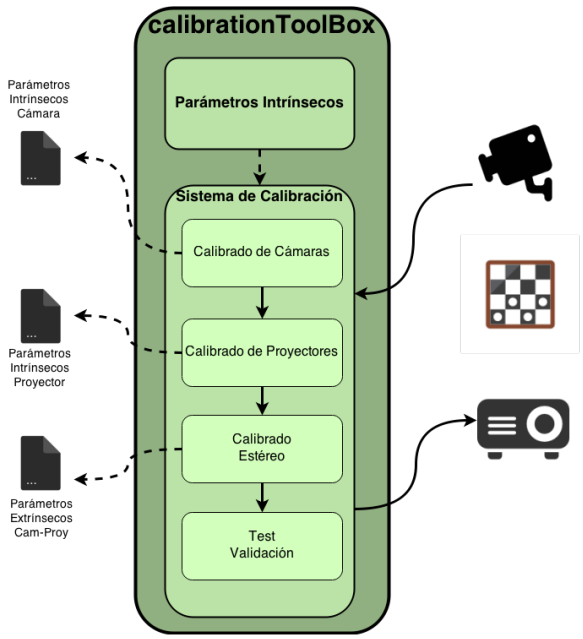


Figura 3.3: Diagrama de la estructura de calibrationToolbox

Se ha diseñado y construido como una utilidad aparte del proceso principal, ya que una vez calibrado el sistema, genera unos ficheros YAML con los parámetros intrínsecos y extrínsecos que se cargan en el proyecto. Mientras que la cámara y el proyector mantengan su posición y rotación entre ellos, no es necesario realizar una nueva calibración y es posible mover todo el sistema.

El módulo implementado está basado en un *plugin* para *openFrameworks* que han realizado Álvaro Cassinelli, Niklas Bergström y Cyril Diagne a partir de un complemento desarrollado por Kyle McDonald. Es capaz de calibrar cámaras y proyectores, consiguiendo los parámetros intrínsecos de ambos, además de los extrínsecos en cuestión de varios minutos.

3.2. Subsistema de captura

El módulo de captura de vídeo se encarga de crear y proporcionar fuentes de vídeo de diversa naturaleza para dar soporte al submódulo de *tracking* y registro, al submódulo de interacción natural de usuario, y al submódulo de identificación de documentos.

Es capaz de dar soporte multicámara. Se pueden crear tantas fuentes de vídeo como se disponga en el sistema, que será de alguno de los siguientes tipos:

- **RaspiCam:** Para un rendimiento óptimo en la Raspberry Pi, al menos la cámara principal debería ser de este tipo.
- **Cámara USB:** En caso de no disponer de una raspiCam o si se quieren incluir cámara adicionales para obtener capturas desde otra posición (videoconferencia,...).

El uso la cámara raspiCam *(Raspberry Pi Camera Board)*, presenta un inconveniente. El dispositivo no es compatible con *video4linux* y se debe utilizar la API MMAL (Multi-Media Abstraction Layer) sobre OpenMAX, para acceder a los datos de la cámara y transferirlos a la pantalla o codificarlo como imágenes o vídeos. Esto se traduce en que no es posible la utilización de OpenCV para la gestión del dispositivo y la estructura de datos para almacenar la imagen no es compatible con `cv::Mat`.

La solución definida en el submódulo de captura es la implementación en dos componentes: el `VideoCapture`, que proporciona la interfaz de creación y gestión de fuentes de vídeo basada en la biblioteca de visión artificial OpenCV y `RaspiCam` para la gestión de la cámara propia de la Raspberry.

Gracias al diseño, en el que las fuentes de vídeo concretas tienen que respetar un interfaz común, se obtiene un acceso homogéneo. Así, el acceso a los recursos que proporciona cualquier fuente es común a todas ellas.

3.3. Subsistema de *tracking* y registro

El cálculo del registro requiere posicionar el sistema cámara-proyector, mediante su posición y rotación, relativo a las hojas de papel que se encuentren en la escena capturada. Los métodos de *tracking*, en general, son los encargados de obtener una estimación de la trayectoria que realiza un objeto.

Para la construcción del framework, se ha optado por implementar un sistema de *tracking* visual basado en una aproximación ***bottom-up***[31], en la que se calculan los seis grados de libertad de la cámara a partir de lo que se está percibiendo en la imagen.

El módulo tiene como entrada el *frame* actual, donde va a realizar la búsqueda de cuadriláteros candidatos a ser hojas de papel; los parámetros de calibración de la cámara y el proyector; las dimensiones de la hoja de papel y por último, si la pose se debe calcular para representarse por pantalla o en el proyector.

La salida consistirá en un vector con las hojas de papel detectadas (definidas por las esquinas), así como sus parámetros extrínsecos correspondientes. Estos parámetros extrínsecos, consistentes en una matriz de rotación y un vector de translación, son los que aplicados a un objeto virtual 3D producen la transformación necesaria, para al realizar su representación esté correctamente situado respecto al sistema de referencia del papel.

Conociendo las posiciones 2D de las aristas y vértices que definen la hoja de papel, y el modelo de proyección de la cámara es posible estimar la posición y rotación 3D de la cámara relativamente al documento. Aprovechando que conocemos la estructura de formato normalizado (según ISO 120/DIN 476) de una hoja de papel, con un tamaño previamente conocido nos permite definir un sistema de coordenadas local de cada hoja detectada, de modo que obtengamos la matriz de transformación 4x4 del sistema de coordenadas de la hoja al sistema de coordenadas de la cámara.

El enfoque utilizado es similar al de ArUCo ó ARToolkit, mediante un algoritmo de detección de bordes y un método de estimación de la orientación. Sobre la imagen obtenida se inicia el primer paso de búsqueda de hojas de papel. La imagen se convierte a blanco y negro para facilitar la detección de cuadriláteros; primero se convierte a escala de grises, y después se binariza eligiendo un parámetro de umbral *«threshold»* que elige a partir de qué valor de gris (de entre 256 valores distintos) se considera blanco o negro. A continua-

(a) Imagen de entrada en el módulo paperDetector (b) Rectángulo detectado

Figura 3.4: Detección de la hoja de papel

ción el algoritmo de visión por computador extrae componentes conectados de la imagen previamente binarizada,cuya área es suficientemente grande como para ser una hoja de papel. A estas regiones se les aplica un algoritmo de detección de contornos, obteniendo a continuación los vértices y aristas que definen la región de la hoja en 2D.

3.3.1. Extracción de contornos

El primer paso en el proceso de detección, consiste en convertir el *frame* capturado por la cámara a escala de grises.

La umbralización consiste en definir un valor umbral y compararlo con cada uno de los píxeles de la imagen. A los píxeles que estén por debajo del umbral se les asigna un valor, y a los que estén por encima otro. De esta forma se divide toda la población de valores en tan sólo dos grupos, reduciendo considerablemente la complejidad de la información a analizar.

Se han desarrollado tres soluciones distintas para tratar de resolver este problema del *thresholding* básico (fijo, adaptativo y el método Canny). En un principio, sólo se implementaron los métodos fijos y adaptativo, pero en una fase de optimización realizada posteriormente, se incluyo el método Canny.

La ventaja aportada por el algoritmo de Canny respecto a los anteriores es que proporciona bordes más precisos, y es tolerante a las variaciones de iluminación, produciendo menores índices de ruido en la imagen binarizada.

El siguiente paso del proceso es la detección de contornos, una operación un tanto distinta a las anteriores por dos motivos principales. El primero es que no consiste en iterar aplicando una misma función de forma monótona sobre todos los píxeles de la imagen. Y el segundo, es que el resultado que se obtiene no es una nueva imagen, sino un colección de conjuntos de puntos sobre la imagen.

Para este paso se parte de la imagen resultante del paso anterior. Es decir, de una imagen que tan sólo consta de píxeles con valor cero o uno. Sobre ella se aplica un algoritmo que la barre, empezando por su esquina superior izquierda, a la busca de un primer píxel a

uno, y que cuando lo encuentra es capaz de seguir la cadena de píxeles con valor uno que se encuentran unidos a él hasta volver al píxel de partida. Esa cadena de píxeles a uno encontrados se denomina contorno. Y es más, el algoritmo es capaz de encontrar todos los contornos presentes en la imagen, ya que cuando termina con uno empieza de nuevo el proceso hasta asegurarse de haber barrido la imagen por completo.

En la documentación, así como la referencia de implementación, se refiere al *paper* original «Topological structural analysis of digitized binary images by border following» de Satoshi Suzuki and Keiichi Abe.

Básicamente hay un contador de contornos encontrados y un *buffer* de píxeles recorridos. El contador se inicializa a cero y el *buffer* con una copia de la imagen original. Se barre el *buffer* de arriba abajo y de izquierda a derecha. Una transición de un píxel 0 a otro 1 indica que se ha detectado un borde exterior, momento en que se suma uno al contador de contornos encontrados y se buscan todos los píxeles 1 vecinos del encontrado. Si el píxel no tiene vecinos a 1 se cambia el valor del píxel por el del contador con signo contrario y se empieza otra vez con el siguiente píxel. En caso contrario se cambia el valor del píxel por el del contador, excepto que su valor sea mayor que 1, lo que significa que ya ha sido visitado, y se continúa buscando vecinos a 1 hasta retornar al píxel inicial. Una transición de un píxel con valor igual o mayor que 1 a otro 0 indica que se ha detectado un borde interior, momento en que se repite el mismo proceso usado para los contornos exteriores.

El algoritmo presenta una pequeña dificultad en los bordes de la imagen, y para solventarlo presupone que la imagen está rodeada por los cuatro lados de píxeles con valor 0. Es decir, que el *buffer* que utiliza tiene una fila más por encima y por debajo, y una columna más a derecha e izquierda, que la imagen original.

Gracias a este algoritmo, no sólo se obtienen los contornos exteriores, como ocurre con otras implementaciones, sino también los interiores a otros, y retornarlos clasificados jerárquicamente.

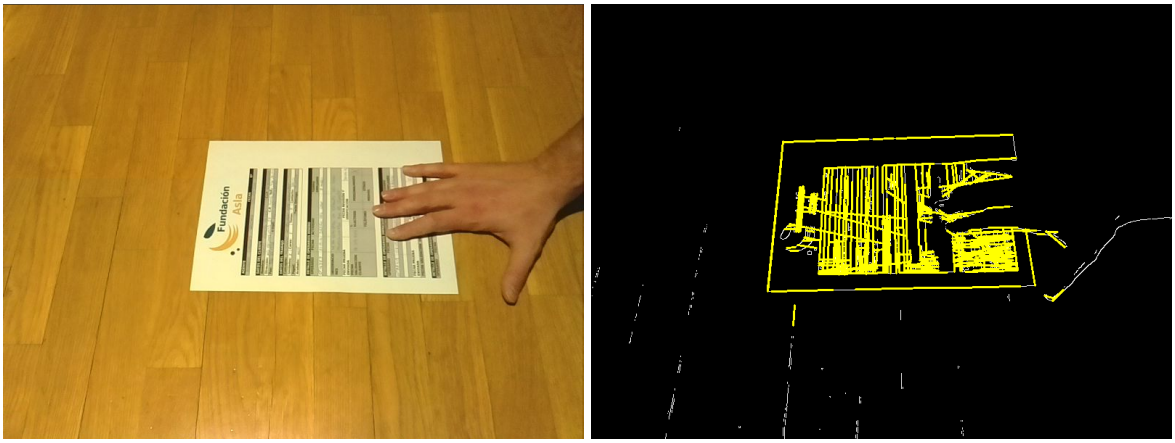
Otra ventaja de esta implementación, al devolver todos y cada uno de los puntos que forman el contorno, es que permite tomar algunas decisiones tempranas, como por ejemplo descartar los contornos que no tengan un mínimo de puntos. Es decir, aquellos que no son susceptibles de formar la hoja de papel.

3.3.2. Aproximación a polígonos

Los contornos de los que se parte para este paso son colecciones de puntos del paso anterior. El objetivo de este método es obtener el polígono que mejor se ajuste a cada uno de los contorno de entrada.

En primer lugar, sobre aquellos contornos que cumplan con la restricción de superficie, se calcula la **envoltura convexa del contorno**. El objetivo del cálculo de la envoltura es la de obtener un contorno *limpio*, ya que tras este paso se han eliminado puntos interiores debidos a reflejos, sombras o ruido en la imagen que se hubiesen detectado.

Matemáticamente se define la envoltura convexa de un conjunto de puntos X de dimensión n como la intersección de todos los conjuntos convexos que contienen a X.



(a) Imagen de entrada con solapamiento (b) Obtención de componentes no conectados

Figura 3.5: Error en la detección de la hoja de papel con solapamiento

Dados k puntos $x_1, x_2, \dots x_k$ su envoltura convexa C viene dada por la expresión:

$$C(X) = \left\{ \sum_{i=1}^{k} \alpha_i x_i \;\middle|\; x_i \in X,\, \alpha_i \in \mathbb{R},\, \alpha_i \geq 0\,, \sum_{i=1}^{k} \alpha_i = 1 \right\} \tag{3.1}$$

Es decir, teniendo un conjunto de puntos en el plano, su envoltura convexa está definida por el polígono convexo de área mínima que cubre todos los puntos (esto es, todos los puntos están dentro del polígono).

Para encontrar la envoltura de una conjunto de puntos 2D se ha utilizdo el algoritmo de Sklansky con una complejidad $O(NlogN)$ en su actual implementación.

A continuación, la conversión de una colección de puntos en un polígono se realiza mediante el algoritmo de Douglas-Peucker. El algoritmo admite como entrada una lista de puntos y una distancia máxima permitida. Define un segmento que va desde el primer punto de la lista hasta el último, y calcula la distancia más corta que hay desde dicho segmento a todos y cada uno del resto de puntos de la lista. Si encuentra un punto a una distancia mayor que la máxima pasada como parámetro divide la lista y el segmento en dos, utilizando el punto encontrado como nuevo extremo. Y vuelve a empezar el proceso comprobando cada nueva lista contra cada nuevo segmento de forma individual, creando nuevas listas y segmentos si fuera necesario, de forma recursiva.

Una vez convertidos los puntos a polígonos se descartan los que no tengan cuatro lados.

En este punto, en los casos en los que no existan grandes solapamientos sobre el papel, debemos obtener una serie de polígonos que se corresponden con cada una de las hojas detectadas en la imagen.

Gracias a la utilización del cálculo de la envoltura convexa como paso previo, nos permite que existan pequeños solapamientos en los bordes, y aún así, se siga detectando el contorno del papel. Sino, las oclusiones supondrían una interrupción en la conectividad de los contornos, y por lo tanto, el fallo en el proceso de detección.

3.3.3. Búsqueda de cuadriláteros en el espacio de Hough

Mientras se este realizando alguna interacción con el documento, nuestra mano, dedos e incluso parte del brazo estarán solapando gran parte de la hoja que queremos detectar. Tras aplicar el detector de bordes a la imagen recibida, obtenemos segmentos no conectados que las funciones anteriores no serán capaces de clasificar como cuadriláteros candidatos a ser una hoja de papel.

En estos casos, la búsqueda de la hoja de papel en la imagen se debe realiza mediante detectores de líneas basados en la transformada de Hough.

Esta función se invocará sólo cuando no se hayan detectado cuadriláteros mediante el algoritmo anterior, ya que esta función es costosa computacionalmente debido al tamaño del espacio de búsqueda.

El resultado será, al igual que la función anterior, un vector con todos los cuadriláteros que corresponden a las hojas de papel detectadas.

La transformada de Hough es una técnica para detectar bordes llevando los puntos al espacio paramétrico donde se transforman en rectas.

Basándose en lo anterior, la recta $y = m * x + n$ se puede representar como un punto (m, n) en el espacio de parámetros. Sin embargo, cuando se tienen rectas verticales, los parámetros de la recta (m, n) no están definidos. Por esta razón se utilizan los parámetros que describen una recta en coordenada polares, denotados (ρ, θ).

El parámetro ρ representa la distancia entre el origen de coordenadas y el punto(x, y), mientras que θ es el ángulo del vector director de la recta perpendicular a la recta original y que pasa por el origen de coordenadas.

Usando esta parametrización, la ecuación de una recta se puede escribir de la siguiente forma:

$$y = (-\frac{\cos\theta}{\sin\theta}) * x + (\frac{\rho}{\sin\theta}) \tag{3.2}$$

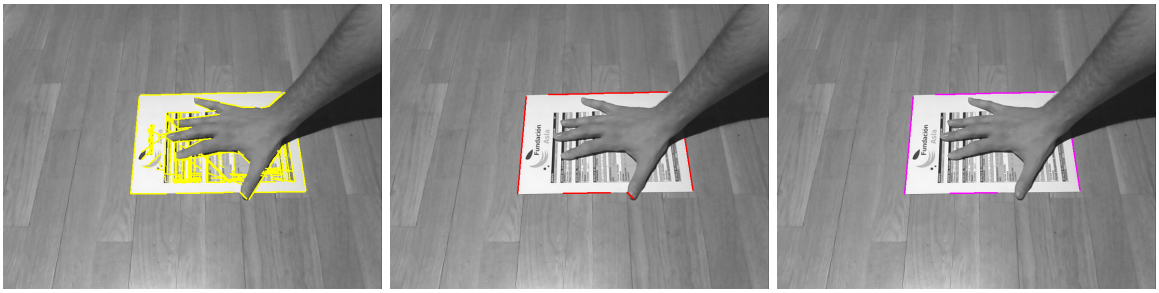
que se puede reescribir como

$$\rho = x * \cos\theta + y * \sin\theta \tag{3.3}$$

Entonces, es posible asociar a cada recta un par (ρ, θ) que es único si $\theta \in [0, \pi)$ y $\rho \in \mathbb{R}$ ó $\theta \in [0, 2\pi)$ y $\rho \geq 0$. El espacio (ρ, θ) se denomina espacio de Hough para el conjunto de rectas en dos dimensiones.

Para un punto arbitrario en la imagen con coordenadas (x_0, y_0), las rectas que pasan por ese punto son los pares (ρ, θ) con $\rho = x * \cos\theta + y * \sin\theta$ donde ρ (la distancia entre la línea y el origen) está determinado por θ. Esto corresponde a una curvas sinusoidal en el espacio (ρ, θ), que es única para ese punto. Si las curvas correspondientes a dos puntos se intersecan, el punto de intersección en el espacio de Hough corresponde a una línea en el espacio de la imagen que pasa por estos dos puntos. Generalizando, un conjunto de puntos que forman una recta, producirán sinusoides que se intersecan en los parámetros de esa línea. Por tanto, el problema de detectar puntos colineales se puede convertir en un problema de buscar curvas concurrentes.

Las zonas de este espacio donde más líneas se cortan se convierten en parámetros de posibles rectas en el espacio de la imagen.



(a) Segmentos dentro de la zona de proyección

(b) Segmentos más externos y mayores que un umbral

(c) Segmentos que forman 90º ó 180º

Figura 3.6: Proceso de detección mediante búsqueda en el espacio de Hough

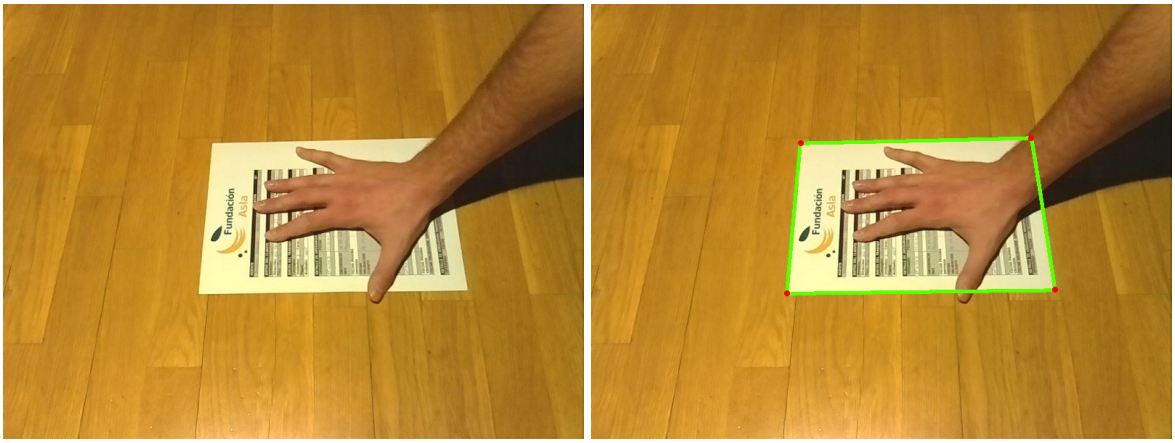
Se seleccionan un conjunto de las rectas detectadas para ser examinado en función de las siguientes características:

1. El segmento se encuentra dentro o en la proximidades de la zona de proyección del sistema. Con esta restricción descartaremos aquellos segmentos que sabemos de antemano que no van a ser hojas de papel, como por ejemplo, otros objetos que se encuentren en la imagen o los bordes de la superficie donde se encuentra en prototipo.
2. El segmento es mayor que un valor umbral.
3. Sólo se tienen en cuenta los segmentos más próximos a los límites de proyección. El detector de Hough puede encontrar rectas en el interior de la hoja de papel que estamos buscando. Así sólo seleccionaremos los posibles candidatos a formar el borde del papel.
4. Los ángulos que formen dos rectas contiguas estará comprendido dentro del rango de 75º a 105º.
5. Los ángulos que formen dos rectas opuestas deben ser cercanos a 180º ó 360º.

Tras la selección anterior obtendremos un conjunto de segmentos de rectas extraídas de la imagen. Cuatro segmentos se consideran que formarán un cuadrilátero candidato si cumplen los siguientes criterios:

- Las intersecciones de los cuatro segmentos están dentro de la imagen y próximos a la zona de proyección.
- Sólo intersecan en cuatro puntos.
- El área del cuadrilátero delimitado por las intersecciones esta entre 47000 y 49000 píxeles, ya que garantiza que el cuadrilátero que se detectó representa una hoja de papel.

Para ello se generan las posibles combinaciones de 4 segmentos del conjunto de rectas detectadas y se analizan los criterios anteriores.



(a) Imagen de entrada con solapamiento (b) Cuadrilátero detectado en el espacio de Hough

Figura 3.7: Detección de una hoja de papel con solapamiento

3.3.4. Estimación de la pose

El reto principal en un sistema de realidad aumentada es la estimación de la *pose*. Básicamente consiste en obtener, en función del sistema de referencia del dispositivo de representación, *donde* se debe colocar el objeto virtual, que *orientación* tiene y con que *tamaño*, por medio de matrices de translación, rotación y escalado respectivamente.

Para convertir las coordenadas del mundo a coordenadas de pantalla, de espacio tridimensional a bidimensional, se utilizará la fórmula 2.12 desarrollada en la sección 2:

$$q = K[R|t]Q \tag{3.4}$$

donde

$$q = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \quad K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \quad Q = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{3.5}$$

$$[R|t] = \begin{bmatrix} R_{1,1} & R_{1,2} & R_{1,3} & T_1 \\ R_{2,1} & R_{2,2} & R_{2,3} & T_2 \\ R_{3,1} & R_{3,2} & R_{3,3} & T_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.6}$$

Por tanto, q representa el espacio bidimensional en coordenadas homogéneas, y Q el espacio tridimensional. f_x es la distancia focal en el eje X, f_y la distancia focal en el eje Y, (c_x, c_y) marcan el punto principal, el punto donde el eje de visión corta el plano de visión (normalmente suele estar en el centro de la imagen, o muy cerca).

Se establece el centro del papel como centro de coordenadas $(0, 0, 0)$, por lo que las esquinas que se le proporcionan a la función son:

$$ObjPoints = \begin{bmatrix} -w/2 & -h/2 & 0 \\ w/2 & -h/2 & 0 \\ w/2 & h/2 & 0 \\ -w/2 & h/2 & 0 \end{bmatrix} \tag{3.7}$$

Siendo h y w la altura y anchura de la hoja de papel respectivamente.

3.3.5. Refinamiento de la *pose*

El posicionamiento calculado puede variar en cada frame, aun cuando el papel no se haya movido realmente. Esto puede ser debido a variaciones en la iluminación que alteran sensiblemente los procesos de detección o errores debido a la naturaleza imprecisa de los métodos utilizados.

Este refinamiento de las percepciones se basa en la capacidad del sistema de almacenar las últimas *n* percepciones. Este parámetro es configurable en tiempo de compilación.

Cuando se recibe una última percepción, se calcula una media ponderada con las cuatro últimas percepciones, de forma que la percepción refinada es la resultante de la siguiente fórmula:

$$P_{actual} = P_1 * 0{,}5 + P_2 * 0{,}25 + P_3 * 0{,}15 + P_4 * 0{,}1 \tag{3.8}$$

Siendo P_1 la percepción más reciente, y el resto las percepciones almacenadas en el histórico. De esta forma, se da más peso a la percepción más actual, pero se tiene en cuenta las anteriores. El resultado es un movimiento más fluido y suave, aunque da la sensación de haber un *delay* al momento de representar la posición. Este retraso es lógico, al tener en cuenta percepciones pasadas.

3.4. Subsistema de identificación de documentos

El objetivo de este módulo es la identificación rápida de documentos empleando algoritmos de recuperación de imágenes, comparando el documento que está siendo analizado con una base de datos de documentos conocidos por el sistema.

El prototipo construido ha sido configurado para funcionar con varios documentos proporcionados por la asociación ASPRONA. Estos documentos son partes de trabajo reales que se utilizan en un taller de serigrafía que tiene la asociación. Otra sugerencia realizada, y que han puesto de manifiesto que sería de gran utilidad de cara una implantación real, es el reconocimiento de facturas.

La solución implementada, debido a los requerimientos anteriores, corresponde a la búsqueda de coincidencias de características locales del documento. Es la más adecuada para la identificación de documentos estructurados o semi-estructurados, como pueden ser el caso de formularios, facturas, billetes de tren o tickets.

A la hora de elegir entre los detectores basados en la extracción de características invariante, se ha preferido SURF frente otros descriptores como SIFT ó BRISK por las siguientes razones:

- **Velocidad de cálculo** considerablemente superior al resto de los detectores, sin ocasionar perdida del rendimiento.
- **Más robusto** ante posibles transformaciones de la imagen.
- **No necesita ningún paso previo de segmentación** y la identificación es para obtener documentos similares (no idénticos) a la imagen utilizada como consulta.

(a) Imagen de Entrada (b) Cuadrilátero extraído

Figura 3.8: Eliminación de la transformación de perspectiva

A continuación se detallan los pasos que realiza el algoritmo.

3.4.1. Eliminación de la transformación de perspectiva

Este paso tiene como entrada la imagen en escala de grises obtenida por la cámara y los cuadriláteros candidatos encontrados en el módulo de detección de hojas de papel. Para cada hoja de papel, extrae la región de la imagen que cubre cada una de ellas eliminando la transformación de perspectiva, es decir, la deformación que se produce en el papel debido a la perspectiva.

El framework utiliza dos funciones para este paso:

- **`cv::getPerspectiveTransform`** calcula una matriz que multiplicada por un punto en el cuadrilátero origen devuelve un punto equivalente en el cuadrado destino.
- **`cv::warpPerspective`** acepta una imagen, un cuadrilátero, una matriz de transformación, y retorna una nueva imagen del área cubierta por el cuadrilátero sobre la que se ha aplicado la matriz de transformación para eliminar la deformación debida a la perspectiva.

El resultado de este proceso es una imagen de cada una de las hojas de papel encontradas. Una para cada cuadrilátero candidato. Se utiliza un tamaño de 600x420 píxeles para las imágenes destino ya que estas dimensiones mantienen la proporcionalidad de tamaño del folio.

Aunque SURF es invariante a la rotación, translación y escala, realizar este paso previo nos proporciona varias ventajas. Por una lado, se puede realizar la identificación de varios documentos dentro de la misma imagen, y al extraer de la imagen inicial el documento a identificar, estamos eliminado *ruido* que podría influir en el resultado de la detección, aumentado la robustez del algoritmo mediante el análisis exclusivo de la región a identificar dentro de la imagen general.

3.4.2. Extracción de Características

El proceso de extracción consiste en la búsqueda de zonas en la imagen con diferente apariencia que las que están a su alrededor, denominadas características *(features)*. Normalmente las características suelen ser bordes, esquinas o zonas más brillantes u oscuras en función del algoritmo utilizado en particular.

Mediante la clase wrapper **FeatureDetector** obtenemos una interfaz común que permite cambiar fácilmente entre diferentes algoritmos.

La función **detect** extrae los puntos clave de la imagen que cumplan el umbral para el valor del determinante de la matriz hessiana, definido en el constructor del extractor. Estos punto se almacenan en una estructura tipo `vector<cv::KeyPoint>`.

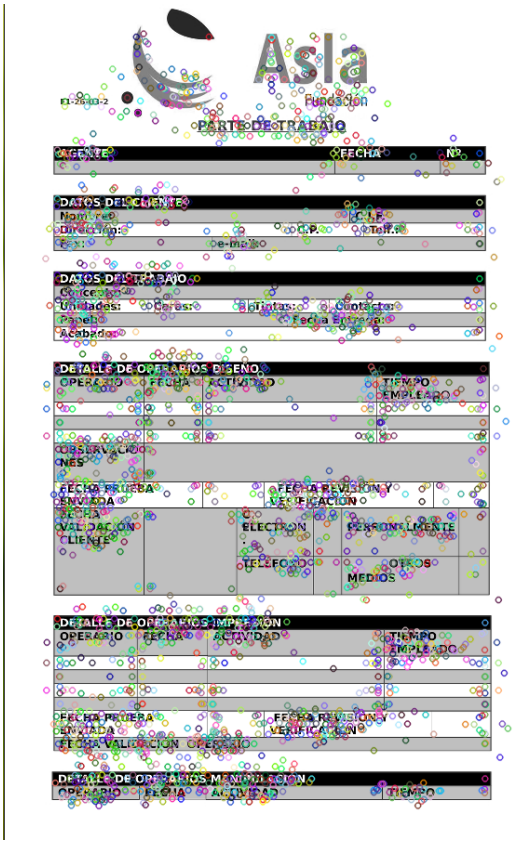


Figura 3.9: Puntos de interés

KeyPoint es una clase genérica que ha sido diseñada para almacenar puntos significativos (con la ubicación, orientación, escala y alguna información adicional).

3.4.3. Generación de Descriptores

La clase **DescriptorExtractor** es la encargada de calcula el vector que describe la característica de un punto significativo para la posterior comparación entre otros puntos de

interés. SURF utiliza el histograma de gradientes, calculado a partir de la cuantificación de los gradientes dentro de una área local. Una zona se divide en subregiones y se calcula el histograma de gradiente en cada una de ellas.

Mediante la función **compute** calcula los descriptores de los puntos clave detectados en la imagen. Un descriptor se representa como un vector de dimensión fija de un tipo básico. La mayoría de los descriptores siguen este patrón, ya que simplifica el cálculo de las distancias entre los descriptores. Por lo tanto, un conjunto de descriptores se representa como un `cv::Mat`, donde cada fila es un descriptor de un punto clave.

3.4.4. Búsqueda de Coincidencias

Para buscar una coincidencia entre varias imágenes, se extraen los descriptores de la imagen de consulta y se accede a la estructura de descriptores de las imágenes referencia con los datos del vector de consulta calculado, devolviendo el vector almacenado más similar.

Si el vector de consulta está vacío, significa que no se han podido extraer descriptores del documento. En el sistema, esto ocurre cuando se coloca la hoja por la parte posterior, que es totalmente blanca y el extractor no puede encontrar puntos de referencia.

El buscador se configura para que utilice la biblioteca FLANN (Fast Library for Approximate Nearest Neighbors) de OpenCV, que contiene una colección de algoritmos para la búsqueda rápida los vecinos más cercanos en grandes conjuntos de datos.

El método de los k vecinos más cercanos *(K-nn)*, es un clasificador supervisado que calcula la probabilidad de que un elemento pertenezca a una clase conocida.

Para su utilización, se configura la clase **Flann** con dos parámetros que especifican el algoritmo a utilizar y su parametrización.

- El primero es **IndexParams** donde se define el índice de búsqueda que se construirá. Para SURF, se recomienda la utilización de árboles k-dimensionales (kd-tree) aleatorios como estructuras de búsqueda, que permiten realizar búsquedas en paralelo.
- El segundo es **SearchParams**, en el que se indica el número de veces que los árboles del índice deben ser recorridos de forma recursiva. Valores altos dan mayor precisión, pero también conllevan un mayor tiempo de procesado. Se utiliza el valor por defecto de 32.

Dos descriptores se consideran coincidentes si la distancia euclídea entre ellos es baja. Para ello se incluye una condición de filtrado basada en NNDR *(Nearest Neighbor Distance Ratio)* con los resultados del buscador FLANN para conseguir búsquedas más robustas y precisas. Una coincidencia de descriptores se eliminará si la distancia desde el descriptor de la consulta a su primer vecino más cercano es mayor que un valor umbral, entre 0 y 1, multiplicado por la distancia al segundo vecino más cercano. Este umbral, por lo general se establece en 0,8 y obliga a que las coincidencias sean únicas. Este requisito se puede hacer más estrictos, reduciendo del valor de este umbral.

Finalmente, al comparar todas las imágenes almacenadas con la imagen de referencia, la que mayor número de coincidencias haya obtenido, será la candidata para devolver su índice como resultado de la identificación.

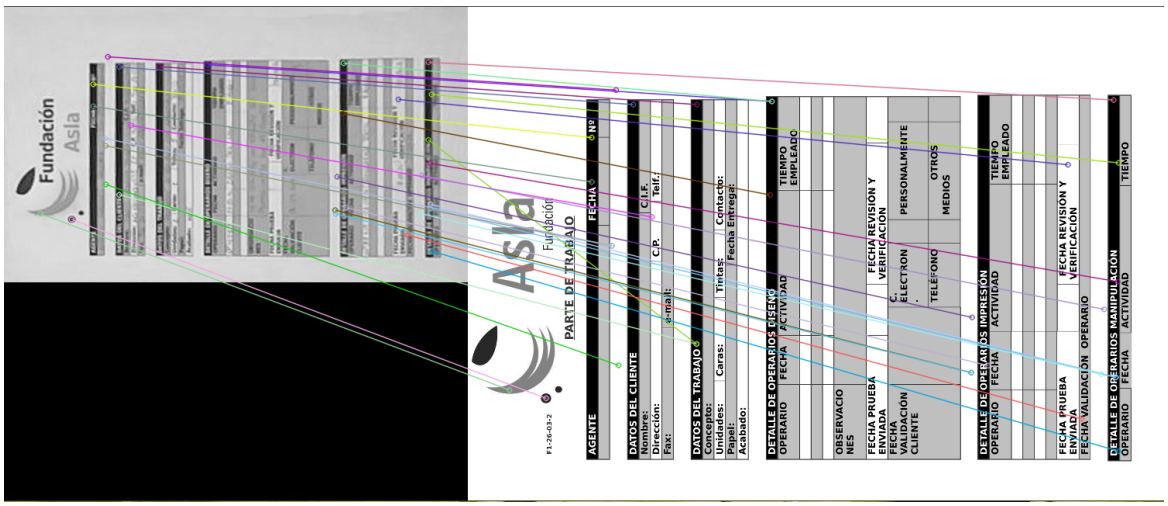


Figura 3.10: Búsqueda de coincidencias de descriptores

3.5. Subsistema de interfaz natural de usuario

Interfaz natural de usuario es un término genérico para una variedad de tecnologías que permiten a los usuarios interactuar con los ordenadores en términos humanos. En este framework, se ha creado una interfaz de usuario basada en zonas de resaltado y botones virtuales que son proyectados directamente sobre el documento. El usuario podrá interactuar directamente en el espacio físico sin utilizar sistemas de mando o dispositivos de entrada tradicionales como sería un ratón, teclado, etc. siendo sustituidos por funciones más naturales como el uso de movimientos gestuales con las manos.

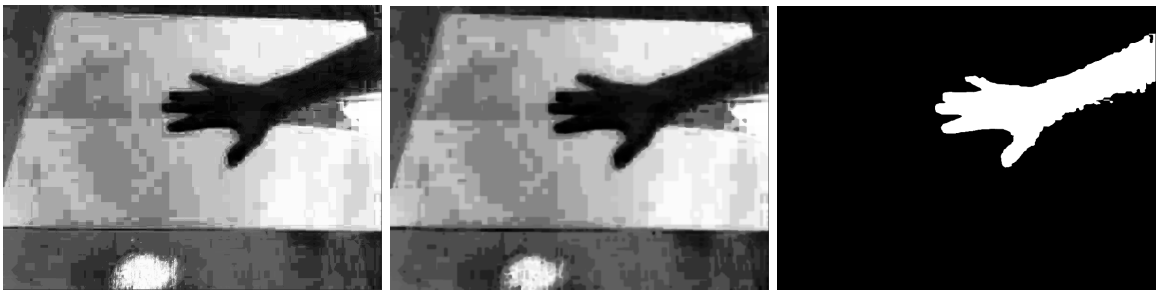
Se ha realizado la implementación del paradigma de superficie táctil interactiva, ampliamente utilizada en los actuales dispositivos portátiles como tablets o smartphones, en la que al pulsar con un dedo sobre la pantalla, reaccione instantáneamente a las acciones del usuario.

Al disponer únicamente de la imagen obtenida por la cámara del sistema, la detección de una pulsación se abstrae a la localización del extremo del dedo índice dentro de una región de interés. Todo ello, realizado mediante técnicas de visión por computador.

3.5.1. Segmentación de la mano

El objetivo de este paso es generar de una máscara que discrimine o diferencie entre los píxeles de una imagen que corresponden al color de la piel y los que no pertenecen. Con está mascara se puede identificar y extraer la mano que aparezca en la imagen y que esté realizando alguna acción sobre el documento a tratar.

Aunque existan diferentes colores de piel, muchos estudios han demostrado que la mayor diferencia se presenta en la intensidad y no el su crominancia, por lo tanto en lugar de utilizar el espacio de color RGB, se utilizará el **espacio de color HSV** (Hue, Saturation, Value), que es más adecuado se aproxima a la percepción humana.



(a) Canal H ecualizado (b) Aplicación del Filtro Bilateral (c) Máscara extraida

Figura 3.11: Segmentación de la mano

Se utilizan los canales de color H y S (matiz y saturación) para detectar la piel. En el canal H, la piel se compone de zonas pequeñas (representadas en negro), y otras mayores (representadas en gris). En primer lugar, se realiza una **ecualización del histograma** del canal H para conseguir una distribución uniforme. Es decir, que exista el mismo número de píxeles para cada nivel de gris del histograma de del canal.

A continuación, se aplica una **erosión morfológica** al canal para reducir los contornos y separar las zonas similares próximas, a la vez que se eliminan las zonas más claras separadas y amplían los pequeños detalles oscuros.

Las operaciones morfológicas, consisten en la alteración de los píxeles de salida de una imagen dependiendo del valor del píxel de entrada y una relación de vecindad definida por un elemento estructural. El elemento estructural define el tamaño y la forma con la que se aplica la operación. En este caso, se aplica un elemento en forma de elipse de tamaño 7x7 píxeles.

Para finalizar el tratamiento previo, se aplica un **filtro bilateral** tanto al canal H como al S. El filtra bilateral es una herramienta muy útil en visión por computador ya que permite suavizar las zonas homogéneas de una imagen manteniendo los bordes.

En este momento la imagen se encuentra preparada para seleccionar píxeles perteneciente a la piel. Se unifican los tres canales y seleccionamos los píxeles que se encuentren entre los umbrales $0 < H < 25$ y $80 < S < 255$

Finalmente se realiza una **limpieza morfológica** mediante las operaciones de apertura y cierre. La apertura suaviza los contornos, elimina pequeñas protuberancias y elimina las conexiones más débiles. El cierre, rellena vacíos en los contornos y elimina pequeños huecos en la imagen.

El resultado de este tratamiento es una imagen en escala de grises, donde las zonas más claras pertenecen a píxeles de piel y los contornos negros son zonas indeterminadas. Estableciendo un umbral, se seleccionan los píxeles candidatos y **se genera la máscara** que corresponderá a la región ocupada por la mano en la imagen.

3.5.2. Análisis de la imagen para la detección de la mano

Una vez que disponemos en una imagen binaria la región de la mano, se aplica un **detector de contornos** sobre la imagen. En caso de no ser único, el contorno que mayor longitud tenga entre los encontrados, corresponderá a la mano.

Según la disposición del prototipo en relación a posición de los documentos (situado a la izquierda de los papeles), establecemos la restricción de que la mano aparecerá por la parte derecha del documento para interactuar con los botones, que también serán proyectados sobre la zona derecha del documento. Además, para la simulación de la pulsación es necesario tener el dedo índice extendido. Por tanto, el extremo del dedo índice corresponde con el **punto del contorno situado más a la izquierda** dentro de la imagen. Seleccionando este punto podemos situar la zona de pulsación dentro de la pantalla.

3.5.3. Cálculo de un punto 3D a partir de un punto en la imagen

La función anterior obtiene el punto del extremo del dedo índice en coordenadas (x, y) de la pantalla. Este dato no sirve para saber donde está colocado el extremo del dedo en relación a las dimensiones reales del papel.

La opción de calcularlo mediante proporcionalidad de los píxeles que forman el papel y sus dimensiones no es válida, ya que la distorsión de la perspectiva generaría un error muy grande.

Para solventar el problema utilizaremos los parámetros intrínsecos y extrínsecos de la cámara calculados de antemano y partiendo de la relación entre los puntos del mundo real y los de la cámara:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K(R \begin{bmatrix} X \\ Y \\ Z_{const} \end{bmatrix} + t) \tag{3.9}$$

donde K es la matriz de la cámara, R la matriz de rotación, t el vector de rotación, y s es el factor de escala de la homografía. $Zconst$ representa la altura donde se están representado los componentes gráficos en relación al papel, que en este caso es 0.

$$R^{-1}K^{-1}s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z_{const} \end{bmatrix} R^{-1}t \tag{3.10}$$

Resolviendo la ecuación anterior sustituyendo los puntos (x, y) de la pantalla en (u, v) para conseguir s, obtenemos el punto 3D relativo al papel, es decir, situando el centro de coordenadas $(0, 0, 0)$ en el centro de la hoja.

3.6. Análisis del rendimiento del sistema

El *profiling* ó análisis de rendimiento, es la observación del comportamiento de un programa utilizando información obtenida desde el análisis dinámico del mismo. Tiene por

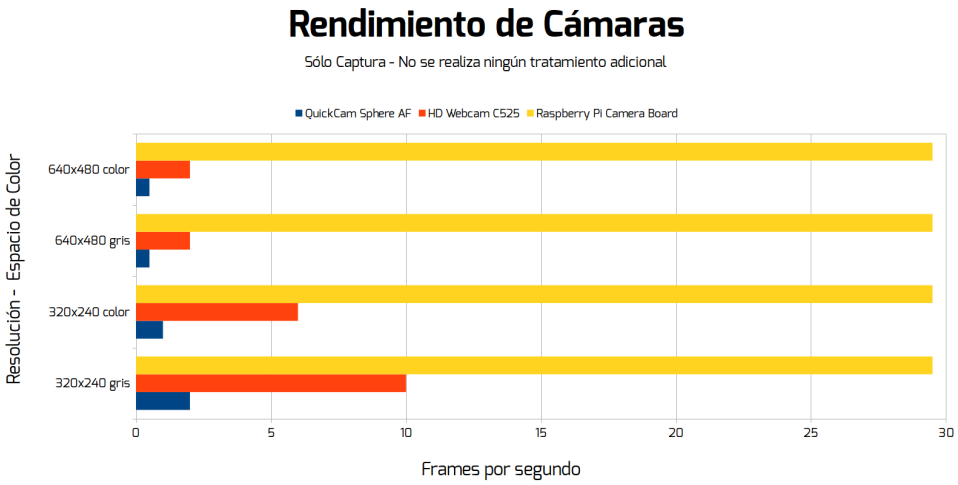


Figura 3.12: Tasa de FPS por modelo de cámara

objetivo averiguar el tiempo dedicado y el consumo de recursos en la ejecución de diferentes partes del software. Se utiliza principalmente para detectar *«cuellos de botella»* del sistema, dónde sea posible llevar a cabo una optimización,

Debido a los limitados recursos y capacidades de los que dispone la Raspberry Pi, durante el desarrollo del framework, se han realizado numerosas mediciones del sistema con el objetivo de tomar decisiones que consiguiesen producir un sistema lo más optimizado posible.

La elección de la cámara era un factor muy importante para garantizar la viabilidad del proyecto. El proceso de captura de imágenes debería ser lo más liviano posible, ya que tiene un uso intensivo a lo largo de la ejecución. Además debe tener un rendimiento, capaz de proporcionar una sensación de tiempo real en el sistema.

En principio se utilizaron cámaras USB, pero la utilización de drivers *video4linux* para la captura de las imágenes, no tenia el rendimiento esperado, presentado las imágenes con *lag* de hasta 2 segundos.

La elección final, fue la utilización de la Raspberry Pi Camera Board, que ofrece una tasa de 29,5 FPS hasta una resolución de 1280x960 a escala de grises y 14 FPS con esa misma resolución, pero esta vez en el espacio de color BGR (propio de OpenCV).

En la figura 3.12 se muestran las medidas realizadas, en *frames por segundo* (FPS), del proceso de captura y representación en vivo de los fotogramas capturados. No se realizar ningún tratamiento adicional a la imagen.

En relación a la resolución de trabajo del proyector, éste cuenta con una resolución nativa de 854x480 en formato 16:9, pero la Raspberry Pi no la soporta. Se eligió una resolución de 1280x720, pero con este tamaño de imagen, el rendimiento disminuía significativamente. También se utilizó 800x600 en formato 16:9, pero la distorsión que realizaba el proyector para mostrar la imagen en formato panorámico no podíamos medirla para realizar una corrección, y los puntos calculados no correspondían con los objetos situados sobre la mesa.

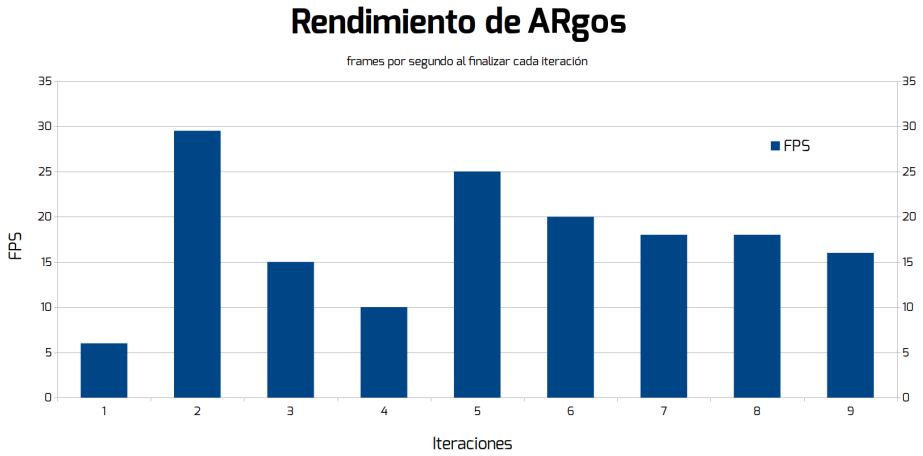


Figura 3.13: Tasa de frames del sistema en cada iteración

La resolución que se estableció finalmente fue de 800x600 en formato 4:3. Se perdía algo de superficie de proyección, pero con aumentar un poco la distancia entre el proyector y la mesa se compensaba esta área, y además, el rendimiento del sistema con esta resolución era adecuado y entraba dentro de los parámetros para un sistema de tiempo real.

Ademas, en cada iteración el software crecía en funcionalidades y complejidad. La Raspberry Pi tenía que soportar, cada vez, una carga superior. En la figura 3.13 se muestra el rendimiento medido del sistema, en FPS, al finalizar cada *sprint*. Se puede observar los malos resultados de la primera iteración al utilizar una cámara web y la recuperación experimentado en el sistema a partir de la iteración 5, que corresponde con la migración de los subsistemas más complejos al servidor.

3.7. Evaluación de los sistemas de tracking

Durante el desarrollo del sistema se han incorporado distintos métodos de tracking para realizar la detección y el seguimiento de la hoja de papel.

Casi todos los algoritmos de tracking obtienen grandes resultados partiendo de imágenes limpias, en alta resolución y con contrastes claramente definidos. Sin embargo, mediante la captura con cámaras de bajo coste, debido a su naturaleza, a la forma en que se realiza la captura y el entorno en que nos encontremos se presentan una serie de dificultades que deben ser tenidas en cuenta:

- **Baja resolución.** Las imágenes obtenidas con las cámaras suelen estar en baja resolución, bien por las limitaciones del sensor, o por que la capacidad de computo del dispositivo que la contiene es limitada.

  El tamaño en píxeles del contorno de la hoja de papel en pantalla corresponde a aproximadamente 820 píxeles.

- **Iluminación no uniforme.** La cámara no tiene control de la iluminación de la escena. En la captura mediante cámaras es normal encontrarse con iluminación no uniforme, varias fuentes de luz con temperaturas de color diferentes, sombras o reflejos que degradan la calidad de la imagen.

  Para las pruebas se ha preescindido de luz natural y se ha utilizado una única fuente de luz colocada en la parte superior del sistema, con el fin de evitar la creación de sombras.

- **Distorsión por perspectiva.** Al realizar las capturas sin estar la cámara paralela al plano en el que se encuentra el documento, se está produciendo una distorsión por perspectiva. Esto provoca que el papel presente distintos tamaños a lo largo de la imagen o que se produzca una deformación que impida el correcto reconocimiento.

  En el entorno de prueba, mediante un trípode se ha mantenido la posición constante del sistema, asi como la del documento de tal forma que la hoja presente siempre las mismas dimensiones en la captura realizada.

- **Zoom y autoenfoque.** Las cámaras actuales están equipadas con sistemas de zoom y autoenfoque. Una captura en la que existan distintos planos de profundidad o una mala iluminación provocará que el sistema de autoenfoque tenga dificultades para estabilizarse y durante ese tiempo las imágenes sean borrosas o fuera de foco.

  Para las pruebas, se ha utilizado una apertura de diafragma suficiente para que todos los puntos se encuentren dentro de foco y se ha desactivado la opcion de autoenfoque.

3.7.1. Tracking mediante detección de rectángulos

La detección de rectángulos fue el primer método de tracking implementado. Sirvió para obtener, mediante segmentación de la hoja de papel, la posición de sus 4 esquinas en píxeles de pantalla.

Este procedimiento es muy sensible a la iluminación. El tipo de luz (natural, fluorescente, incandescente,...), intensidad de la luz natural (luz por la mañana, al mediodia o por la tarde), dirección de la luz (crea sombras en uno u otro sentido), los reflejos producidos por las superficies se convierten en interferencias en forma de grandes manchas en la imagen. Incluso se han detectado variaciones en la detección al encontrase varias personas entre el sistema y una ventana.

Es un método de tracking bastante rápido, pero como se observa en la tabla 3.1, no permite ninguna oclusión.

3.7.2. Tracking mediante envoltura convexa (Convex Hull)

Tras un estudio de técnicas alternativas, se opta por implementar el cálculo de la envoltura convexa del contorno. Además de ser un algoritmo más reducido, la mayor ventaja que aporta este método es que tolera ciertos solapamientos en los bordes del papel y siempre que no se queden esquinas ocultas.

Para optimizar la función, se sustituyó el método de binarización de la imagen, de umbralización adaptativa por el método de Canny. Con este cambio, se obtiene un algoritmo más robusto, más tolerante a la iluminación y que devuelve bordes más finos, con lo que también aumenta la precisión del sistema.

% Oclusión	Píxeles Aprx.	Lados Afectados	Lados Ocultos Totalmente	Esquinas Ocultas	¿Detección?	Tiempo de Detección
0	0	0	0	0	**Si**	7,052 ms
10	82	1	0	0	No	X
10	82	2	0	1	No	X
20	164	1	0	0	No	X
20	164	3	1	2	No	X
30	246	1	1	2	No	X
30	246	2	0	2	No	X
40	328	3	0	2	No	X
40	328	3	1	1	No	X
50	410	2	2	3	No	X
50	410	3	0	2	No	X
60	492	3	0	3	No	X
60	492	3	2	3	No	X
70	574	3	0	3	No	X
70	574	3	2	4	No	X
80	656	4	0	4	No	X
80	656	4	3	4	No	X
90	738	4	3	4	No	X
100	820	4	4	4	No	X

Tabla 3.1: Tracking mediante detección de rectángulos (500 frames)

Cómo se puede comprobar en la tabla 3.2, los tiempos son similares, y por tanto se toma como base como sistema de tracking sin oclusiones.

3.7.3. Tracking mediante búsqueda en el espacio de Hough

Analizando los solapamientos que se realizan sobre el papel en función de las posiciones de la mano, se observa que normalmente quedan visibles partes de los lados del papel. Se decide realizar la búsqueda de segmentos rectos mediante la transformada de Hough y generar las posibles combinaciones de 4 segmentos del conjunto de rectas detectadas. Aquellos que cumplan las restricciones de formar una hoja de papel, serán los posibles candidatos.

Este tercer método de detección es más costoso que el anterior, por lo que se determinó que de forma predeterminada, el sistema utilizase la detección basada en la envoltura convexa cuando no existiesen solapamientos, y en caso de fallo, se aplicase esta nueva función.

Esta nuevo algoritmo detectaba el papel aún existiendo grandes solapamientos, tanto en bordes como en esquinas del documento, llegando a funcionar con incluso un 80 % de oclusión del contorno. La única restricción, como se puede observar en la tabla 3.3, es que sean visibles (aunque sean muy pocos píxeles) los cuatro lados del papel.

En los casos marcados con un asterisco (*) el sistema ha identificado la hoja de papel, pero de manera erronea. Ya que al no ser visible uno de los lados, no ha considerado la superficie completa.

% Oclusión	Píxeles Aprx.	Lados Afectados	Lados Ocultos Totalmente	Esquinas Ocultas	¿Detección?	Tiempo de Detección
0	0	0	0	0	**Si**	6,969 ms
10	82	1	0	0	**Si**	7,025 ms
10	82	2	0	1	No	X
20	164	1	0	0	**Si**	6,934 ms
20	164	3	1	2	No	X
30	246	1	1	2	No	X
30	246	2	0	2	No	X
40	328	3	0	2	No	X
40	328	3	1	1	No	X
50	410	2	2	3	No	X
50	410	3	0	2	No	X
60	492	3	0	3	No	X
60	492	3	2	3	No	X
70	574	3	0	3	No	X
70	574	3	2	4	No	X
80	656	4	0	4	No	X
80	656	4	3	4	No	X
90	738	4	3	4	No	X
100	820	4	4	4	No	X

Tabla 3.2: Tracking mediante envoltura convexa de Hull (500 frames)

% Oclusión	Píxeles Aprx.	Lados Afectados	Lados Ocultos Totalmente	Esquinas Ocultas	¿Detección?	Tiempo de Detección
0	0	0	0	0	**Si**	21,422 ms
10	82	1	0	0	**Si**	22,485 ms
10	82	2	0	1	**Si**	23,334 ms
20	164	1	0	0	**Si**	23,767 ms
20	164	3	1	2	No*	16,931 ms
30	246	1	1	2	No*	18,783 ms
30	246	2	0	2	**Si**	23,825 ms
40	328	3	0	2	**Si**	25,288 ms
40	328	3	1	1	No	X
50	410	2	2	3	No	X
50	410	3	0	2	**Si**	29,943 ms
60	492	3	0	3	**Si**	32,701 ms
60	492	3	2	3	No	X
70	574	3	0	3	**Si**	35,925 ms
70	574	3	2	4	No	X
80	656	4	0	4	**Si**	39,319 ms
80	656	4	3	4	No	X
90	738	4	3	4	No	X
100	820	4	4	4	No	X

Tabla 3.3: Tracking mediante búsqueda en el espacio de Hough (500 frames)

4

Conclusiones y trabajo futuro

En este capítulo se realiza un análisis sobre los objetivos alcanzados durante el desarrollo del presente Trabajo Fin de Máster. A continuación, se exponen nuevos puntos de vista, que se pueden tratar en futuros trabajos para mejora ó ampliación del sistema, indicando una posible implementación y una estimación del coste temporal en caso de realizarlos.

El objetivo principal de este trabajo es el desarrollo del sistema de captura, tracking, registro e identificación de documentos dentro del *Proyecto ARgos*. Con tal fin, se ha construido un prototipo real basado en una Raspberry Pi, con una cámara de bajo coste y un proyector portátil, que montado sobre un trípode, muestra información visual alineada sobre un documento que se encuentre dentro de la zona de proyección. Para efectuar esta visualización, se tiene en cuenta el posicionamiento 3D relativo entre el documento y el sistema cámara-proyector, para que el registro de la amplificación visual sea perfecto.

Además, el sistema responde a distintas acciones que el usuario pueda efectuar sobre el espacio físico, ampliando la información relacionada que sea relevante a la acción realizada.

Para conseguir una experiencia totalmente inmersiva, se han desarrollado técnicas de tracking y registro que permiten el cálculo de la *pose* (rotación y translación del documento en el espacio 3D) en tiempo real. Esto nos permite, que podamos ofrecer al usuario la información proyectada sobre la hoja de papel, independiente de su posición, rotación o plano en el que se encuentre. Siempre y cuando, no se salga de los límites de la zona de proyección.

Para el cálculo de los parámetros intrínsecos y extrínsecos tanto de la cámara como del proyector, se ha desarrollado una aplicación externa que permite un calibrado rápido y sencillo del sistema, mediante un patrón de tipo tablero de ajedrez, cada vez que sea necesario. Una vez realizada la calibración, obtenemos tres ficheros XML que son lo necesarios para poder realizar los cálculos de registro en el prototipo.

El usuario puede interactuar directamente en el espacio físico sin utilizar sistemas de mando tradicionales, como ratones o teclados. Sólo es necesario encender el prototipo, y según el usuario vaya colocando documentos, los desplace por la zona de proyección, realice

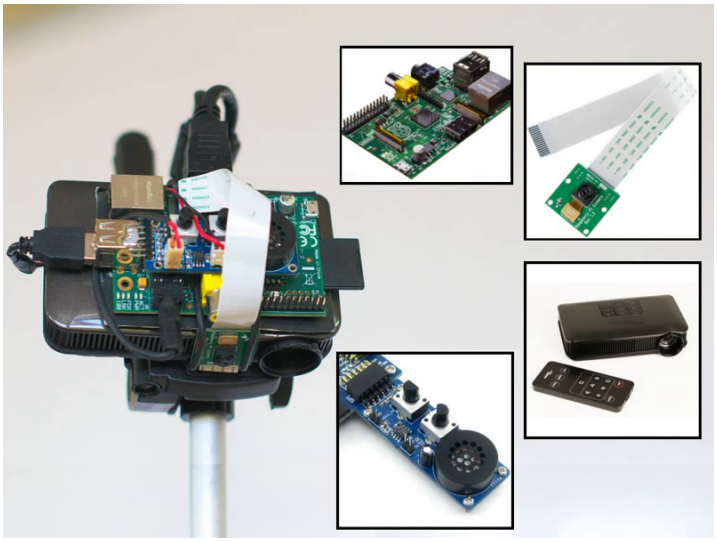


Figura 4.1: Prototipo de ARgos

interacciones señalándolo ó girándolo para mostrar la cara posterior, el sistema responderá automáticamente a la acción realizada.

En cuanto a posibles ampliaciones o mejoras de la arquitectura presentado en este trabajo y líneas de trabajo futuro, podemos destacar las siguientes:

Tracking mediante aproximaciones *top-down*. Estas técnicas se basan en la estimación mediante modelos de movimiento basados en filtros bayesianos para predecir la posición de la cámara. A partir de la posición obtenida, se buscan referencias en la imagen que puedan corregir y ajustar la predicción.

Los filtros bayesianos a utilizar pueden clasificarse en dos tipos. Aquellos que trabajan con modelos de movimiento gausianos, se denominan Filtros de Kalman, y los que, por las características del ruido no pueden ser modelados mediante modelos gausianos, y se implementan mejor mediante Filtros de Partículas.

Estos métodos proporcionan robustez al proceso de tracking, ya que permite seguir detectando la hoja de papel, aun cuando exista una gran oclusión de la misma.

Para la implementación de esta mejora, habría que valorar, además, el tiempo para analizar ambos enfoques y determinar cual de ellos se adapta mejor a las características de su desplazamiento por la zona de proyección.

Detección de páginas de texto mediante LLAH o similares. La implementación actual de detección de documentos está basado en la búsqueda de coincidencias de características locales, ya que estos métodos obtienen mayor precisión de acierto en la detección de documentos con información semiestructurada, como es el caso de facturas o formularios.

Sería interesante, como ampliación al sistema de detección, que se pudiesen reconocer documentos con alto contenido textual sin una estructura predeterminada de antemano. Para ello, es necesario la implementación de algún algoritmo de detección basado en las características inherentes de la estructura del documento y el texto que contiene.

LLAH [23] tiene una alta escalabilidad y el esquema de indexación y recuperación empleado es extremadamente rápido. Los autores han confirmado que LLAH tiene una tasa de acierto del 99 % y con un tiempo de procesamiento de 50 ms en una base de datos de 20 millones de páginas.

Existen varias implementaciones optimizadas sobre LLAH para salvar las limitaciones iniciales del algoritmo. Sería un trabajo futuro la tarea de buscar sólo las publicaciones que se hayan hecho sobre LLAH e intentar crear una implementación conjunta con todas las optimizaciones.

Estimación inteligente de parámetros según la iluminación existente. Los algoritmos de filtrado (suavizado de imágenes, detección de bordes,...) se definen a través de parámetros que determinan su comportamiento y el resultado de la imagen obtenida. Normalmente se determinan por decisión humana y permanecen estáticos durante toda la ejecución del programa. Elegidos en función de los mejores resultados obtenidos en unas circunstancias determinadas, las variaciones en la iluminación de la escena provocan que un parámetro que en un principio se pudo considerar óptimo, al instante no sea válido. También nos encontramos el caso de que exista un rango de valores que nos satisfagan.

Se propone por tanto la creación de un sistema mediante técnicas de *softcomputing* que analice la escena actual y determine qué valores son los más adecuados para obtener un resultado conocido o esperado, en una serie de filtros utilizados en el framework.

Las ventajas de realizar esta implementación proporcionaría, que los métodos de detección fuesen más robustos y tolerantes a los cambios de iluminación que se puedan producir durante la utilización del sistema.

Calibrado del sistema cámara-proyector mediante luz estructurada. Otra técnica para el calibrado de proyectores es el propuesto por Daniel Moreno y Gabriel Taubin, basado luz estructurada, en el paper «***Simple, Accurate, and Robust Projector-Camera Calibration***».

La implementación, consistiría en los siguientes pasos a realizar [21]:

- Detectar la localización de las esquinas de un patrón de tablero de ajedrez en cada una las orientaciones capturados.
- Estimar los componentes de luz directa y global.
- Decodificar los patrones de luz estructurada.
- Calcula una homografía local para cada una de las esquinas del patrón detectado.
- Trasladar las posiciones de las esquinas a coordenadas del proyector mediante homografías locales.
- Obtener los parámetros intrínsecos de la cámara utilizando las posiciones de las esquinas en la imagen.
- Obtener los parámetros intrínsecos del proyector con las posiciones trasladadas al sistema de referencia del proyector.
- Ajustar los parámetros intrínseco de la cámara y el proyector y obtener los parámetros extrínsecos del sistema en conjunto.
- Opcionalmente, se puede realizar una optimización conjunta de todos los parámetros (intrínsecos y extrínsecos).

Este método no requiere ningún equipamiento especial y, según sus autores, es más preciso que otras técnicas de calibrado, ya que utilizan el modelo pinhole completo con distorsión radial.

En base a la experiencia en el desarrollo del proceso de calibrado, esta historia se podría plantear para realizarse en un sprint de 3 ó 4 semanas de duración.

Utilización de cámaras con sensor de profundidad. Sustituyendo la cámara principal, por otra equipada con sensores de infrarrojos (IR) capaces de medir la profundidad de la escena, obtendríamos una mayor precisión y la posibilidad de aumentar la variedad de gestos reconocidos.

Por ejemplo, realizar la acción de click (o doble click) con los dedos sobre la superficie de la mesa, ó la simulación de una superficie multitáctil.

El coste temporal de esta propuesta puede ser mayor que las anteriores. Habría que estudiar las bibliotecas existentes para la utilización del dispositivo (por ejemplo, kinect), proporcionar un nuevo sistema de calibrado para la cámara IR, y también, seria necesaria la modificar los métodos de reconocimiento gestual, que tendrían un enfoque distinto, debido a la información adicional de la cámara de profundidad.

A

Asignaturas cursadas

En este Apéndice se muestran todas las asignaturas cursadas por el alumno en el Máster de Tecnologías Informáticas Avanzadas impartido por la UCLM, haciendo un resumen de los conocimientos adquiridos en cada una de ellas.

Metodologías y Técnicas de Investigación en Informática

- **Profesorado:** D. Mario Gerardo Piattini Velthuis, D. José Antonio Cruz Lemus y Dña. Marcela Genero Bocco.
- **Cuatrimestre:** Primer cuatrimestre.
- **Resumen:** El objetivo de la asignatura es presentar los métodos de investigación más adecuados para la validación y contrastación de las hipótesis de investigación, así como de ofrecer a los alumnos un conjunto de técnicas que les facilite la realización de su tesis doctoral y escribir una comunicación científica.

  De manerá práctica, se forma al alumno, para realizar revisiones de la literatura existente mediante el enfoque de una revisión sistemática. Este método, nos permite mediante la aplicación de un protocolo, identificar, evaluar e interpretar todos los estudios importantes o significativos (llamados estudios primarios) para una pregunta de investigación en particular.

  La evaluación de la asignatura, consiste en la realización y exposición de una Revisión Sistemática sobre un tema propuesto que sea relevante a la investigación personal que está realizando de cada alumno.

  Como trabajo personal, definí un Mapeo Sistemático sobre la literatura existente entorno a algoritmos de visión por computador para la identificación y recuperación de documentos impresos mediante dispositivos móviles. Parte del estudio de este trabajo se encuentra reflejado en el estado del arte del presente documento.

Nuevos Paradigmas en Interacción-Persona Computador

- **Profesorado:** Dña. María Teresa López Bonal, D. José Pascual Molina Massó, D. Miguel Ángel Fernández Graciani y D. Antonio Fernández Caballero.
- **Cuatrimestre:** Primer cuatrimestre.
- **Resumen:** El curso aborda los siguientes temas:
  - **Visión artificial.** Comienza con una introducción a la visión artificial, realizando un estudio de todos los parámetros que guían el reconocimiento de patrones en imágenes, tanto estáticas como en movimiento (detección de movimiento), adquiriendo conocimientos y técnicas que nos permiten procesar las imágenes en busca de información. Del mismo modo se aborda la representación en 3D en escenas, concluyendo con el papel de la visión artificial en la interacción persona-computador.
  - **HCI en la red.** Las interfaces persona-computador han sabido adaptarse a los tiempos actuales, en los que Internet juega un papel importante. Entran en juego nuevas formas de interacción en forma de red social, nuevos usuarios, etc. Se toma conciencia de las características que debe tener una interfaz que pretenda ser usada por el mayor número de usuarios posible.
  - **Interfaces de usuario 3D.** Comenzando con una revisión a la historia de las interfaces 3D, se estudia la metodología que sigue este tipo de interfaces, experimentando con alguno de los lenguajes que permiten su diseño. Se estudian nuevos paradigmas de interacción basados en la tecnología 3D como puede ser la realidad aumentada y la realidad virtual.

Sistemas Avanzados de Interacción Persona-Computador: Sistemas Colaborativos y Computación Ubícua

- **Profesorado:** Dña. Ana Isabel Molina Díaz, D. Miguel Ángel Redondo Duque y D. Crescencio Bravo Santos.
- **Cuatrimestre:** Primer cuatrimestre.
- **Resumen:** El objetivo de esta asignatura fue la del aprendizaje de los distintos sistemas de interacción avanzados existentes, relacionados con la computación ubícua. Además, se presentaron distintas técnicas y *frameworks* para el desarrollo y análisis de estos sistemas.

  El primer y segundo bloque, impartido por Miguel Ángel Redondo, enseñaba los fundamentos de la computación ubícua y los sistemas colaborativos. Para la evaluación de esta parte, se realizaron una serie de trabajos que posteriormente eran defendidos en clase:

  - El estudiante debe buscar la definición de varios conceptos fundamentales relacionados con los sistemas colaborativos, para posteriormente ser contrastados con los encontrados por el resto de compañeros.

- Se propone un trabajo para realizar una recopilación de artículos científicos, que se puede escoger entre distintas temáticas. En mi caso, se optó por la búsqueda de publicaciones sobre tecnologías actuales para el desarrollo de sistemas que soportan computación ubicua como paradigma de interacción.
- Realizar una clasificación de los sistemas de soporte a actividades en grupo que se han visto durante este bloque temático según el artículo de Grudin (1994), en el que se hace una versión extendida de la Taxonomía del Tiempo y el Espacio de Johansen (1991).

El tercer bloque, que fue impartido por el profesor Crescencio Bravo, se estudiaban las bases teóricas para el diseño y desarrollo de sistemas colaborativos. Se presentaron distintos *frameworks* y herramientas como *JSDT*. Además, se estudió el caso real de una herramienta, COLLECE, a la que se le aplicaron los conocimientos adquiridos. El ejercicio asignado para este bloque consistió en análisis de dos aplicaciones, el juego de ajedrez distribuido *ChessBase* y *Twitter*, para identificar el soporte colaborativos que ofrece a cada uno de ellos.

El cuarto bloque, impartido por la profesora Ana Isabel Molina, explicaba los diferentes conceptos relacionados con la evaluación de los sistemas colaborativos, y la problemática asociada a ello. El trabajo para esta parte consistió en una evaluación de la interacción y la colaboración en el diseño de interfaces gráficas de usuario para dispositivos móviles.

Para finalizar, se tuvo que realizar una defensa pública en el que se explicaba, a modo de repaso, todos los conocimientos adquiridos y trabajos realizados a lo largo de la asignatura, y relacionarlos con la investigación principal del alumno.

Grid Computing

- **Profesorado:** Dña. María Emilia Cambronero Piqueras y D. Fernando López Pelayo.
- **Cuatrimestre:** Segundo cuatrimestre.
- **Resumen:** El temario de esta asignatura, impartida en el Campus de Albacete, presenta dos bloques bien diferenciados:

  El primer bloque, impartido por María Emilia Cambronero Piqueras presenta los conceptos relacionados con la especificación y verificación de contratos electrónicos. Dichos conceptos engloban la especificación formal de contratos electrónicos y el proceso de verificación a seguir en este tipo de procesos.

  El segundo bloque, impartido por el profesor Fernando López Pelayo, expone el análisis de rendimiento que puede aplicarse a grandes sistemas a partir de técnicas formales se centran en álgebras de procesos como ROSA (Reasoning On Stochastic Algebra). Mediante estos procesos se pueden realizar mediciones de rendimiento y analizar los distintos camininos de ejecución de los algoritmos. Gracias a ello, es posible detectar cuellos de botella, o puntos de posible paralelización y aceleración de algoritmos.

  La evaluación de la asignatura consistió en la especificación de los requisitos del contrato electrónico de una plataforma de *crowdfunding*, su modelado mediante diagramas C-O y la realización de una aproximación a la lógica TCTL de cada uno de los requisitos iniciales para poder validar el modelo mediante herramientas verificación como UPPAAL.

Técnicas de Softcomputing

- **Profesorado:** D. Luis Jiménez Linares, D. Javier Alonso Albusac Jiménez, D. José Jesús Castro Sánchez y D.Juan Moreno García.
- **Cuatrimestre:** Segundo cuatrimestre.
- **Resumen:**Esta asignatura, dividida en tres bloques, introduce el concepto de *Softcomputing* para la resolución de problemas demasiado complejos como para poder ser tratados de forma algorítmicamente tradicional. Estas técnicas se basan principalmente en el uso de la *lógica difusa*, por lo que se aprenden los conceptos de esta.

  En el primer bloque, impartido por Luis Jiménez, se enseñan las bases de la lógica difusa, los conjuntos difuso y los diferentes métodos de *fuzzificación* y *defuzzificación* existentes.

  El segundo bloque, impartido por Javier Albusac, se estudia el framework *pyfuzzy*, para la definición de sistemas difusos en el lenguaje *Python*. Mediante este framework se pueden implementar reglas, conjuntos difusos y operaciones de lógica difusa.

  El tercer bloque, impartido por José Jesús Castro, explica el proceso de desarrollo de sistemas difusos, mediante la definición de las reglas y los conjuntos que van a representar el sistema.

  El cuarto bloque, impartido por el profesor Juan Moreno García, trata sobre el modelado de sistemas dinámicos, la inducción automática y el aprendizaje de los sistemas difusos, utilizando un conjunto de casos reales.

  Por último, como trabajo final para la asignatura, se propuso la redacción de un artículo sobre Relaciones Difusas, exponiendo en que consisten, que aplicaciones tienen y recogiendo la publicaciones más recientes sobre el tema.

Cognición y Colaboración

- **Profesorado:** Dña. Carmen Lacave Rodero, D. Jose Angel Olivas y D. Jesús Serrano Guerrero.
- **Cuatrimestre:** Segundo cuatrimestre.
- **Resumen:** Esta asignatura está dividida en tres bloques de temario:

  El primer bloque, impartido por Carmen Lacave, explicó conceptos de Redes Bayesianas para modelar el conocimiento probabilista, y las formas de inferirlo a partir de ellas. También se presentaba el programa Elvira para realizar este tipo de inferencias.

  El segundo bloque, impartido por el profesor Jesús Serrano, consistió en el aprendizaje de los sistemas de recomendaciones, las técnicas de análisis de redes sociales y la mineria de opiniones.

  El tercer bloque, impartido por José Ángel Olivas, explica las diferentes técnicas existentes para la recuperación inteligente de información y procesamiento de lenguaje natural.

  El trabajo final de la asignatura consistió en escoger un tema de investigación que se enmarcase en uno de los tres bloques anteriores. La elección, consistió en realizar

un estudio sobre las distintas técnicas de *machine learning* existentes para el seguimiento de objetos en secuencias de imágenes, incluso cuando se presentan grandes oclusiones.

B

Currículum Vítae

Datos personales

- **Nombre:** Manuel Hervás Ortega
- **Titulación:** Graduado en Ingeniería Informática
- **Email:** *hervas.manuel@gmail.com*
- **Teléfono:** 676 834 017.
- **Fecha de nacimiento:** 25 de Agosto 1981.
- **Dirección:** Avda. Primero de Mayo 11 Bloque 2 1A. Miguelturra (Ciudad Real) C.P: 13170.

Situación actual

- Ingeniero Mainframe en Axa Seguros. Adaptación de sistema de recibos en zSeries para dar soporte a nuevos productos dentro de Programas Internacionales.
- Cursando Máster en Tecnologías Informáticas Avanzadas en la Escuela Superior de Informática(Campus de Ciudad Real) por la Universidad de Castilla-La Mancha.

Formación académica

- **2012-2015** Grado en Ingeniería Informática por la Escuela Superior de Informática (Campus de Ciudad Real - UCLM).

- **1999-2004** Ingeniería Técnica en Informática de Sistemas por la Escuela Superior de Informática (Campus de Ciudad Real - UCLM).

Experiencia laboral

- **2015** Ingeniero Software en Banco Popular. Desarrollo de una aplicación para la creación de informes financieros para la comisión de riegos.
- **2013-2014** Ayudante de Investigación en el Grupo de Investigación AIR de la Universidad de Castilla la Mancha. Trabajando en el proyecto ARgos para realizar la interfaz de un sistema de gestión de documentos impresos basado en Realidad Aumentada y Visión por Computador orientado a personas con discapacidad.
- **2011-2013** Mainframe DevOps en Santander Global Banking & Markets. Desarrollo de la capa de conexión contable de tesorería. Mantenimiento, optimización y resolución de incidencias de la arquitectura contable en entorno de producción. Planificación, integración y despliegue de software en entornos de producción.
- **2008-2011** Jefe de Equipo en Indra Sofware Labs para Mapfre. Gestión y dirección de 6 analistas-programadores, adaptación de nueva BBDD de personas en seguros de ahorro y mantenimiento de aplicaciones de seguros colectivos.
- **2007-2008** Analista en Indra Software Labs. Elaboración de modelos de procesos, diseños técnicos y pruebas de integración y responsable de la línea de outsourcing entre España y Filipinas para la externalización de desarrollos para el Banco Santander.
- **2005-2007** Ingeniero de Software en Soluziona. Construcción y pruebas unitarias de aplicaciones zSeries COBOL-CICS-DB2 en modelo de factoría de software para el Banco Santander.
- **2004-2005** Ingeniero de Soporte Técnico en el Ministerios de Educacíon. Coordinación de técnicos desplazados y soporte al usuario (Escuelas, Institutos, Escuelas de Idiomas y Centros de Adultos).

Cursos de formación y certificaciones

Idiomas

- Español (nativo)
- Inglés (Nivel B1 MERC)

Bibliografía

[1] Mitsuru Ambai and Yuichi Yoshida. Card: Compact and real-time descriptors. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 97–104, Washington, DC, USA, 2011. IEEE Computer Society.

[2] Sunil Arya, David M. Mount, Nathan S. Netanyahu, Ruth Silverman, and Angela Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. ACM*, 45(6):891–923, November 1998.

[3] Ronald. T. Azuma. A survey of augmented reality. *Presence: Teleoperators and Virtual Environments*, 6(4):355–385, 1997.

[4] H. Bay, A. Ess, and T. Tuytelaars. Speeded-up robust features. *Computer Vision and Image Understanding*, 110(3):346–359, June 2008.

[5] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *Proceedings of the 11th European Conference on Computer Vision: Part IV*, ECCV'10, pages 778–792, Berlin, Heidelberg, 2010. Springer-Verlag.

[6] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[7] Andrew I. Comport, Éric Marchand, and François Chaumette. A real-time tracker for markerless augmented reality. In *Proceedings of the 2Nd IEEE/ACM International Symposium on Mixed and Augmented Reality*, ISMAR '03, pages 36–, Washington, DC, USA, 2003. IEEE Computer Society.

[8] Amaury Dame and Éric Marchand. Accurate real-time tracking using mutual information. In *ISMAR*, pages 47–56. IEEE, 2010.

[9] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the Twentieth Annual Symposium on Computational Geometry*, SCG '04, pages 253–262, New York, NY, USA, 2004. ACM.

[10] Michael Donoser, Peter Kontschieder, and Horst Bischof. Robust planar target tracking and pose estimation from a single concavity. In *Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality*, ISMAR '11, pages 9–15. IEEE Computer Society, 2011.

[11] Nate Hagbi, Oriel Bergig, Jihad El-Sana, and Mark Billinghurst. Shape recognition and pose estimation for mobile augmented reality. *IEEE Transactions on Visualization and Computer Graphics*, 17(10):1369–1379, 2011.

[12] Chris. Harris and Mike. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, pages 147–151, 1988.

[13] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[14] Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1-3):185–203, 1981.

[15] H. Kato and M. Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Augmented Reality, 1999. (IWAR '99) Proceedings. 2nd IEEE and ACM International Workshop on*, pages 85–94, 1999.

[16] Vincent Lepetit and Pascal Fua. Keypoint recognition using randomized trees. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(9):1465–1479, September 2006.

[17] Stefan Leutenegger, Margarita Chli, and Roland Y. Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 2548–2555, Washington, DC, USA, 2011. IEEE Computer Society.

[18] D. Lowe. Distinctive image features from scale-invariant key-points. *Intl. Journal of Computer Vision*, 60:91–110, 2004.

[19] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'81, pages 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.

[20] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In Paul L. Rosin and A. David Marshall, editors, *BMVC*. British Machine Vision Association, 2002.

[21] Daniel Moreno and Gabriel Taubin. Simple, accurate, and robust projector-camera calibration. In *Proceedings of the 2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission*, 3DIMPVT '12, pages 464–471, Washington, DC, USA, 2012. IEEE Computer Society.

[22] Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In Alpesh Ranchordas and Helder Araújo, editors, *VISAPP (1)*, pages 331–340. INSTICC Press, 2009.

[23] Tomohiro Nakai, Koichi Kise, and Masakazu Iwamura. Real-time retrieval for images of documents in various languages using a web camera. In *Proceedings of the 2009 10th International Conference on Document Analysis and Recognition*, ICDAR '09, pages 146–150, Washington, DC, USA, 2009. IEEE Computer Society.

[24] Raphael Ortiz. Freak: Fast retina keypoint. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '12, pages 510–517, Washington, DC, USA, 2012. IEEE Computer Society.

[25] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua. Fast keypoint recognition using random ferns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(3):448–461, 2010.

[26] Jun Park, Bolan Jiang, and Ulrich Neumann. Vision-based pose computation: Robust and accurate augmented reality tracking. In *Proceedings of the 2Nd IEEE and ACM International Workshop on Augmented Reality*, IWAR '99, pages 3–, Washington, DC, USA, 1999. IEEE Computer Society.

[27] Jun Park, Suya You, and Ulrich Neumann. Natural feature tracking for extendible robust augmented realities. In *Proceedings of the International Workshop on Augmented Reality : Placing Artificial Objects in Real Scenes: Placing Artificial Objects in Real Scenes*, IWAR '98, pages 209–217, Natick, MA, USA, 1999. A. K. Peters, Ltd.

[28] Youngmin Park, Vincent Lepetit, and Woontack Woo. Esm-blur: Handling & rendering blur in 3d tracking and augmentation. In *Proceedings of the 2009 8th IEEE International Symposium on Mixed and Augmented Reality*, ISMAR '09, pages 163–166, Washington, DC, USA, 2009. IEEE Computer Society.

[29] Edward Rosten and Tom Drummond. Fusing points and lines for high performance tracking. In *Proceedings of the Tenth IEEE International Conference on Computer Vision - Volume 2*, ICCV '05, pages 1508–1515, Washington, DC, USA, 2005. IEEE Computer Society.

[30] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 2564–2571, Washington, DC, USA, 2011. IEEE Computer Society.

[31] David Marimón Sanjuan. *Advances in Top-Down and Bottom-Up Approaches to Video-Based Camera Tracking*. PhD thesis, École Polytechnique Fédérale de Lausanne, 2007.

[32] Jianbo Shi and Carlo Tomasi. Good features to track. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 94, pages 593–600. IEEE, 1994.

[33] Didier Stricker, Gundrun Klinker, and Dirk Reiners. A fast and robust line-based optical tracker for augmented reality applications. In *Proceedings of the International Workshop on Augmented Reality : Placing Artificial Objects in Real Scenes: Placing Artificial Objects in Real Scenes*, IWAR '98, pages 129–145, Natick, MA, USA, 1999. A. K. Peters, Ltd.

[34] Daniel Wagner, Gerhard Reitmayr, Alessandro Mulloni, Tom Drummond, and Dieter Schmalstieg. Pose tracking from natural features on mobile phones. In *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, ISMAR '08, pages 125–134, Washington, DC, USA, 2008. IEEE Computer Society.

[35] Xiang Zhang, Stephan Fronz, and Nassir Navab. Visual marker detection and decoding in ar systems: A comparative study. In *Proceedings of the 1st International Symposium on Mixed and Augmented Reality*, ISMAR '02, pages 97–, Washington, DC, USA, 2002. IEEE Computer Society.