RECONOCIMIENTO DE COMPORTAMIENTOS DE LA MOSCA DE LA FRUTA USANDO MODELOS OCULTOS DE MARKOV

Por
SAÚL TORRES DEOLARTE
Asesor
Dra. Janeth Cruz Enríquez
Coasesor
Dr. Manuel Martin Ortiz

REQUISITO PARCIAL PARA OBTENER EL
GRADO DE
LICENCIADO EN CIENCIAS DE LA COMPUTACIÓN
EN LA
BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA BUAP
AV. SAN CLAUDIO ENTRE 18 Y 22 SUR, COL. CD. UNIVERSITARIA,. C.P. 72560
MAYO 2008

Agradecimientos

En primer lugar agradezco por la vida que mis padres me han dado y por el calor de hogar que siempre me han proporcionado así como el amor que nunca me han negado, por todo esto hice posible realizar este proyecto de tesis. En segundo lugar agradezco a mis asesores junto con el Laboratorio de Visión del Instituto Nacional de Astrofísica, Óptica y Electrónica porque me brindaron el equipo necesario para realizar las pruebas pertinentes y poder obtener los resultados planteados. Gracias a ello, fue posible evaluar los resultados con ambientes reales. Por último agradezco a todos mis amigos y personas que me rodearon en el trascurso de esta elaboración ya que me apoyaron en todo momento.

Puebla, México
Mayo, 30, 2008.

Saúl Torres Deolarte

Tabla de Contenido

Índice de Figuras

Índice de Tablas

Resumen

Los modelos ocultos de Markov (Hidden Markov Model), HMM por sus siglas en ingles, inicialmente introducidos a finales de los 60's y principios de los 70's, han llegado a ser muy populares en los últimos años debido a dos grandes razones: la primera razón es que los modelos poseen una estructura matemática rica por lo cual es posible formar bases teóricas. La segunda razón es que los modelos, cuando son aplicados apropiadamente, funcionan muy bien en muchas aplicaciones importantes. Por las razones anteriores los modelos ocultos de Markov son utilizados en este proyecto de tesis para el reconocimiento de los comportamientos de la mosca del mediterráneo. Los modelos ocultos de Markov consisten de estados y probabilidades de transición de estados para cada uno de los comportamientos de la mosca del mediterráneo. Estos comportamientos son básicos, por lo cual no es posible separarlos en estados más simples. Debido a esta razón los estados en los modelos ocultos de Markov son directamente los comportamientos una vez definidos. En este proyecto se utiliza el algoritmo de K-Nearest Neighbor para obtener el conjunto de observaciones necesarias como primer argumento del algoritmo de Viterbi. El segundo argumento de este algoritmo es el modelo oculto de Markov. El modelo se obtiene mediante tres parámetros necesarios, la matriz de transición de estados, la matriz de emisión de estados y el vector inicial de probabilidades de los estados. Los tres argumentos necesarios para el modelo son obtenidos de un conjunto de entrenamiento, subconjunto del conjunto de observaciones. El algoritmo de Viterbi es utilizado para suavizar las observaciones obtenidas, es decir, reclasificar las obsevaciones y de esta manera incrementar la precisión en el sistema. La precisión del algoritmo de KNN es del 65.02 % comparado con la reclasificación o suavizado de las observaciones con el modelo oculto de Markov utilizando el algoritmo de Viterbi con una precisión del 75.93 %. El conjunto clasificado con KNN que funciona como conjunto de entrenamiento para el modelo y la obtención de las observaciones tienen una precisión del 65.02 %, por lo que se concluye que el modelo oculto de Markov obtenido en este proyecto tiene una precisión aceptable.

Capítulo 1

Introducción

1.1. Antecedentes y Motivación

La agricultura en nuestro país, es sin duda una de las principales áreas económicas que tenemos, por lo cual debemos apoyarla con la más alta tecnología disponible y así incrementar la investigación y la explotación de nuestros campos fértiles. Sin duda la tecnología se aplica en cualquier rama de las ciencias, en particular en la biología la cual se encarga de estudiar todos los aspectos relacionados con la vida. En este proyecto de tesis se tratará un problema que se tiene en la agricultura, particularmente en cultivos de fruta. Hay muchas causas por las cuales los cultivos de fruta son destruidos y por ende existen pérdidas, que se reflejan en la economía de un país. La mosca del mediterráneo es una de las plagas que ha afectado fuertemente a cultivos en México y EU, por lo cual centros de investigación como la Universidad de California Davis y ECOSUR Campus Tapachula Chiapas, se han dedicado al estudio biológico de la mosca del mediterráneo con la finalidad de poder controlar esta plaga y así evitar pérdidas de grandes extensiones de cultivos, tener una buena cosecha y por lo tanto tener un crecimiento económico mayor. En la actualidad el Instituto Nacional de Astrofísica Óptica y Electrónica desarrolla un sistema automático de reconocimiento de comportamientos de la mosca del mediterráneo mediante algunas técnicas que se describirán más adelante. Dicho sistema tiene una infraestructura particular en el departamento de visión, la cual proporciona información importante acerca de la vida general de la mosca del mediterráneo. El sistema actualmente identifica los primeros 5 de los 9 comportamientos definidos como: comer, tomar agua, volar, caminar, descansar, patas arriba (supine), limpiarse, oviposición y convulsión. La forma de identificar los comportamientos se basa en la distancia, que existe de la posición obtenida en 3D, del centro de masa de la mosca del mediterráneo en un tiempo t con respecto a un tiempo t+1 o a la posición de las regiones donde se encuentra la comida y el agua (posiciones 3D). Cabe mencionar que esta

identificación actualmente se realiza con una precisión aceptable. El objetivo de este proyecto de tesis es determinar técnicas y algoritmos robusto de reconocimiento de los comportamientos de la mosca del mediterráneo basados en los Modelos Ocultos de Markov (HMM) debido a la gran importancia y desempeño que esta técnica presenta en el reconocimiento de comportamientos [1]. Los modelos HMM consisten de estados y probabilidades de transición de estados para cada uno de los comportamientos que se definen. Es necesario determinar cuáles serán los estados o movimientos de la mosca del mediterráneo en cada una de las imágenes que compone la secuencia y etiquetarlos de manera manual. Cabe destacar que los comportamiento que han sido definidos, posterior a este trabajo de tesis, no son posibles separarlos en estados o movimientos más simples ya que estos comportamientos son básicos. Con esta afirmación el etiquetado manual será directamente de cada uno de los comportamientos. A partir de esto, se propone un algoritmo de reconocimiento del comportamiento de la mosca del mediterráneo basado en KNN (K-Nearest Neighbours) y en HMM. Se construirá un vector prototipo de cada uno de los comportamientos que se definan utilizando un algoritmo de agrupamiento “clustering"KNN, con la finalidad de obtener la secuencia de observación para los modelos ocultos de Markov. Por otra parte se obtendrá un HMM basado en transiciones probabilísticas de cada uno de los comportamientos, obtenidas de conjuntos de entrenamiento tanto de observaciones como de clasificación. Una ves obtenida una secuencia de observación, es decir, la salida del clasificador KNN y el HMM, con ayuda del algoritmo de Viterbi se obtendrá un suavizado de tal observación y así incrementar la precisión del sistema.

1.2. Descripción del Problema

Como se ha mencionado, la plaga de la mosca del mediterráneo deja pérdidas económicas enormes, por lo cual hay que controlarla usando métodos efectivos sin que dañen los cultivos. La investigación actual de la mosca del mediterráneo (Ceratitis Capitata) dirigida hacia su erradicación, ha tenido gran relevancia en varias partes del mundo. Es, sin duda, necesario conocer el comportamiento de tal insecto para conocer sus debilidades y así poder controlarlo. Entender cómo la naturaleza trabaja en el medio donde vivimos, nos ha mostrado que su fuerza es muy grande como para erradicar por completo algún fenómeno natural, que para nuestra ambición, será sin duda remunerable. La finalidad de cualquier proyecto que esté basado en problemas naturales, deberían enfocarse al entendimiento de su funcionamiento, nunca en tratar de erradicarlo. Por las razones anteriores, no se mencionará que los estudios que se tienen, tanto de la biología como de la ciencia de la computación, están enfocados a la erradicación de la plaga de la mosca del mediterráneo, si no a su entendimiento y control para un beneficio tanto natural como económico. En la actualidad se tienen dos importantes tipos de control de esta plaga, aspersión de cebo

con insecticida por tierra o por aire y la técnica del insecto estéril (TIE). Esta última técnica tiene la tarea de producir gran cantidad de moscas estériles, para después soltarlas en lugares infestados. La esterilización de las moscas del mediterráneo se realiza aplicando pocas cantidades de radiación sobre ellas [2]. Es sin duda, necesario conocer los diferentes comportamientos que la mosca del mediterráneo tiene a lo largo de su vida, para poder determinar técnicas para el control biológico de la plaga sin dañar a la naturaleza. El motivo principal de este proyecto es conocer de manera automática los comportamientos que tiene la mosca del mediterráneo siendo observada en un laboratorio con condiciones de vida controladas. Actualmente, para el estudio de tal plaga, los biólogos observan moscas del mediterráneo dentro de pequeñas celdas, anotan el comportamiento que ven cada 10 minutos para poder realizar estudios biológicos posteriores. Esta tarea es realmente muy tediosa y no es exacta, debido a que cada 10 minutos la mosca del mediterráneo pudo haber tenido muchos movimientos los cuales pueden pertenecer a comportamientos particulares, razón por la cual se construirá un sistema automático que pueda proporcionar datos confiables a biólogos expertos para que realicen sus tareas de investigación biológica con la finalidad de conocer de manera rápida y confiable las condiciones sobre las cuales pueden controlar a la mosca del mediterráneo de manera que generen las estadísticas necesarias para estudios de reproducción y longevidad de la mosca.

1.3. Objetivos

- El objetivo general de este proyecto es desarrollar un algoritmo que realice el reconocimiento automático de los diferentes comportamientos de la mosca del mediterráneo a partir del procesamiento y análisis de imágenes de la mosca dentro de su habitat de estudio.

Objetivos Específicos

- El primero es obtener un algoritmo de identificación de la posición 2D de la mosca basado en técnicas de análisis de imágenes.
- Un algoritmo de identificación de los comportamientos de la mosca basado en KNN.
- Obtener un modelo oculto de Markov que representen los comportamientos de la mosca del mediterráneo.
- Un algoritmo de identificación de los comportamientos de la mosca basado en HMM.

1.4. Contribuciones

En la actualidad muchas personas están analizando el problema que se tiene desde diferentes puntos de vista, en particular, en esta tesis, el enfoque está en dirección a encontrar modelos que puedan ser útiles para la obtención de más información acerca de la vida de la mosca del mediterráneo.

Se encontrarán modelos HMM para los comportamientos a través de las características 3D que proporciona el laboratorio de visión de la institución. Tales modelos es la principal contribución de esta tesis.

1.5. Organización de la Tesis

El resto de la tesis se organiza de la siguiente manera. En el capítulo 2 se analizan algunos trabajos relacionados a este proyecto de tesis, también se presentan los modelos HMM y los diferentes algoritmos que son necesarios para obtener la secuencia correcta para cada uno de los comportamientos definidos. Está presente, adicionalmente, el algoritmo de clasificación KNN el cual es utilizado para la obtención de las observaciones en las secuencias dadas. En el capítulo 3 se encuentra la propuesta de solución basada en HMM como principal herramienta para el reconocimiento de los comportamientos de la mosca del mediterráneo. En el capítulo 4 se pueden verificar los resultados obtenidos, basados en la técnica propuesta. Se presenta el comportamiento del sistema en varias secuencias diferentes. Al final de este proyecto de tesis se tienen las conclusiones y trabajo a futuro. Finalmente la parte de conclusiones se presenta en el capítulo 5.

Capítulo 2

Fundamentos

2.1. Introducción

Los Modelos Ocultos de Markov o HMM por sus siglas en ingles se basan en la teoría de cadenas discretas de Markov. Se mostrará cómo el concepto de estados ocultos pueden ser utilizados para modelar el comportamiento y cómo las observaciones en los modelos llegan a ser representadas como funciones de probabilidad de los estados. Se discutirán tres problemas fundamentales de los modelos HMM, uno de los problemas es encontrar la secuencia más probable dependiendo de un HMM y un conjunto de observaciones. Se explicará el algoritmo de KNN (K-Nearest Neighbor) debido a que este fue utilizado para la obtención de las observaciones.

2.2. Trabajo Relacionado

Existen muchas aplicaciones en las cuales se utilizan diferentes algoritmos para el reconocimiento de comportamientos como se verá a continuación. Tucker Balch, Zia Khan y Manuela Veloso hacen un estudio de la vida social de colonias de insectos como dominio para el desarrollo y aplicación de herramientas de modelado de sistemas multi-agentes, en el cual se tienen algunos objetivos particulares como: seguimiento simultáneo de múltiples hormigas, reconocimiento de comportamientos individuales y de colonia completa de hormigas y adquisición de nuevos modelos de agentes múltiples y simples. Ellos se basan en un acercamiento híbrido que combina la clasificación basada en color (identificación de regiones) y clasificación basada en movimientos. Para la segmentación de las regiones de interés primero obtienen el fondo utilizando la técnica de substracción del fondo adaptable. Ya obteniendo el fondo con la técnica del fondo adaptable se procede a identificar las regiones

en movimiento haciendo uso de la diferencia entre el fondo adaptable y el imagen actual. Esta técnica de substracción tiene como desventaja la asociación de objetos en movimiento. Para resolver este problema se utiliza una distancia mínima entre dos regiones en un tiempo *t* y *t+1*. Asumiendo que la hormiga caminará muy poco en un radio de un círculo con un área determinada. Para fortalecer la asociación se genera un conjunto de todos los posibles puntos de empate observados entre dos imágenes consecutivas. Los resultados que obtienen en base al seguimiento simultáneo de múltiples hormigas, para los 120 ejemplos, está en un promedio de 10.5 hormigas presentes. El promedio entre el humano y el sistema difieren entre 1.2 hormigas para cada ejemplo (alrededor del 11 %). Por lo tanto se tiene un 89 % en promedio de una detección de movimiento [3]. Por otro lado, en [4] se realiza el reconocimiento y modelado de maniobras de un conductor para un nivel táctico con énfasis especial en cómo el contexto afecta la funcionalidad del conductor. Ellos pueden predecir la próxima maniobra del conductor e infieren las intenciones de este, en base a algunas características esenciales como por ejemplo la información contextual, la cual se demuestra que es crítica para el reconocimiento exacto de algunas maniobras. El método que ocupan para tal reconocimiento está basado en aproximaciones Bayesianas para retroalimentar eventualmente el sistema perceptual, combinan las observaciones y los comportamientos a priori complejos, tales observaciones están en base a las características del auto y del conductor. La obtención de las características se realizan por medio de 4 cámaras y algunos sensores instalados en el auto, las cámaras tienen el objetivo de determinar las características contextuales. Con la ayuda de los sensores instalados en el auto, se obtienen las características del auto, como su posición por ejemplo. El procedimiento que realizaron para la detección de comportamientos es el siguiente:

1. Se conduce un auto alrededor de 1.5 horas en la ciudad de Boston, en el auto van dos personas, el conductor e instructor.

2. El instructor pide al conductor que se dirija a una dirección determinada.

3. El instructor etiqueta cada una de las maniobras del conductor.

4. El instructor le pide al conductor comente la próxima maniobra que realizará y enseguida realizarla.

Las características que se utilizaron para el experimento son: señales contextuales, posición del carril por donde el auto está pasando y la dirección de la vista del conductor.

El sistema que se desarrolla en este trabajo reconoce las siguiente maniobras que el conductor realiza:

1. Auto en movimiento.

2. Cambio de carríl a la derecha e izquierda.
3. Vuelta a la derecha e izquierda.
4. Inicio
5. Fin

La precisión que obtienen se puede observar en la tabla 2.1.

	CAR	CAR & LANE	CAR & GAZE
pass	100	100	100
turn left	0	33.3	66.7
change lane right	0	12.5	6.3
change lane left	29.4	17.6	23.5
start	100	66.7	83.3
stop	100	100	100

Tabla 2.1: Precisión del sistema de reconocimiento del comportamiento de un conductor.

La primera columna representa resultados obtenidos sólo teniendo en consideración las señales contextuales, en la segunda columna se tiene en consideración también el cambio de carril y la última columna se consideran todas las características incluyendo la dirección de la vista del conductor. La precisión aumenta cuando se toman más características, sin embargo en muchos casos hay características irrelevante que sólo causarían pérdida de posición en los resultados.

Nam T. Nguyen y algunos investigadores más en [5] realizan detección de comportamientos de personas en un ambiente complejo de trabajo basado en AHMM Abstract Hidden Markov Model. Se presenta este tipo de modelos AHMM ya que en el ombiente de trabajo se tienen muchas regiones de interés donde este tipo de modelos pueden realizar eficientemente la detección de comportamientos. En los AHMM una simple cadena de Markov en un Modelo Oculto de Markov HMM es reemplazado por una política jerárquica de Markov. En esta jerarquía de políticas, cada comportamiento puede ser representado como una política para cada nivel correspondiente de abstracción. En la detección se presenta cierta cantidad de ruido lo cual se maneja con un HMM y un filtro de Rao-Blackwelli de tal manera que obtenien la probabilidad de la política actual para los diferentes niveles de la jerarquía y enseguida utilizan el filtro Kalman para el reconocimiento de comportamientos del más bajo nivel integrado al AHMM. El ambiente donde se realiza la detección de comportamientos está compuesto por un corredor y un laboratorio de visión. En cada una de estas regiones se instalan cámaras de tal manera que sea posible observar el comportamiento de cada persona que ingresa a las regiones. Dentro del laboratorio

de visión se encuentra un servidor Linux, un servidor NT y una impresora, la cámara está colocada de tal manera que se puedan localizar sin problema alguno estas tres regiones de interés. Por otra parte dentro del pasillo se encuentra una región de lectura, la cámara está dispuesta de tal forma que se puede observar quién está en la región de lectura, así como también, quién entra por la derecha o izquierda del corredor y quién entra en el laboratorio de visión desde este punto (corredor). En los AHMM es necesario establecer jerarquías, por lo cual construyen una región jerárquica de tres niveles comenzando de la jerarquía más alta como se puede verificar a continuación:

1. Ambiente completo.
2. Corredor y al laboratorio de visión.
3. Solamente Corredor.

Una vez que se plantearon los tres niveles de jerarquía, proceden a plantear cada una de las políticas adecuadas para cada una de las regiones de los tres niveles, tal como se muestra a continuación:

1. Dentro de la Región del corredor.
2. Dentro de la Región de lectura (en corredor).
3. Dentro de la Región del Servidor Linux (en laboratorio de visión).
4. Dentro de la Rregión de la impresora (en laboratorio de visión).
5. Dentro de la Región del Laboratorio de Visión
6. Dentro de la Región del Servidor NT (en laboratorio de visión).
7. Entrando por la Izquierda de la Región del corredor.
8. Entrando por la Derecha de la Región del corredor.
9. Entrando al Laboratorio de Visión, desde el corredor.
10. Entrando al corredor, desde el laboratorio.
11. Movimiento dentro de la Región de lectura.
12. Movimiento dentro de la Región del Servidor Linux.
13. Movimiento dentro de la Región de la impresora.
14. Movimiento dentro de la Región del Servidor NT.

Obtienen gráficas acerca del comportamiento y probabilidades en base al tiempo t de las secuencias de imágenes para una determinada acción.

Adam Feldman y Tucker Balch en [1] presentan un sistema que puede automáticamente analizar movimientos de animales, y etiquetar, de acuerdo a un modelo, el comportamiento que una persona experta determina. Con un sólo conjunto de entrenamiento y etiquetado de los datos, el sistema automaticamente realiza el proceso reconocimiento de comportamientos etiquetando cada uno de las imágenes de una secuencia. Para realizar el experimento, graban la actividad de la colmena en un video digital, este video es procesado para obtener las posiciones (x,y) de cada una de las abejas y entonces las características numéricas tal como la velocidad y el cambio del etiquetado son extraidas. Las características son usadas para etiquetar cada uno de los movimientos que definen como:

1. Movimiento hacia la derecha.
2. Movimiento hacia la izquierda.
3. Movimiento en línea recta.
4. Movimiento oscilatorio.
5. Movimiento muy despacio con dirección no específica.
6. Sin movimiento.

La aproximación del reconocimiento de los movimientos anteriores está basado en una combinación del clasificador K-Nearest Neighbor (KNN) y en la técnica de hidden Markov model HMM. El sistema fue evaluado sobre cientos de trayectorias extraidas de abejas, apartir de la grabación del video con duración de 15 minutos observando la colmena entera. Los resultados que se obtienen se encuentran en la tabla 2.2.

N	Motion Accuracy	Behavior Accuracy
0	85.00 %	85.10 %
0.05	85.50 %	85.80 %
0.1	72.50 %	73.80 %
0.15	52.50 %	54.60 %
0.2	56.50 %	55.30 %
0.3	46.50 %	51.10 %

Tabla 2.2: Presición del sistema de reconocmiento de comportamientos de la Abeja de la Miel.

Notemos que en este último proyecto es muy similar a la problemática que se tiene, por tal razón, este proyecto de tesis se basará en gran parte en[1], sin dejar

pasar algunas ideas como la obtención del fondo en[3] y la obtención de características contextuales relevantes en[4] y por último el análisis de los modelos AHMM como una extensión de HMM que se plantea en[5].

2.3. Teoría de los Modelos Ocultos de Markov

En el mundo real existen diversos procesos que generalmente producen salidas observables que pueden ser caracterizadas en señales, estas pueden ser discretas como por ejemplo caracteres de un alfabeto finito. Pueden también ser continuas, por ejemplo: el habla, medidas de temperatura, música, etc.

Este tipo de señales que podemos encontrar en nuestra vida cotidiana pueden o no variar en el tiempo y pueden o no estar corruptas por muchos factores como el ruido o la distorsión por transmisión. El interés fundamental es poder caracterizar estas señales del mundo real en modelos de señales, debido a tres grandes ventajas que los modelos pueden proveernos [6]:

1. El modelado de las señales pueden proveer la base para la descripción teórica de un sistema para el procesamiento de estas. Es decir, si queremos realzar una señal corrupta por ruido y distorsión de transmisión, podemos usar el modelado de la señal para desarrollar un sistema el cual removerá óptimamente el ruido y disminuirá la distorsión de la transmisión.

2. El modelo nos deja aprender mucho acerca de la señal sin tener disponible la fuente, esto es muy importante cuando el costo en la obtención de las señales es muy alto, con un buen modelado podemos simular la señal de la fuente y así encontrar muchas vías de solución del problema.

3. Trabajan extremadamente bien en la práctica, permitiéndoos realizar importantes sistemas prácticos.

En las siguientes subsecciones se explicará la división de los Modelos, las cadenas de Markov y al final se extenderá la idea de tales cadenas hacia los Modelos Ocultos de Markov. Finalmente teniendo bien definido los Modelos Ocultos de Markov se enfocará esta tesis hacia tres problemas fundamentales de tales modelos:

1. Evaluación de la probabilidad de una secuencia de observaciones determinadas de un Modelo Oculto de Markov específico.

2. Determinar la mejor secuencia del modelo de estados.

3. Ajuste de los parámetros del modelo que explique lo mejor de la señal observada, es decir, maximizar la probabilidad de que la señal observada pertenezca a un modelo determinado.

2.3.1. Modelos

Los modelos puedes ser divididos en dos grandes clases: Modelos Determinísticos y Modelos Estadísticos. Los Modelos Determinísticos explotan el conocimiento de las propiedades específicas de la señal. Ejemplo de esto, es la onda del seno o una suma de exponenciales. En este tipo de modelos se obtiene lo deseado de manera determinística es decir libre de riesgo debido a que este tipo de modelos, a través de las propiedades bien conocidas, se pueden determinar los valores de sus parámetros, es decir, si conocemos la onda del seno podremos, sin riesgo alguno, conocer por ejemplo su amplitud y su frecuencia.

Por otro lado, los Modelos Estadísticos intentan caracterizar las propiedades estadísticas de una señal determinada, ejemplo de esto son los procesos Gaussianos, procesos de Poison, procesos de Markov, procesos ocultos de Markov, entre otros. Lo esencial de este tipo de modelos es que la señal puede ser bien caracterizada como un proceso paramétrico aleatorio y que los parámetros del proceso estocástico pueden ser determinados o estimados de manera exacta bien definida [6].

La presente tesis estará enfocada a un tipo de modelo estocástico: Modelo Oculto de Markov o por sus siglas en ingles HMM (Hidden Markov Model), para el resto de la tesis se tendrá presente sólo HMM para referirnos al Modelo Oculto de Markov.

2.3.2. Cadenas de Markov

Para entender los HMM es necesario entender, en un principio, las cadenas de Markov ya que a través de estas los modelos son definidos.

El matemático ruso Andrei Markov en el año de 1907 introduce un proceso probabilístico que tiene una propiedad en particular, *un evento sólo depende del evento anterior*. En la actualidad tal proceso es llamado cadena de Markov [7].

Una cadena de Markov es una serie de eventos en la cual la probabilidad de que ocurra un evento sólo depende del evento inmediato anterior. Formalmente se define como un proceso estocástico que cumple con la propiedad de Markov: *El evento o estado t+1 sólo depende del evento o estado t y no de los eventos o estados posteriores a t*, es decir:

$$P[q_t = S_j | q_{t-1} = S_i, q_{t-2} = S_k, ...] = P[q_t = S_j | q_{t-1} = S_i] \tag{2.3.1}$$

Donde t representa el tiempo 1,2,3,...,N y cada q_t es el estado actual en un tiempo t. Existen varios tipos de cadenas de Marcov, en este caso en particular veremos las

cadenas de Markov discretas de primer orden, debido a que con estas podemos formar los HMM más utilizados en algunos de los trabajos revisados [1]. Llamados de primer orden ya que este tipo de cadenas tienen "memoria uno". A continuación se explicará este término con un ejemplo. Un pronóstico de la probabilidad de ocurrencia de cierto fenómeno, el clima por ejemplo, calculando las probabilidades condicionales obtenidas por medio de las cadenas de Markov para uno y sólo un día anterior. Es decir, primer orden significa que el pronóstico del clima del día de hoy sólo depende del clima del día de ayer (propiedad de Markov).

Las cadenas de Markov cuentan con una matriz de transición de estados para cada posición a_{ij} y cumple con la propiedad de Markov, es decir:

$$a_{ij} = P[q_t = S_j \mid q_{t-1} = S_i\ ],\ 1 \leq i,j \leq N \tag{2.3.2}$$

Todos los coeficientes a_{ij} tienen la siguiente propiedad

$$a_{ij} \geq 0 \tag{2.3.3}$$

$$\sum_{j=1}^{N} a_{ij} = 1 \tag{2.3.4}$$

Se puede notar que las reglas anteriores obedecen a reglas estocásticas estándares, debido a que los resultados del proceso son probabilidades que cambian en el tiempo y por lo cual su comportamiento es dinámico.

2.3.3. Modelos Ocultos de Markov

Los HMM fueron inicialmente estudiados e introducidos por L. E. Baum en los años 70's, Baum propone este modelo como un método estadístico de estimación de las funciones probabilísticas de una cadena de Markov [8]. Los HMM han llegado a ser muy populares por dos grandes razones. La primera razón es que los modelos son muy ricos en su estructura matemática por lo cual es posible realizar bases teóricas para usarlas en diferentes aplicaciones. La segunda razón es que si los modelos son aplicados de manera apropiada, estos trabajan muy bien en muchas aplicaciones importantes. Un ejemplo de esto es el reconocimiento automático del habla que se plantea en [6].

Se explicará el HMM con un ejemplo y enseguida se procederá a formalmente establecer un HMM. Consideremos un simple modelo de Markov de tres estados representando el clima como se encuentra en la parte de abajo:

1. Estado 1: Lluvioso
2. Estado 2: Nublado
3. Estado 3: Soleado

Se considera que el clima para cada día t es caracterizado por uno y sólo uno de los estados de arriba y la matriz A de probabilidades de transición de estados es:

$$A = \{ a_{ij} \} = \left[\begin{array}{c|ccc} & S_1 & S_2 & S_3 \\ \hline S_1 & 0.4 & 0.3 & 0.3 \\ S_2 & 0.2 & 0.6 & 0.2 \\ S_3 & 0.1 & 0.1 & 0.8 \end{array}\right]$$

Dado que el clima en el día 1 ($t = 1$) es soleado (estado 3), se podría plantear la siguiente pregunta: ¿cuál es la probabilidad, de acuerdo al modelo anterior, de que el clima en los próximos 7 días fuera: $S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3$? Sea la observación O = $\{S_3, S_3, S_1, S_1, S_3, S_2, S_3\}$, ahora lo que se desea determinar es presisamente la probabilidad de O dependiento del modelo. Por lo cual se puede expresar lo anterior de la siguiente manera:

$$\begin{aligned} P(O|Modelo) &= P[S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3|Modelo] \\ &= P[S_3] * P[S_3|S_3] * P[S_3|S_3] * \\ &\quad P[S_1|S_3] * P[S_1|S_1] * P[S_3|S_1] * \\ &\quad P[S_2|S_3] * P[S_3|S_2] \\ &= \pi_3 * (0.8) * (0.8) * (0.1) * (0.4) * (0.3) * (0.1) * (0.2) \\ &= 1.536 * 10^{-}4 \end{aligned} \tag{2.3.5}$$

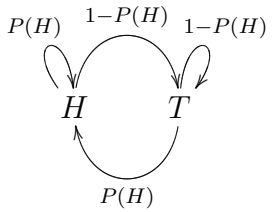
donde $\pi_i = P[q_1 = S_i]$, $1 \leq$ i $\leq$ N los cuales denotan las probabilidades de los estados iniciales.

Notese que las observaciones son bien definidas en el ejemplo anterior, ahora se supondrá que las observaciones son funciones de probablidad del estado, es decir el resultado del modelo es un proceso doblemente estocástico en el cual uno de esos procesos estocásticos no es observable (este es oculto). Este proceso puede ser observable gracias al otro proceso estocástico que produce las secuencias de observaciones. El ejemplo siguiente mostrará el significado del enunciado anterior.
Supóngase el siguiente escenario. Se encuentra una persona dentro de un cuarto con una barrera, por ejemplo una cortina, por la cual esta persona no puede ver lo que sucede en el otro extremo. Por el otro extremo de la cortina se tiene otra persona la

cual está realizando un experimento con tiros de monedas (volados). La persona que realiza el experimento no le dice a la otra persona con exactitud cómo está realizando el experimento, sólo le dirá el resultado obtenido en cada uno de los tiros ya sea Cara o Cruz, se entenderá por Cara con la letra H y Cruz con la letra T (por sus siglas en inglés Cara: Head y Cruz: Tails). Una vez realizado el experimento se forma una serie de tiros *ocultos* y una secuencia de observaciones que consisten en una serie de H o T. Generalmente una secuencia de observaciones puede verse de la siguiente manera:

$$\mathbf{O} = O_1 O_2 O_3 O_4 O_5 ... O_T \tag{2.3.6}$$
$$= TTHHHTHHT...T \tag{2.3.7}$$

El problema principal es determinar un HMM el cual pueda explicar la secuencia de observación anterior. Lo primero que debemos determinar es saber cuales serán los estados correspondientes en el modelo y entonces determinar cuales serán los estados que formen el modelo. Una posible elección podría ser suponer un simple tiro de la modena (un sólo volado). En este caso se podría modelar la situación con un modelo de dos estados correspondientes a H o T como se muestra en la figura 2.1.



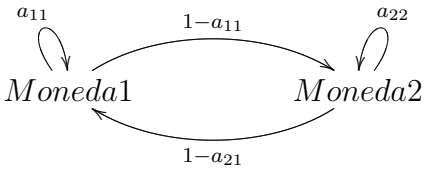
O = HHTTTHHHTHHTHTT

Figura 2.1: Primer posible modelo

En este caso el modelo de Markov es observable, lo único que falta por determinar en este caso es la probabilidad de decidir H o T. Un HMM equivalente a la figura 2.1 podría ser visto como una forma degenerada de un HMM en el cual el símbolo de probabilidad de observación en cada estado es 1.0 para un símbolo en particular y 0.0 para todos los demás símbolos [6]. En otras palabras, cada estado en un modelo de Markov reconoce o genera exactamente un símbolo. Si se elimina de manera eficiente la distribución de probabilidad del símbolo de observación se remueve el estado oculto supuesto en un HMM. Un modelo de Markov difiere de un HMM degenerado en que ellos incorporan estadísticos adicionales en las ligas de los modelos y en sus nodos, los

cuales pueden ser usados para la construcción, modificación y explotación de modelos dinámicos [9].

Un segundo ejemplo para explicar los HMM se presenta en la figura 2.2 donde se tienen ahora 2 monedas y en las cuales se debe de elegir una de ellas, basado en un cierto evento probabilístico. Esta podría ser otra forma de explicar la secuencia observada de tiros de la moneda presentes en la figura. Así también como se muestra en la figura 2.3 de manera análoga a la figura 2.2.

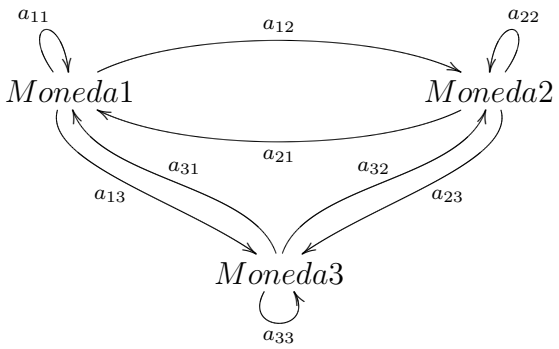


$$
\begin{aligned}
Observación : O &= HHHTTHTHTHTHHHTTH \\
Modena : S &= 21212121222122221 \\
P(H) &= P_{moneda1} \\
P(H) &= P_{moneda2} \\
P(T) &= 1 - P_{moneda1} \\
P(T) &= 1 - P_{moneda2}
\end{aligned}
$$

Figura 2.2: Segundo posible modelo

Se observa que se puden tener muchos modelos diferentes para poder explicar las observaciones obtenidas en el experimento, con esto surge una pregunta muy natural: cuál de los modelos presentes en las figuras anteriores modelan de la mejor manera la observación obtenida en el experimento.

Debe de estar claro que en la figura 2.1 se tiene un sólo parámetro desconocido, en la figura 2.2 se tienen 4 parámetros desconocidos y en la figura 2.3 se tienen 9 parámetros desconocidos, así parecería que el HMM más grande sería capaz de modelar el HMM más pequeño. Aunque esto teóricamente es cierto, no siempre es lo más conveniente[6]. Los ejemplos anteriores de HMM, los cuales modelan un determinado fenómeno, ilustran algunos de los elementos necesarios para, formalmente, poder definir un HMM. Se observa que es necesario definir de alguna forma las probabilidades de la secuencia de observación, así como las probabilidades de transición entre los estados ya que con esta información podemos resolver algunos problemas como, dada una observación, la probabilidad de que esta pertenezca a un



$$
\begin{aligned}
Observación : O &= HHTTTHTHTHTHTTHHT \\
Modena : S &= 21122333232313223 \\
P(H) &= P_{moneda1}, P_{moneda2}, P_{moneda3} \\
P(T) &= 1 - P_{moneda1}, 1 - P_{moneda2}, 1 - P_{moneda3}
\end{aligned}
$$

Figura 2.3: Tercer posible modelo

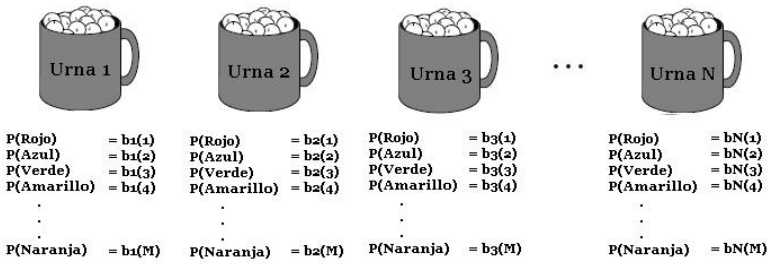
determinado modelo. Enseguida se plantea un último ejemplo clásico y más complejo y después se definen los elementos de un HMM.

Se considera un sistema de urnas y balones como se muestra en la figura 2.4.

Se supondrá que hay N urnas en un cuarto. En cada una de las urnas hay un número M determinado de balones de distinto color. Las observaciones se obtienen mediante un proceso aletorio de la siguiente manera: una persona elige una urna inicial de la cual elige un balón, el color de este balón seleccionado será anotado dentro del conjunto de observaciones, después el balón es reemplazado en la urna de donde se ha seleccionado. Una nueva urna es seleccionada y se repite el procedimiento nuevamente. Tal proceso genera una secuencia de observaciones finita de colores la cual es necesaria modelar como una salida observable de un HMM. El modelo más simple del proceso anterior es uno en el cual cada estado corresponde a cada una de las urnas y se define la probabilidad del color para cada estado como se muestra en la figura 2.4. En base a los ejemplos anteriores se procederá a mencionar los elementos de un HMM.

2.3.4. Elementos de un HMM

Un HMM se caracteriza por lo siguiente:



O = {Verde, Verde, Azul, Amarillo, Rojo, Rojo,..., Azul}

Figura 2.4: Un modelo de urnas y balones de N estados el cual ilústra un caso general de un HMM discreto.

1. N, el número de estados en el modelo. Estos estados son ocultos, sin embargo en muchas aplicaciones prácticas hay algún significado físico anexado a los estados o conjunto de estados del modelo. Por la razón anterior, en el ejemplo de las monedas y el ejemplo de las urnas y balones, se tiene que los estados son precisamente las monedas y las urnas respectivamente. Generalmente los estados están interconectados de tal manera que cualquier estado puede ser alcanzable desde cualquier otro estado (ejemplo de esto es un modelo ergódico[1]). Se denotará desde este momento los estados individualmente como $S = \{S_1, S_2, \ldots, S_N\}$ y el estado en un tiempo t como q_t

2. M, el número de distintos símbolos de observación por cada estado, es decir, el tamaño discreto del alfabeto. Los símbolos de observación corresponden a la salida física del sistema que está siendo modelado. En el ejemplo de las monedas, los símbolos de la observación son Cara o Cruz; por parte del ejemplo de las urnas y balones los símbolos de la observación son los colores de los balones que se seleccionaban de cada una de las urnas. Se denotará los símbolos individuales como $V = \{v_1, v_2, \ldots, v_M\}$.

3. La distribución de probabilidad de transición de estados $A = \{a_{ij}\}$ donde

$$a_{ij} = P[q_{t+1} = S_j | q_t = S_i], 1 \leq i, j \leq N. \tag{2.3.8}$$

4. La distribución de probabilidad del símbolo de observación en el estado j, $B = \{b_i(k)\}$ donde

[1]Ergódico viene de los vocablos griegos *ergon* y *hodos* que significan trabajo y camino, modelo ergónico significa entonces que se requiere un esfuerzo no trivial para pasar por cada uno de los estados que tiene el modelo a través del camino que une a dichos estados

$$b_i(k) = P[v_k \textit{ en un tiempo } \text{t } |q_t = S_j], 1 \leq j \leq N, 1 \leq k \leq M. \tag{2.3.9}$$

5. El estado inicial de distribución $\pi = \{\pi_i\}$ donde

$$\pi_i = P[q_1 = S_i] 1 \leq i \leq N. \tag{2.3.10}$$

Dada una secuencia de observación

$$O = O_1, O_2, ..., O_T. \tag{2.3.11}$$

donde cada observación O_t es un símbolo de V, y T es el número de observaciones en la secuencia, se puede construir un HMM para generar la secuencia de observaciones si es que proporcionamos los valores apropiados de N, M, A, B, y π. El proceso de saber cómo una secuencia de observaciones dada fue generada por un HMM apropiado es de la siguiente manera:

1. Elegir un estado inicial $q_1 = S_i$ de la distribución del estado inicial π.
2. Sea $t = 1$.
3. Elegir $O_t = v_k$ de la distribución de probabilidad de los símbolos en un estado S_i, es decir, $b_i(k)$.
4. Transitar a un nuevo estado $q_{t+1} = S_j$ de la distribución de probabilidad de transición de estados para el estado S_i, es decir, a_{ij}.
5. Sea $t = t+1$; regresar al paso 3 si $t < T$; en otro caso el procedimiento termina.

De una manera mas simple y compacta un HMM puede ser representado de la siguiente manera:

$$\lambda = (A, B, \pi) \tag{2.3.12}$$

2.3.5. Problemas básicos de los HMMs

Los problemas fundamentales que tienen que ser resueltos en un HMM son los siguientes:

1. Dada la secuencia de observación $O = O_1, O_2, ..., O_T$, y el modelo λ = *(A, B, π)*, ¿Cómo computar eficientemente *P(O|λ)*, la probabilidad de la secuencia de observación dado el modelo?.

2. Dada la secuencia de observación $O = O_1, O_2, ..., O_T$, y el modelo λ, ¿Cómo elegir una secuencia de estados correspondientes a $Q = q_1 q_2 ... q_T$ el cual sea óptimo para un cierto sentido significativo?, es decir, que explique lo mejor posible las observaciones.

3. ¿Cómo ajustar los parámetros del modelo λ = *(A, B,* Π*)* para maximizar *P(O|λ)*?

Solución del problema número 1

En este problema se desea calcular la probabilidad de la secuencia de observación, $O = O_1, O_2, ..., O_T$, dado el modelo λ, es decir *P(O|λ)*. El camino más directo para realizar este cálculo, es enumerar cada posible secuencia de estados de longitud T (número de observaciones). Se considera una secuencia de estados fija

$$Q = q_1, q_2, ..., q_T \tag{2.3.13}$$

donde q_1 es el estado inicial. La probabilidad de la secuencia de observación O, dada la secuencia de estados anterior es

$$P(O|Q, \lambda) = \Pi_{t=1}^{T} P(O_t|Q_t, \lambda) \tag{2.3.14}$$

donde se asume que se tiene independencia estadística de las observaciones. Así tenemos que

$$P(O|Q, \lambda) = b_{q_1}(O_1) * b_{q_2}(O_2) * ... * b_{q_T}(O_T). \tag{2.3.15}$$

la probabilidad de tal secuencia de estados Q puede ser escrita como

$$P(Q|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} ... a_{q_{T-1}\ q_T} \tag{2.3.16}$$

La probabilidad de que O y Q ocurran simultaneamente, es simplemente el producto de las dos ecuaciones de arriba

$$P(O, Q|\lambda) = P(O, Q|\lambda) P(Q|\lambda) \tag{2.3.17}$$

La probabilidad de O dado el modelo, se obtiene por la sumatoria de las probabilidades simultaneas de O y Q sobre todas las posibles secuencias q obteniendo

$$\begin{aligned} P(O|\lambda) &= \sum_{allQ} P(O, Q|\lambda) P(Q|\lambda) \\ &= \sum_{q_1 q_2 ... q_T} \pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) ... a_{q_{T-1} q_T} b_{q_T}(O_T) \end{aligned} \tag{2.3.18}$$

Las ecuaciones anteriores se pueden expresar de la siguiente manera. Inicialmente (en un tiempo *t=1*) un estado q_1 con probabilidad π_{q_1} genera el símbolo O_1 con probabilidad $b_{q_1}(O_1)$. El tiempo avanza de un tiempo t a un tiempo *t+1* es decir *t=2* y se realiza una transición de estado de q_1 a q_2 con una probabilidad de $a_{q_1 q_2}$ el cual genera un símbolo O_2 con probabilidad $b_{q_2}(O_2)$. Este proceso continua de esta manera hasta llagar a un tiempo T de un estado q_{T-1} a un estado q_T con una probabilidad $a_{q_{T-1} q_T}$ y genera un símbolo O_T con probabilidad $b_{q_T}(O_T)$. Se nota claramente que el orden de la ecuación de arriba, la cual resuelve $P(O|\lambda)$, está alrededor de $2T * N^T$ cálculos, lo cual si tuvieramos N = 5 estados y T = 100 observaciones necesitaríamos resolver $2 * 100 * 5^{100} \simeq 10^{72}$ cálculos. Esta claro que es necesario un procedimiento para resolver este tipo de cálculo, afortunadamente existe tal procedimiento llamado *forward-backward* o procedimiento de avance y retroceso.

El procedimiento de avance y retroceso visto en [6] es el siguiente. Se considera la variable *avance* $\alpha_t(i)$ y la variable *retroceso* $\beta_t(i)$ definidas de la siguiente manera.

$$\alpha_t(i) = P(O_1 O_2 ... O_t, q_t = S_i|\lambda) \tag{2.3.19}$$

es decir, la probabilidad de la secuencia de observación parcial, $O_1 O_2 ... O_t$ en un tiempo t y el estado S_i en un timpo t dado el modelo λ. Es posible resolver $\alpha_t(i)$ inductivamente de la siguiente manera:

$$
\begin{aligned}
&1)Inicio: \\
&\alpha_1(i) = \pi_i b_i(O_i), 1 \leq i \leq N. && (2.3.20)\\
&2)Inducción: \\
&\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}) && (2.3.21)\\
&\qquad 1 \leq t \leq T-1, 1 \leq j \leq N \\
&3)Termina: \\
&P(0|\lambda) = \sum_{i=1}^{N} \alpha_T(i). && (2.3.22)
\end{aligned}
$$

y de manera análoga para $\beta_t(i)$

$$\beta_t(i) = P(O_{t+1}O_{t+2}...O_T|q_t = S_i, \lambda) \quad (2.3.23)$$

es decir, la probabilidad de la secuencia de observación parcial, desde *t+1* hasta el final, dado un estado S_i, en un tiempo t y el modelo λ. Tambíen es posible resolver $\beta_t(i)$ inductivamente de la siguiente manera:

$$
\begin{aligned}
&1)Inicio: \\
&\beta_T(i) = 1, 1 \leq i \leq N. && (2.3.24)\\
&2)Inducción: \\
&\beta_T(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j), && (2.3.25)\\
&\qquad t = T-1, T-2, ..., 1.1 \leq i \leq N. \\
&3)Termina: \\
&P(0|\lambda) = \sum_{i=1}^{N} \alpha_T(i). && (2.3.26)
\end{aligned}
$$

El orden de la $P(O|\lambda)$, calculada de esta manera, está en N^2T [6], por lo cual para el ejemplo de N = 5 estados y T = 100 observaciones necesitaríamos resolver $5^2 * 100 = 2500$ cálculos.

Los cálculos de las variables *avance* y *retroceso* son necesarias para resolver el problema número dos planteado con anterioridad, tal como está presente en las siguientes líneas.

Solución del problema número 2

El problema de determinar la secuencia de estados $Q = q_1 q_2 ... q_T$ tal que sea la secuencia de estados más probables dado una secuencia de observación O y el modelo λ es resuelto de la siguiente manera en [6].

Sea la variable

$$\gamma_t(i) = P(q_t = S_i|O, \lambda) \tag{2.3.27}$$

es decir, la probabilidad de que sea el estado S_i en un tiempo t, dada la observación O y el modelo λ, la ecuación anterior puede ser expresada en términos de las variables de avance y retroceso.

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(O|\lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i-1}^{N} \alpha_t(i)\beta_t(i)} \tag{2.3.28}$$

es decir, puesto que $\gamma_t(i)$ considera la secuencia de observación parcial $O_1, O_2, ..., O_t$ y un estado S_i en un tiempo t, mientras que $\beta_t(i)$ considera el resto de la secuencia de observación $O_{t+1}, O_{t+2}, ..., O_T$ y un estado S_j en un tiempo t. La normalización del factor $P(O|\lambda) = \sum_{i-1}^{N} \alpha_t(i)\beta_t(i)$ hace que $\gamma_t(i)$ una medida de probabilidad tal que

$$\gamma_t(i) = 1 \tag{2.3.29}$$

Utilizando $\gamma_t(i) = 1$, se puede resolver el estado individualmente más probable q_t en un tiempo t como

$$q_t = argmax[\gamma_t(i)] \tag{2.3.30}$$

Aunque maximiza el número de estados correctos (eligiendo el estado más probable en cada tiempo t) podría haber problemas con la secuencia de estados resultante. Por ejemplo, cuando el HMM tiene transiciones de estados con probabilidad cero es decir

$$a_{ij} = 0 \ para \ algunos \ i,j \tag{2.3.31}$$

la secuencia óptima puede no ser una secuencia válida del estado, debido a que $q_t = argmax[\gamma_t(i)]$ determina simplemente el estado más probable para cada instancia sin considerar la probablidad de conjuntos de secuencias de estados. Para resolver este problema se tiene un algoritmo capaz de maximizar $P(Q|O,\lambda)$ lo cual es equivalente a maximizar $P(Q,O|\lambda)$ planteado en [6] y llamado algortimo de Viterbi.

El algoritmo de Viterbi encuentra la única mejor secuencia de estados $Q = q_1\ q_2\ ...\ q_T$ para dada la secuencia de observación $O = O_1\ O_2\ ...\ O_T$. Es necesario definir la cantidad

$$\delta_t(i) = max_{q_1,q_2,...,q_{t-1}} P[q_1, q_2, ..., q_t = i, O_1, O_2, ..., O_t|\lambda] \tag{2.3.32}$$

es decir, $\delta_t(i)$ tiene la más alta probabilidad a lo largo de una trayectoria en un tiempo t, la cual considera las primetas t observaciones y las últimas en un estado S_i. De esta manera y con ayuda de la recursión planteada en [6] es posible obtener la secuencia de estados más probable dado un modelo λ y una secuencia de observaciones O.

Solución del problema número 3

El tercer problema es determinar un método para ajustar los parámetros del modelo (A, B, π) para maximizar la probabilidad de la secuencia de observación dado el modelo. En esta sección se presenta un procedimiento iterativo basado en el clásico trabajo de Baum y sus colegas para reestimar los parámetros del modelo.

Este procedimiento se encentra en [6] y se explica a continuación.

Primeramente se define $\xi_t(i,j)$ la probabilidad de comenzar en un estado S_i en un tiempo t, y estado S_j en un tiempo $t+1$ dado el modelo y la secuencia de obsevación, es decir

$$\xi_t(i,j) = P(q_t = S_i,\ q_{t+1} = S_j|O,\lambda). \tag{2.3.33}$$

Se puede escribir $\xi_t(i,j)$ en función de las variables de avance y retroceso de la siguiente forma

$$\begin{aligned} \xi_t(i,j) &= \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{P(O|\lambda)} \\ &= \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)} \end{aligned} \tag{2.3.34}$$

Es posible relacionar las ecuaciones $\gamma_t(i)$ y $\xi_t(i,j)$ de la siguiente manera

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j) \tag{2.3.35}$$

si se suma $\gamma_t(i)$ en un tiempo t se obtiene una cantidad que puede ser representada como el número esperado de veces que el estado S_i es visitado o equivalente al número de transiciones hechos desde el estado S_i. De forma similar se tiene que la suma de $\xi_t(i,j)$ sobre un tiempo t puede ser interpretado como el número esperado de transiciones que del estado S_i al estado S_j es decir

$$\sum_{t=1}^{T-1} \gamma_t(i) = \textit{Número esperado de transiciones de } S_i \tag{2.3.36}$$

$$\sum_{t=1}^{T-1} \xi_t(i,j) = \textit{Número esperado de transiciones de } S_i \textit{ a } S_j \tag{2.3.37}$$

Utilizando las ecuaciones de arriba y los conceptos de conteo de ocurrencias de eventos es posible obtener un método de reestimación de los parámetros de un HMM.

Una manera razonable de encontrar las fórmulas de reestimación de los parámetros de A, B y π basados en las afirmaciones anteriores son

$$\begin{aligned} \bar{\pi}_i &= \textit{Frecuencia esperada (número de veces) en el estado } S_i \textit{ en un tiempo (t=1)} \\ &= \gamma_t(i) \end{aligned} \tag{2.3.38}$$

$$\begin{aligned} \bar{a_{ij}} &= \frac{\textit{Número esperado de transiciones de } S_i \textit{ a } S_j}{\textit{Número esperado de transiciones de } S_i} \\ &= \frac{\sum_{t=1}^{T-1} \gamma_t(i)}{\sum_{t=1}^{T-1} \xi_t(i,j)} \end{aligned} \tag{2.3.39}$$

$$\begin{aligned} \bar{b_j(k)} &= \frac{\textit{Número esperado de veces en el estado } j \textit{ y observación del símbolo } v_k}{\textit{Número esperado de veces en el estado } j} \\ &= \frac{\sum_{t=1,O_t=v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \end{aligned} \tag{2.3.40}$$

Con las fórmulas anteriores se tiene por un lado un modelo $\bar{\lambda}$ y por otro lado un modelo λ. Con la afirmación anterior se tiene que es posible realizar la comparación de que la $\mathrm{P}(\mathrm{O}|\bar{\lambda}) \geq \mathrm{P}(\mathrm{O}|\lambda)$ con el objetivo de reestimar los nuevos parámetros del modelo de forma iterativa como se muestra en [6], en el cual también se llega a las fórmulas necesarias para la reestimación como se muestra a continuación

$$\pi_i = \frac{\pi_i \frac{\partial P}{\partial \pi_i}}{\sum_{k=1}^{N} \pi_k \frac{\partial P}{\partial \pi_k}} \tag{2.3.41}$$

$$a_{ij} = \frac{a_{ij} \frac{\partial P}{\partial a_{ij}}}{\sum_{k=1}^{N} a_{ik} \frac{\partial P}{\partial a_{ik}}} \tag{2.3.42}$$

$$b_j(k) = \frac{b_j(k) \frac{\partial P}{\partial b_j(k)}}{\sum_{l=1}^{M} b_j(l) \frac{\partial P}{\partial b_j(l)}} \tag{2.3.43}$$

donde $\frac{\partial P}{\partial x}$ es un multiplicador de Lagrage el cual ayuda a maximizar la función de probabilidad P para obtener, de esta manera y de forma iterativa $\mathrm{P}(\mathrm{O}|\bar{\lambda}) \geq \mathrm{P}(\mathrm{O}|\lambda)$ [6].

2.3.6. Algoritmo de clasificación K-Nearest Neighbor(KNN)

Hasta este momento se ha resuelto de alguna manera los tres problemas fundamentales de un HMM. De manera general se ha resuelto, que dado un modelo λ, la probabilidad de la secuencia de observación y todo lo que esto implica.

Lo que no se ha planteado aún es cómo se obtendrá la secuencia de observación. En [1] se explica que las observaciones son obtenidas con la salida del algoritmo de clasificación KNN y debido a que la problemática que es resuelta en ese artículo es muy similar a la que se tiene en este proyecto de tesis, se obtendrá la secuencia de

observación de la misma manera, por lo cual se explica este algoritmo de clasificación KNN a continuación.

La idea básica de KNN es que un nuevo caso se va a clasificar en la clase más frecuente a la que pertenecen sus K vecinos más cercanos, por lo cual se fundamenta en una idea muy intuitiva y simple, lo que unido a su fácil implementación hace que sea un algoritmo de clasificación muy extendido.

Las entradas del algoritmo de clasificación KNN son el conjunto de entrenamiento y el valor de K. El conjunto de entrenamiento está formado por una tupla, el primer elemento es un conjunto con todas las características que se tomarán en cuenta para la clasificación, mientras que el segundo elemento es la clase a la que pertenecen dichas características. En la tabla de abajo se muestra de manera gráfica de un posible conjunto de entrenamiento.

$\{x_1, x_2, ..., x_k\}$	c_1
$\{x_1, x_2, ..., x_k\}$	c_2
$\{x_1, x_2, ..., x_k\}$	c_3
...	...
$\{x_1, x_2, ..., x_k\}$	c_{n-1}
$\{x_1, x_2, ..., x_k\}$	c_n

Tabla 2.3: Representación de conjunto de entrenamiento para el algoritmo de clasificación KNN.

n: número de instancias.

$c_i \epsilon \{C_1, C_2, ..., C_m\}$ y C_i es una clase determinada y m: número de clases.

$x_i \epsilon \{X_1, X_2, ..., X_t\}$ y X_i es una característica numérica determinada y t: número características.

Por otro lado el valor de K es el número de K vecinos más cercanos a ser clasificada la nueva instancia dependiendo de un criterio de distancia, es decir para cada instancia en el conjunto de entrenamiento se obtiene la distancia al nuevo punto a ser clasificado. Se obtienen los K vecinos más cercanos hacia el nuevo punto y entonces se decide a que clase pertenece tal punto como lo muestra la figura 4.5. En la figura se puede notar que existen 2 clases diferentes ***cruz*** y ***circulo*** y donde llega una nueva instancia ***punto***. Esta nueva instancia se ha clasificado con K = 3 a la clase ***circulo***, se puede notar que si K = 1 se hubiera clasificado esta instancia a la clase ***cruz***.

La manera más común del cálculo de distancia es utilizando la distancia euclidiana

$$d(p_1, p_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{2.3.44}$$

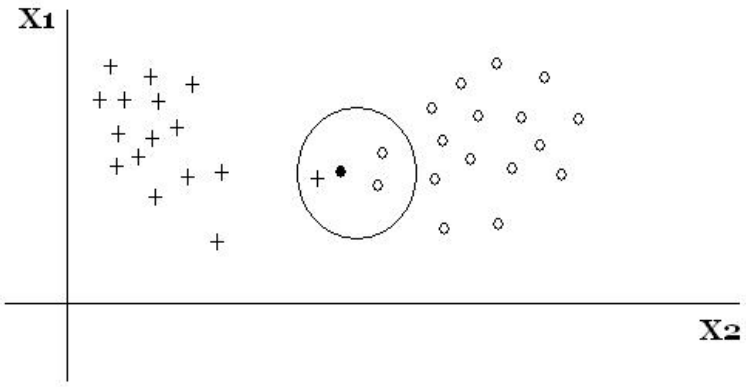


Figura 2.5: Conjunto de entrenamiento para KNN

Capítulo 3

Propuesta de Solución

En este capítulo se propone una solución de reconocimiento de tres de los comportamientos de la mosca del mediterráneo basado en HMM y KNN. Los comportamientos son definidos en la sección de clasificación manual ya que hasta esa parte son ocupados para separar el conjunto de entrenamiento y el conjunto de clasificación automática. El procedimiento realizado está dividido en cuatro bloques: Preparación de archivos, Análisis de imágenes, Clasificación manual y Clasificación automática. En el diagrama de bloques de la figura 3.1 muestra cómo cada uno de estos bloques está conectado para finalmente obtener una solución al problema planteado. La descripción de cada bloque está en secciones con los mismos nombres dentro de este capítulo.

En la primera sección se describe la estructura de los diferentes archivos ocupados para almacenar los resultados obtenidos de cada proceso en la clasificación manual y clasificación automática. También se describe la estructura de cada imagen y la forma de generarla ocupando los archivos de datos de entrada.

La sección de Análisis de imágenes está compuesta por tres subsecciones Visualizar imágenes, Determinar parámetros de segmentación y finalmente la Segmentación. Estas subsecciones describen la solución al problema de obtener una sola región en el método de segmentación, la cual debe pertenecer a la región de interés.

Para la clasificación manual se describe el método en cómo cada imagen se clasifica utilizando los archivos necesarios para este procedimiento. En esta sección también se definen los comportamientos a ser reconocidos de manera automática por el sistema.

Los algoritmos de KNN y Viterbi, así como el modelo oculto de Markov, se describen en la última sección de clasificación automática. Se muestra cómo estos algoritmos son utilizados para la obtención de resultados finales, así como el algoritmo de Viterbi suaviza los resultados obtenidos con el primer algoritmo de clasificación KNN de tal manera que se incrementa la precisión del sistema.

A continuación se describe a fondo cada una de las secciones para el desarrollo de

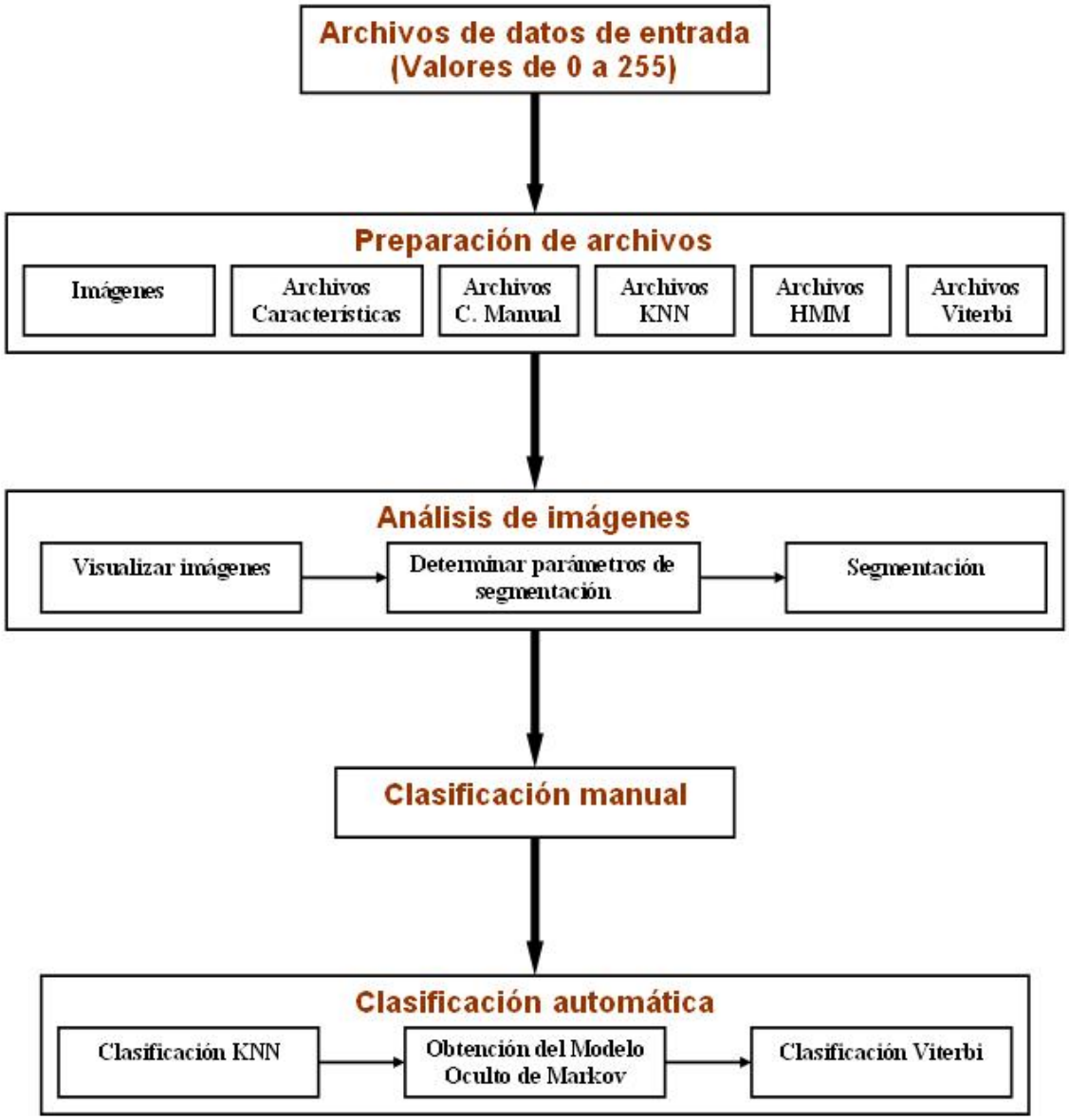


Figura 3.1: Diagrama de bloques.

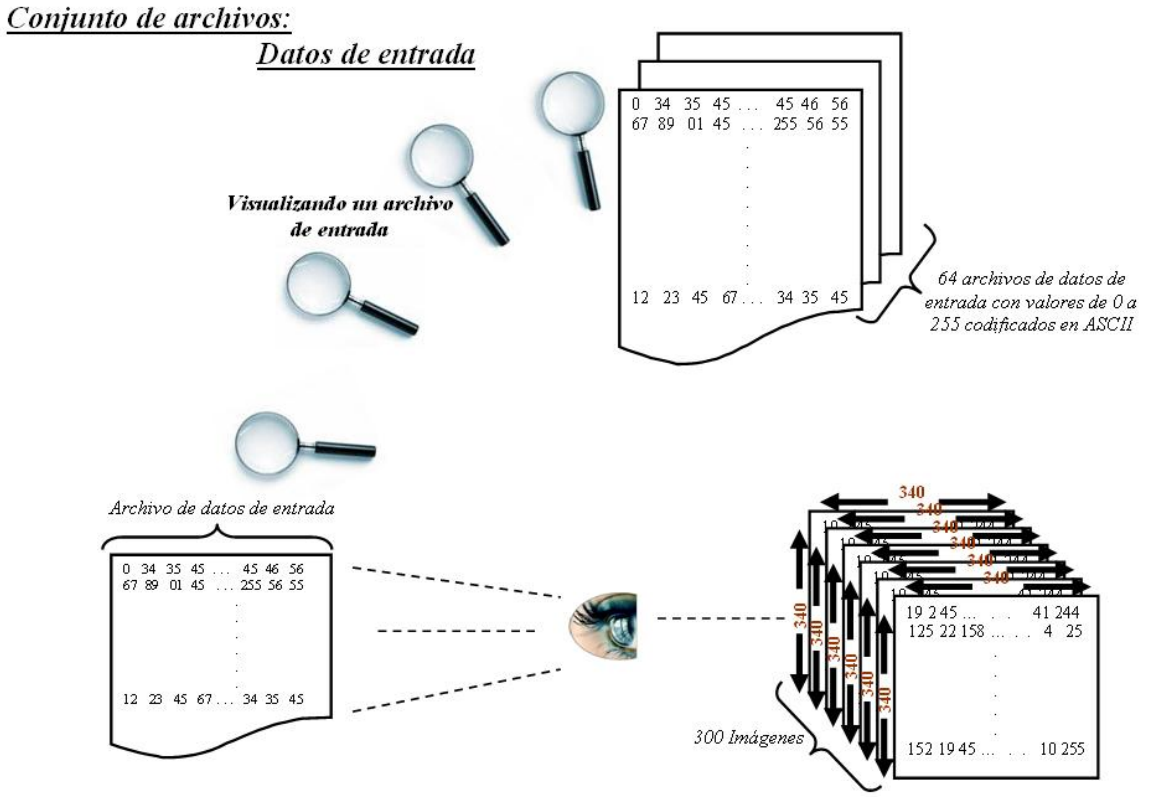


Figura 3.2: Imágenes organizadas en archivos de datos de entrada.

este proyecto.

3.1. Preparación de archivos

Es necesario determinar la forma en cómo los resultados se almacenan, por esta razón se describen las estructuras de archivos que ayudan a obtener los resultados en cada una de las secciones de análisis de imagen, clasificación manual y clasificación automática. Los datos de entrada para este proyecto son proveídos por el laboratorio de visión del INAOE. La figura 3.2 muestra la estructura de cada uno de estos archivos de datos. A continuación, se describen las estructuras de datos creadas para cada módulo de este proyecto.

1. Archivos de datos de entrada (Valores de 0 a 255). Es un conjunto de 64 archivos con extensión .dat, cada uno contiene 300 imágenes y cada imagen tiene una resolución de 340 x 340 pixeles. Cada pixel está en escala de grises almacenados

en un byte con valores de 0 a 255 codificado en formato ASCII, lo cual está representado en la figura 3.2. Este conjunto es proveído por el laboratorio de visión del INAOE.

2. Imágenes. Las imágenes se generan utilizando un algoritmo simple de lectura de datos. El algoritmo está formado por 3 ciclos anidados, uno para leer los 64 archivos, el siguiente para la lectura de las 300 imágenes para cada archivo y el final para leer los 340 x 340 píxeles por cada imagen. El algoritmo 1 crea un conjunto de 19200 imágenes.

Algorithm 1: Generar Imágenes.

Data: $C = \{A_1[\,], A_2[\,], ..., A_{64}[\,]\}$ // conjunto de archivos de datos.
Result: $C' = \{I_1, I_2, ...I_{19200}\}$ // conjunto de imágenes.
begin
  for $A \in C$ **do**
    for $i = 1; i \leq 300; i++$ **do**
      for $j = 1; j \leq 340 * 340; j++$ **do**
        $a[j] \longleftarrow A[(i-1) * 300 + j];$
      $I \longleftarrow GeneraImagen(a[j]);$
      $C' \longleftarrow I;$
end

3. Archivos Características. Este conjunto de archivos está formado por 64 elementos, los cuales están relacionados con los 64 archivos de datos de entrada. Cada archivo contiene 301 filas por 4 columnas (figura 3.3). La primera fila es el encabezado, donde se colocan nombres para determinar la función de cada columna. En la primera columna llamada Imagen, está presente el número de imagen leída desde el archivo de datos. Las siguientes dos columnas son utilizadas para almacenar las posiciones 2D de la región de interés, llamadas Fila y Columna respectivamente. Enseguida está la columna con nombre D. Euclidiana, en esta se encuentran los valores almacenados del cálculo euclidiano entre las posiciones 2D de la imagen actual con una anterior como se muestra en la figura 3.4. En la sección de análisis de imágenes se define la manera de calcular esta distancia para todo el conjunto de imágenes.

4. Archivos C. Manual. Los elementos que pertenecen al conjunto de archivos de características son actualizados colocando una nueva columna en la que se escribe de manera manual uno de los tres comportamientos definidos en la sección de clasificación manual. La figura 3.5 muestra esta actualización junto

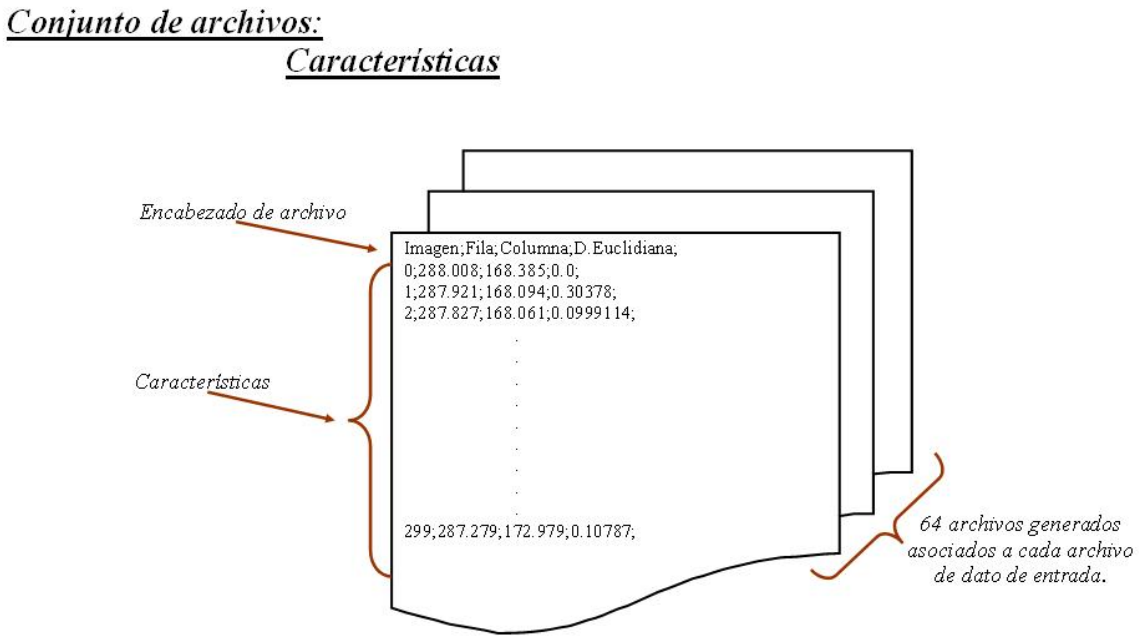


Figura 3.3: Archivos de características.

con la clasificación manual concluida. En la sección clasificación manual se describe la forma en como esto es realizado.

5. Archivos para KNN. Para trabajar con el algoritmo de clasificación supervisada KNN es necesario tener dos conjuntos: el de entrenamiento y el conjunto a clasificar. Para obtener estos dos conjuntos se divide el conjunto de archivos de clasificación manual en dos subconjuntos de 32 elementos cada uno. El primer subconjunto es el conjunto de entrenamiento para KNN y el segundo subconjunto es el conjunto a ser clasificado por KNN. Para cada elemento de este último subconjunto se agrega una nueva columna colocando el nombre KNN la cual es utilizada por el algoritmo de clasificación KNN (figura 3.6).

6. Archivos HMM. El modelo oculto de Markov se forma por medio de observaciones las cuales son obtenidas por el algoritmo de clasificación KNN, por esta razón el conjunto de clasificación con KNN se divide en dos subconjuntos de la misma cardinalidad. El primer subconjunto de 16 elementos se utiliza para generar el modelo oculto de Markov. En la figura 3.7 se muestra la forma que tiene este conjunto de archivos.

7. Archivos Viterbi. Por último se tienen los conjuntos de archivos ocupados para obtener los resultados finales de este proyecto. Este conjunto está formado por

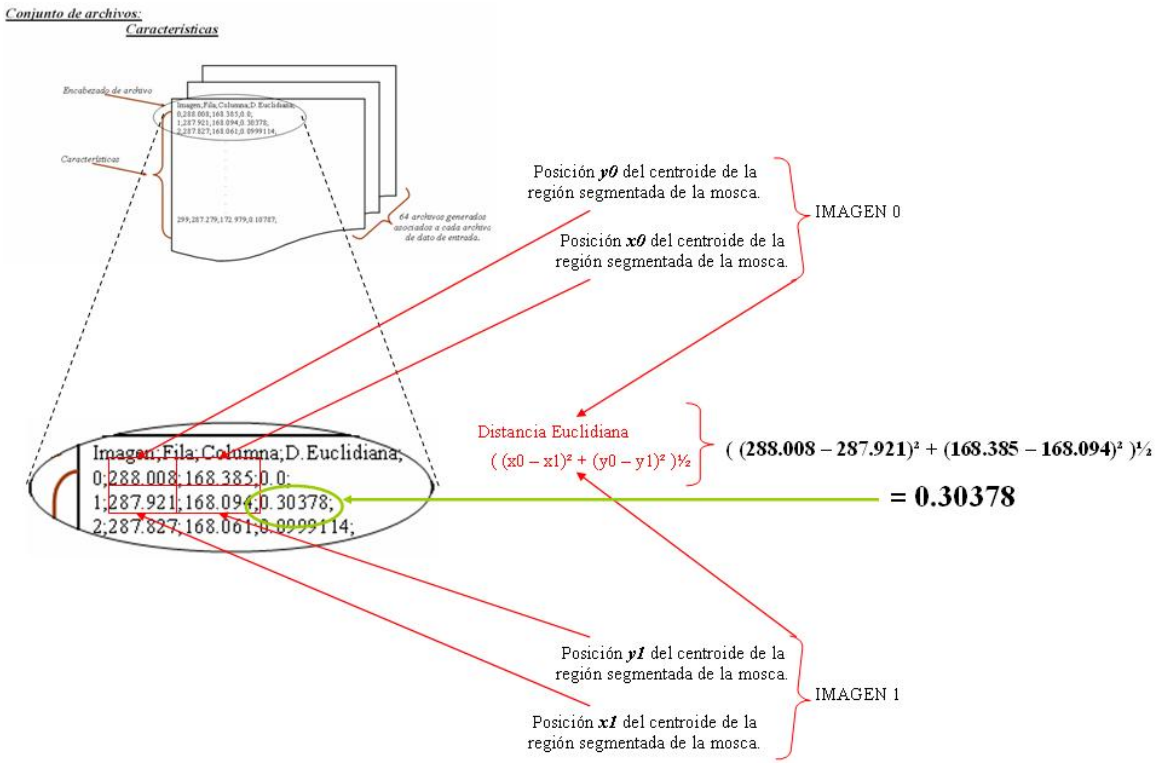


Figura 3.4: Cálculo de la distancia euclidiana.

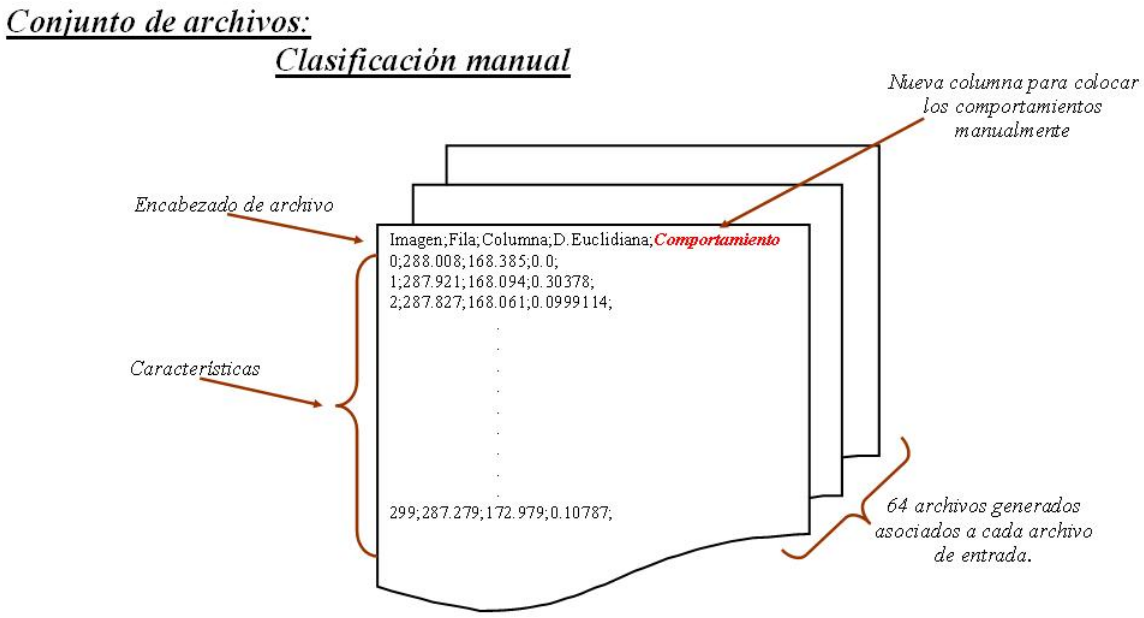


Figura 3.5: Conjunto de archivos clasificados manualmente.

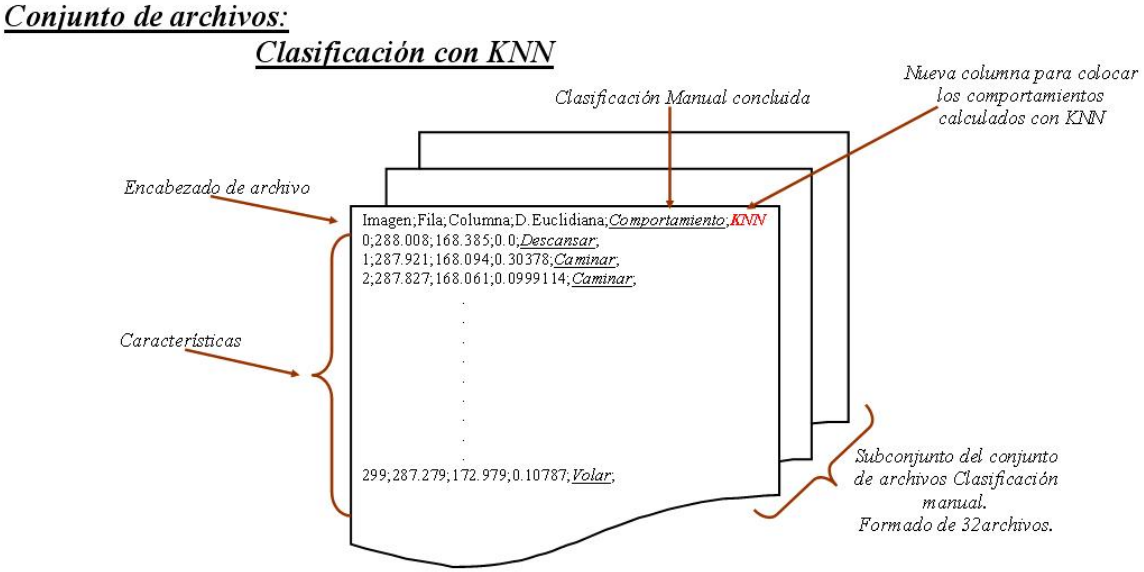


Figura 3.6: Conjunto de archivos a ser clasificados con KNN.

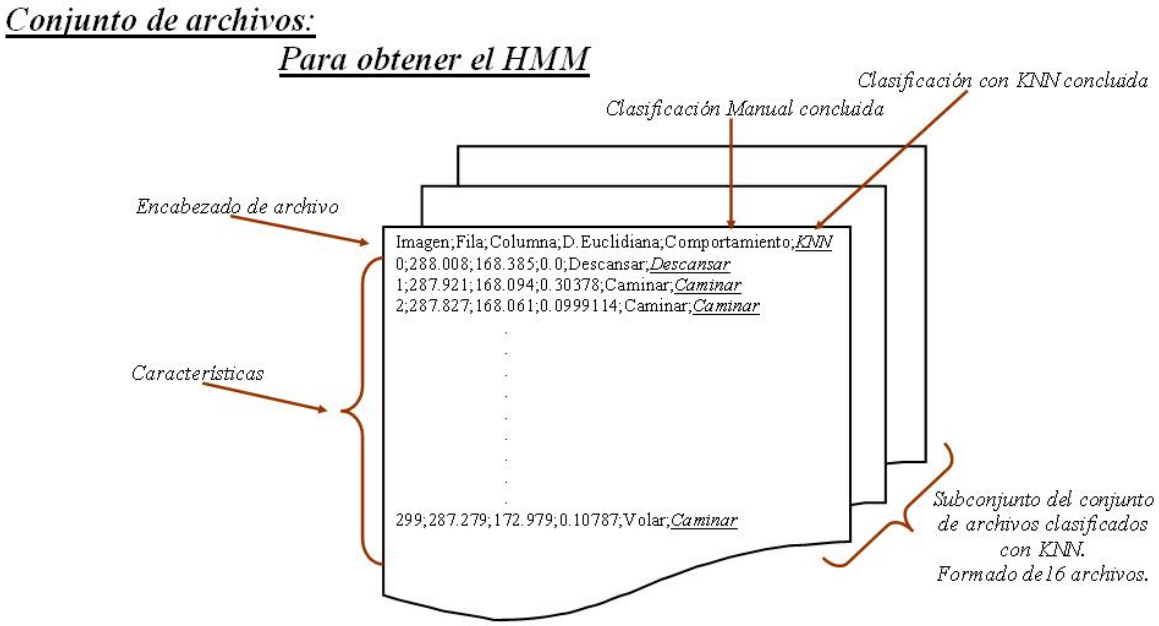


Figura 3.7: Conjunto de archivos ocupado para generar el modelo oculto de Markov.

16 archivos que pertenecen al subconjunto restante del conjunto de archivos para KNN. En cada uno de estos archivos se crea una nueva columna para los resultados finales obtenidos por el algoritmo de Viterbi. En la figura 3.8 muestra la actualización de los archivos que serán utilizados al final del proyecto.

Los archivos para obtener los resultados son generados de manera automática y son utilizados por cada uno de los procesos que se describirán a continuación.

3.2. Análisis de imágenes

El enfoque del análisis de imágenes se basa en la segmentación y la obtención de posiciones 2D de la región de interés, en este caso la región segmentada de la mosca. El problema que se presenta en la segmentación es garantizar que la región obtenida sea la región de interés, ya que si se obtuvieran más de una región de interés se tendría que decidir cuál región debería ser tomada como la correcta. Por esta razón el análisis está basado en 3 subsecciones importantes: observación de imágenes, determinar parámetros de segmentación y por último la segmentación.

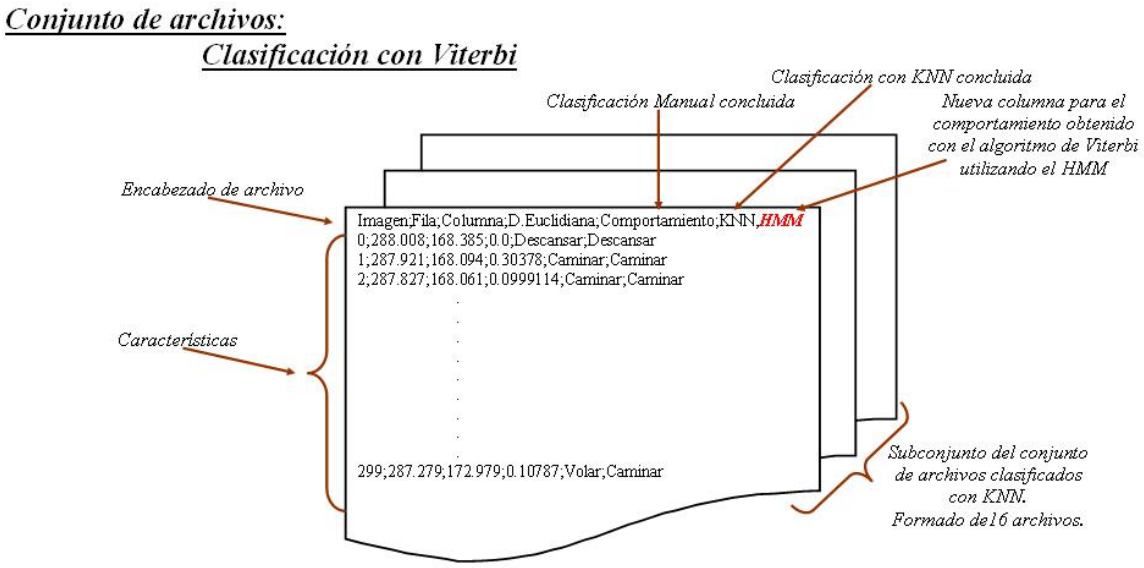


Figura 3.8: Conjunto de archivos ocupado por el algoritmo de Viterbi.

3.2.1. Observación de imágenes

En el método de segmentación algunos parámetros son necesarios para garantizar una sola región de interés. Por esta razón es imprescindible observar las imágenes obtenidas. La observación de las imágenes se realiza para analizar posibles variaciones de luz, posibles mezclas de regiones (merge) o separaciones de la región de interés (split). Cada imagen se va observando en un sistema generado en la aplicación Halcon la cual ayuda a determinar la variación del área de la región de interés en cada imagen.

En el transcurso de la observación de imágenes se determina que es posible obtener los parámetros necesarios para tener una sola medida de la región de interés debido a que el ambiente es totalmente controlado. El ambiente donde la cámara captura las imágenes y donde la mosca se encuentra es un poliedro en forma de pirámide hacia abajo con un plano que corta la punta formando una base de 4cm x 4cm, como se muestra en la figura 3.9. Se observa que la mosca tiene varios comportamientos en lapsos de tiempo muy cortos alrededor de 20 segundos, en ocasiones está comiendo y tomando agua, mucho del tiempo pasa caminando y bastante tiempo pasa sacudiendo las patas delanteras y traseras.

Se observa que las imágenes tomadas con la resolución de 340 x 340 en niveles de gris no es posible detectar a simple vista características importantes como el tamaño, la forma y el color. Con respecto al tamaño y la forma varían dependiendo de la posición en que se encuentre la mosca en cada momento de capturar una imagen. El color de la mosca también es importante debido a que de esta manera sería posible

determinar la dirección en la que está colocada, ya que los ojos de este tipo de moscas es de color verde a diferencia de todo su demás cuerpo.

En el transcurso de esta observación se detecta que existe un problema de división de regiones presente en la región de interés debido a que los niveles de gris que pertenecen a la región del agua y a la región de la comida son iguales que la región de la mosca. y en cada una de las regiones de agua y de comida se puede verificar que existe un círculo con niveles de gris mucho mas bajos que los niveles de gris de la mosca y del resto del área de comida y agua. Por lo cual en esta sección es posible determinar división de regiones en esta área.

Las observaciones que se realizan en esta sección son utilizadas para poder determinar los parámetros de segmentación necesarios en la proxima sección y de esta forma obtener una sola región de interés.

En la sección siguiente se definen los valores necesarios para cada parámetro que se utiliza en la segmentación de cada una de las imágenes garantizando la obtención de una sola región de interés perteneciente a la región de la mosca.

3.2.2. Determinar parámetros de segmentación

El método de segmentación necesita ciertos parámetros para obtener el centroide, representado en coordenadas 2D, de la región de interés. Estos parámetros de segmentación deben garantizar obtener un solo centroide para cada una de las imágenes en el video y que este pertenezca a la región de interés. Para resolver este problema es necesario tener regiones que sea posible analizarlas y entonces determinar cual pertenece a la región de interés utilizando los parámetros adecuados. La forma más común de obtener regiones en movimiento, en el análisis de imágenes, es realizar una resta de imágenes entre la imagen actual y la imagen del fondo, el resultado es la obtención de varias regiones. De esta manera se realiza este proyecto, se obtiene la imagen del fondo y enseguida se realiza una resta entre cada una de las imágenes obtenidas. El procedimiento completo de segmentación se define en la siguiente subsección, en esta subsección solo se determina la forma de obtener la imagen del fondo ya que es importante para determinar los parámetros de segmentación que más adelante se mencionarán.

La imagen del fondo es obtenida utilizando el algoritmo llamado substracción del fondo adaptable basado en mezclas gaussianas, el cual no se implementa en este proyecto, sólo sirve como herramienta para generar la imagen del fondo de todas las imágenes. El algoritmo de substracción del fondo adaptable está implementado de tal forma que solo recibe como parámetro de entrada la ruta del directorio donde se tienen las imágenes utilizadas para la obtención de la imagen del fondo. Este algoritmo está implementado en Visual C++ 6.0, en la cual se tiene una variable local llamada *pathfiles* en donde se coloca la ruta de las imágenes para obtener la imagen del fondo.

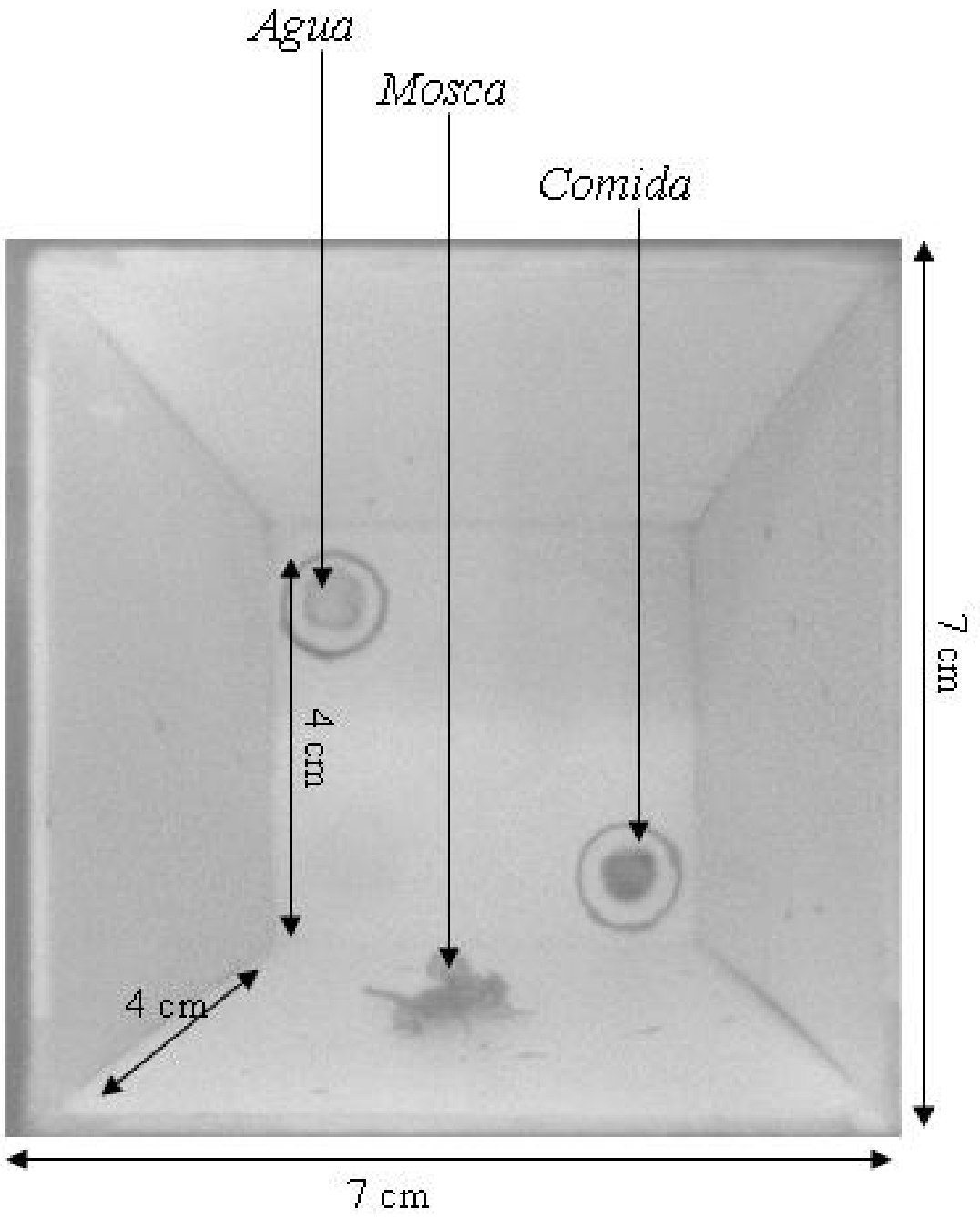


Figura 3.9: Habitad de la mosca para el experimento.

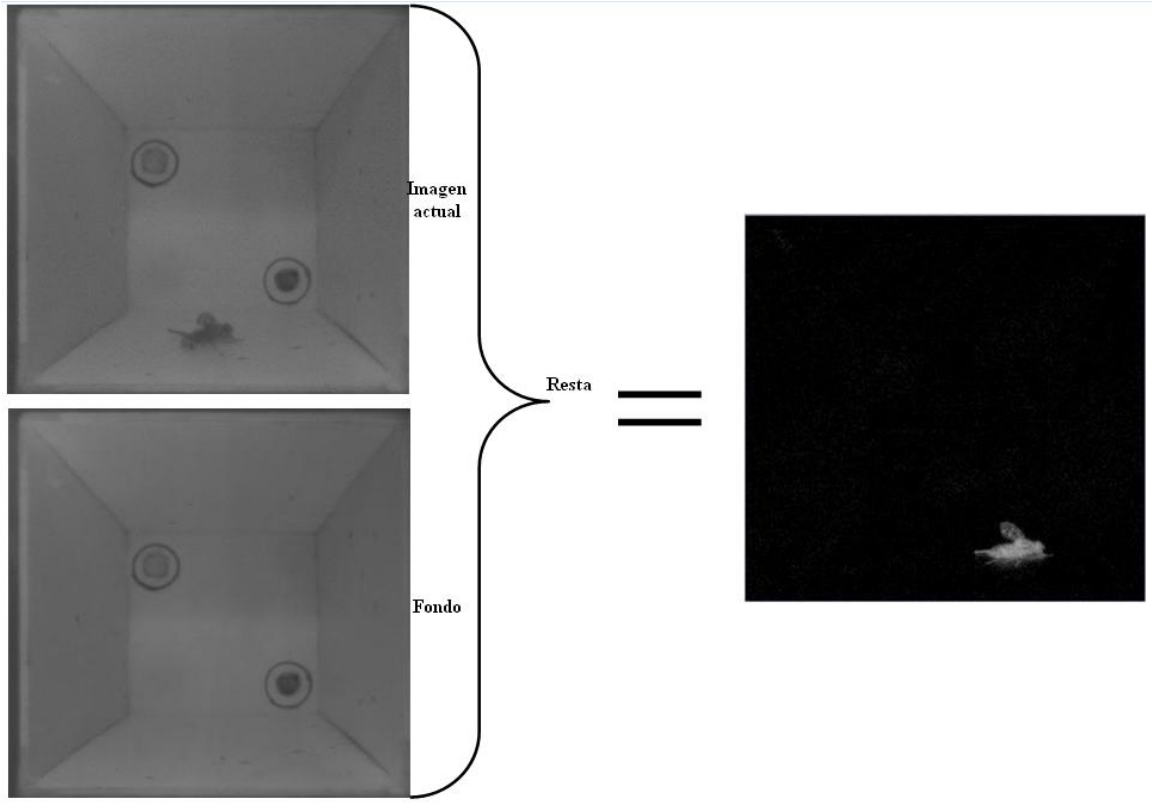


Figura 3.10: Imagen resultante de la resta entre el fondo con una imagen actual.

Teniendo hasta este punto el análisis de la observación de todas las imágenes, la imagen de fondo obtenido con el algoritmo basado en mezclas gaussianas y el procedimiento para obtener las regiones ocupando la resta entre imágenes, se analizan las imágenes resultantes de la resta entre la imagen actual con la imagen de fondo para determinar los parámetros de segmentación necesarios. Estos parámetros son dos, el primero es el umbral de nivel de gris en la región de interés y el segundo es el umbral que determina el área de la región de interés. En la figura 3.10 se muestra la manera en cómo se obtiene la región de interés para cada una de las imágenes obtenidas inicialmente. Se muestra también la región a ser analizada.

Para determinar los dos umbrales (parámetros de la segmentación) se verifican las imágenes en donde exista división de la región de interés, estas imágenes son cuando la mosca está sobre la región del agua o de la comida ya que en estos puntos se tienen niveles de gris similares entre la región de interés y circulo en la imagen de fondo donde se encuentra la comida o el agua. Es posible determinar el umbral de nivel de gris para la región de interés ya que esta región es detectada en la resta entre el fondo

y la imagen actual. Por otra parte el área se determina dependiendo del tamaño de la región más grande. Los valores iniciales que se ocupan y prueban para todas las imágenes obtenidas son los siguientes: nivel menor de gris es 80 y el nivel de gris mayor es de 255, mientras que el umbral del área es el siguiente: el valor mínimo del área es 90 pixeles y el valor mayor es 99999. los valores máximos de los dos umbrales se colocan a su máximo valor ya que se intenta acotar sólo el valor mínimo tanto de nivel de gris como del área correspondiente. Con este primer intento se tiene el problema de división de la region de interés (split) como se muestra en la figura 3.11, se tienen 3 imágenes en donde la región de interés se divide en varias regiones. Después del análisis correspondiente y pruebas continuas con niveles de gris más bajos y valores mayores de área es posible obtener una sola región que no tiene el problema de división y que corresponde a toda el área de la región de interés. Los valores adecuados para este problema son los siguientes: ***nivel menor de gris es 50 y el nivel de gris mayor es de 255, mientras que el umbral del área es el siguiente: el valor mínimo del área es 900 pixeles y el valor mayor es 99999 pixeles.*** En la figura 3.12 se muestra la obtención de una sola región sin problema de división en la región de interés, esto es posible con los parámetros adecuados antes mencionados.

Teniendo los parámetros necesarios para la segmentación que garantizan obtener una sola región y que además esta pertenece a la región de interés se genera un algoritmo para obtener el conjunto de archivos de características descritos con anterioridad de forma automática. El algoritmo de segmentación se presenta a continuación.

3.2.3. Segmentación

Con los parámetros de segmentación bien definidos es necesario generar un algoritmo para crear el conjunto de archivos de características de forma automática. El algoritmo se implementa en Visual C++ 6.0 ya que se utilizan librerías del sistema Halcon para obtener la región de interés con los parámetros de segmentación encontrados. El algoritmo inicia en la substracción de la imagen de fondo con la imagen en un tiempo t, enseguida se seleccionan las regiones con niveles de gris dependientes del umbral y dependientes del umbral del área. Por último se obtienen las posiciones 2D de la región de interés y después se actualiza el archivo de características correspondiente a cada conjunto de imágenes. El algoritmo que se ocupa está presente a continuación (2).

En la figura 3.13 se muestra el proceso de segmentación para una sola imagen del conjunto de imágenes. Por otra parte como se muestra en el algoritmo anterior, se generaliza este proceso para todas las imágenes del conjunto.

Enseguida se procede con la clasificación automática con KNN y para ello es necesario tener un conjunto de entrenamiento debido a que este clasificador es

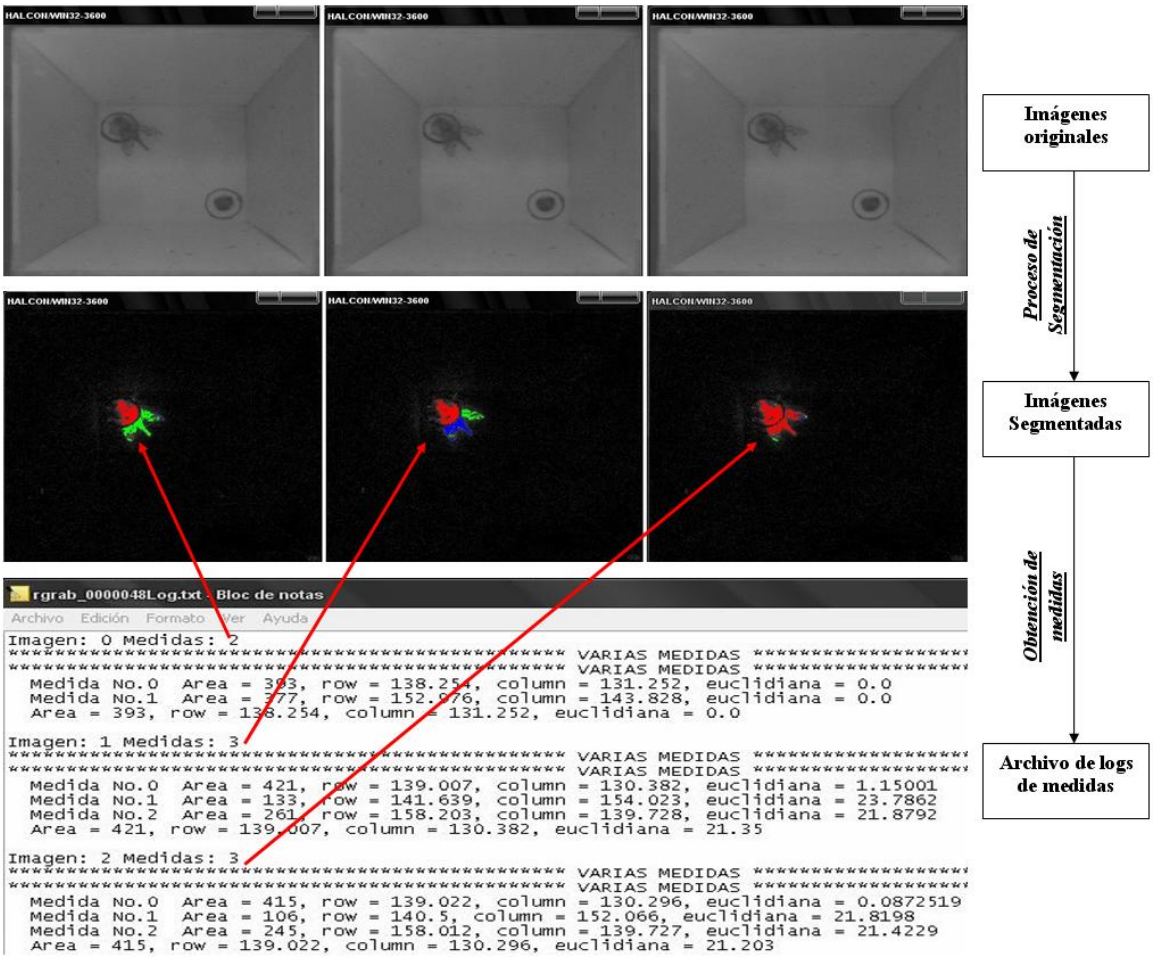


Figura 3.11: División de la región de interés.

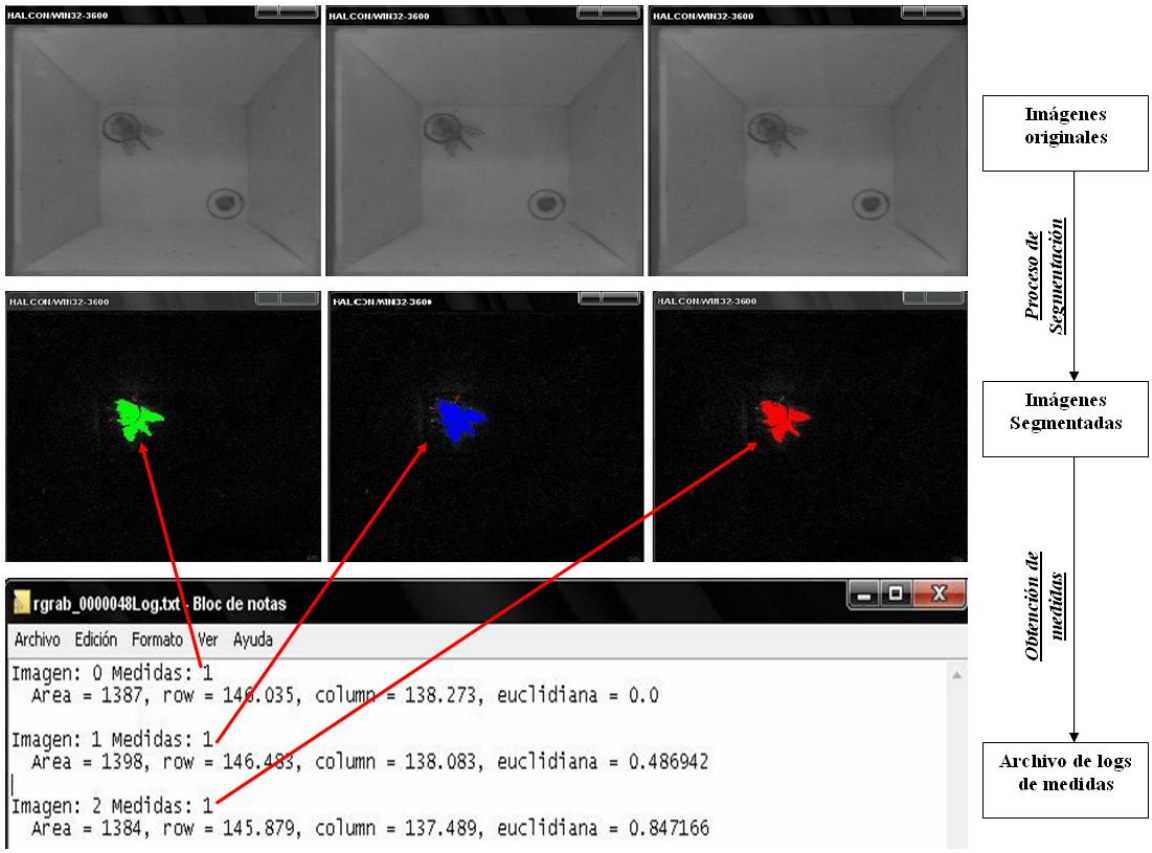


Figura 3.12: Una sola región que pertenece a la región de interés.

Algorithm 2: Algoritmo de segmentación.

Data: imagen fondo
Data: $C' = \{I_1, I_2, ...I_{19200}\}$ // conjunto de imágenes.
Result: Archivos de características.
$Fondo = LecturaImagen(fondo)$
$TotalImage = 64 * 300$
$minGray = 50$
$maxGray = 255$
$features = area$
$operation = and$
$minT = 900$
$maxT = 99999$
begin
  for $i = 1; i \leq TotalImage; i++$ **do**
    ImageI = LecturaImagen(I[i])
    Img = Resta(ImagenI, Fondo)
    RegionesI = SeleccionaObjetos(Img, minGray, maxGray)
    RegionesII= SeleccionaObjetos(RegionesI, features,operation,minT,maxT)
    Area = Centroide(RegionesII)
    Almacenar(Area, archivos_características)
  end
end

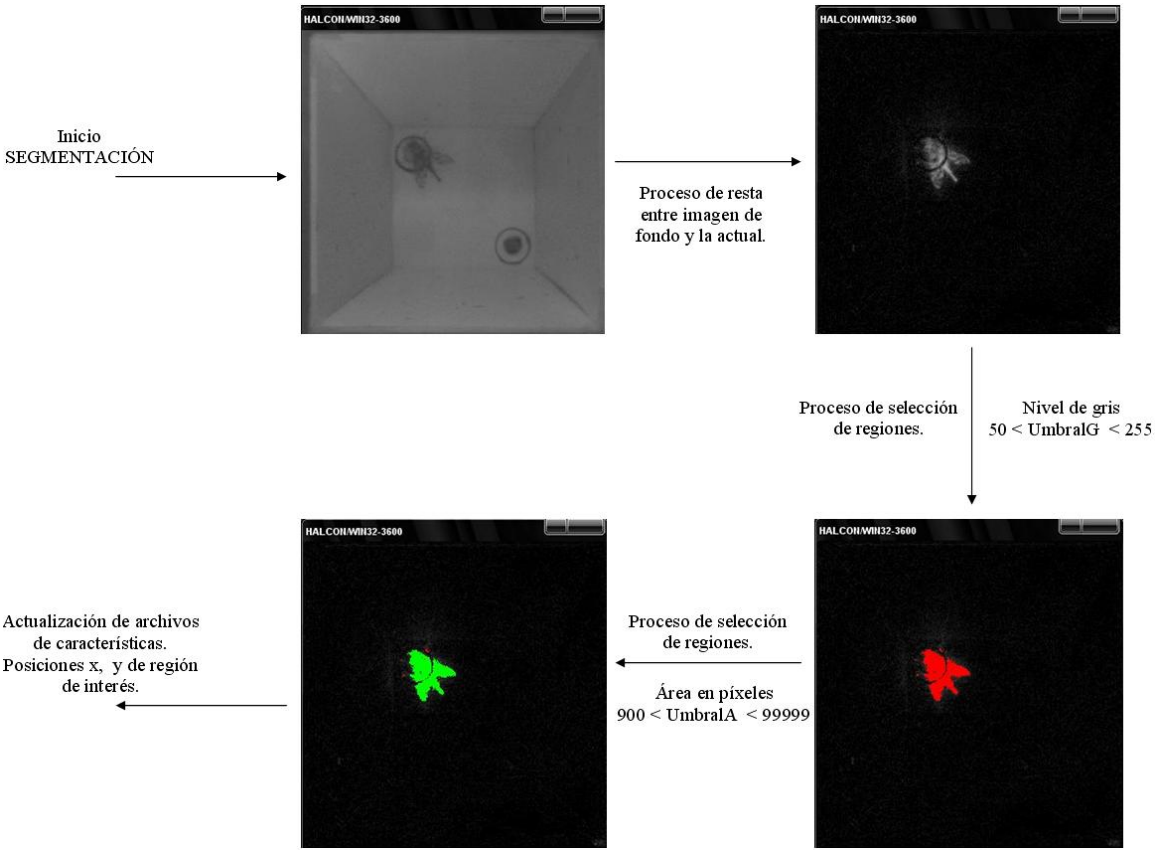


Figura 3.13: Una sola región que pertenece a la región de interés.

supervisado. Por lo tanto se presenta a continuación la subsección de clasificación manual, esta explica la forma de actualizar los archivos de características obtenidos hasta el momento.

3.3. Clasificación manual

Los algoritmos de clasificación automática ocupados en este proyecto de tesis son algoritmos de clasificación supervisados. El conjunto de entrenamiento es indispensable para este tipo de algoritmos de clasificación, por lo cual se actualizan de forma manual todos los archivos que pertenecen al conjunto de archivos de características (figura 3.3) de tal forma que exista una nueva columna que representa el comportamiento que tiene la mosca en cada una de las imágenes obtenidas. Esta actualización del conjunto se muestra en la figura 3.5. Todos los archivos son clasificados manualmente ya que es importante para generar los resultados finales. Enseguida se definen los tres comportamientos que el sistema reconoce bajo tal definición.

Descansar, Caminar, Volar se definen de la siguiente manera:

1. ***Descansar.*** Es el comportamiento en el cual la mosca no se mueve algún pixel o cuando esta se mueve a lo mas 15 píxeles de su centroide, todo esto en un lapso de 5 frames. De forma visual es posible notar que la mosca solo sacude sus patas delanteras o traseras sin caminar, o simplemente gira al rededor de su centroide.

2. ***Caminar.*** Es el comportamiento en el cual la mosca tiene un desplazamiento de entre 16 y 30 píxeles en el lapso de 3 frames. Visualmente se puede determinar cuando la mosca está caminando por todo el panel donde se encuentra encerada. No se tiene una deformación de la región como en el siguiente comportamiento.

3. ***Volar.*** Es el comportamiento que se determina cuando la mosca se mueve mas de 30 píxeles de un frame a otro. Visualmente se observa que la región de la mosca tiene un cambio brusco de posición y sufre una deformación debido a que el cambio de posición es mucho más rápido que velocidad de la cámara.

Para actualización del conjunto de archivos de características o clasificación manual se observan nuevamente cada una de las imágenes obtenidas y se determina, en base a la definición anterior, el comportamiento presente que la mosca tiene. Se crea un sistema de visualización de imágenes con un módulo que interacciona con el conjunto de archivos de características con el objeto de actualizar estos archivos dependiendo del comportamiento que se coloque en el sistema. Este sistema es generado para que

la actualización sea más rápida ya que se observa que en lapsos de tiempo muy largos la mosca está caminando o bien descansando y rara ves está volando.

La figura 3.14 muestra el proceso de clasificación manual, muestra como del conjunto de archivos es seleccionado el número 48 y enseguida en la visualización se observa que de la imagen 0 a la imagen 96 la mosca está descansando, por lo cual el comportamiento colocado para esa serie de imágenes es descansar. Se muestra también que el archivo de características se actualiza con ese comportamiento. De esta manera cada una de las imágenes que pertenece al conjunto de imágenes es etiquetada con un comportamiento particular.

Después de concluir con este proceso, entonces es posible ocupar cualquier algoritmo de clasificación supervisada ya que se pueden formar subconjuntos que sean los responsables del entrenamiento del algoritmo. En este proyecto se utiliza el algoritmo de KNN para obtener las observaciones necesarias para el algoritmo de Viterbi. En seguida se presenta la sección de clasificación automática que abarca los dos algoritmos antes mencionados junto con el proceso de obtención del modelo oculto de Markov.

3.4. Clasificación automática

Con la clasificación manual concluida se tiene un conjunto de archivos de características actualizado con una columna nueva en el cual se escribe el comportamiento correspondiente a cada una de las imágenes. En este punto se divide este conjunto de archivos en subconjuntos de entrenamiento y clasificación para los dos algoritmos de clasificación ocupados. También se obtiene un subconjunto para obtener el modelo oculto de Markov.

3.4.1. Clasificación con KNN

Para clasificar las imágenes con el algoritmo de clasificación supervisada KNN se divide en dos subconjuntos el conjunto de archivos clasificados manualmente con el objeto de tener un conjunto de entrenamiento y un conjunto de clasificación respectivamente. El segundo subconjunto de archivos para la clasificación se actualiza con una nueva columna con etiqueta KNN la cual es colocada de forma automática por el algoritmo. Se realizan varias pruebas para determinar el valor preciso de K, el cual se coloca en 7. El algoritmo se divide en varios métodos que van desde leer archivos de datos hasta la escritura en ellos, pasando por la lectura de cada una de las líneas que representan las imágenes. En el algoritmo 3 inicia con la obtención de cada una de las 300 líneas de cada uno de los 64 archivos y enseguida pasar esta información al método de clasificación el cual trabaja con KNN para obtener la clase correspondiente de cada imagen. El método de clasificación se muestra en el algoritmo

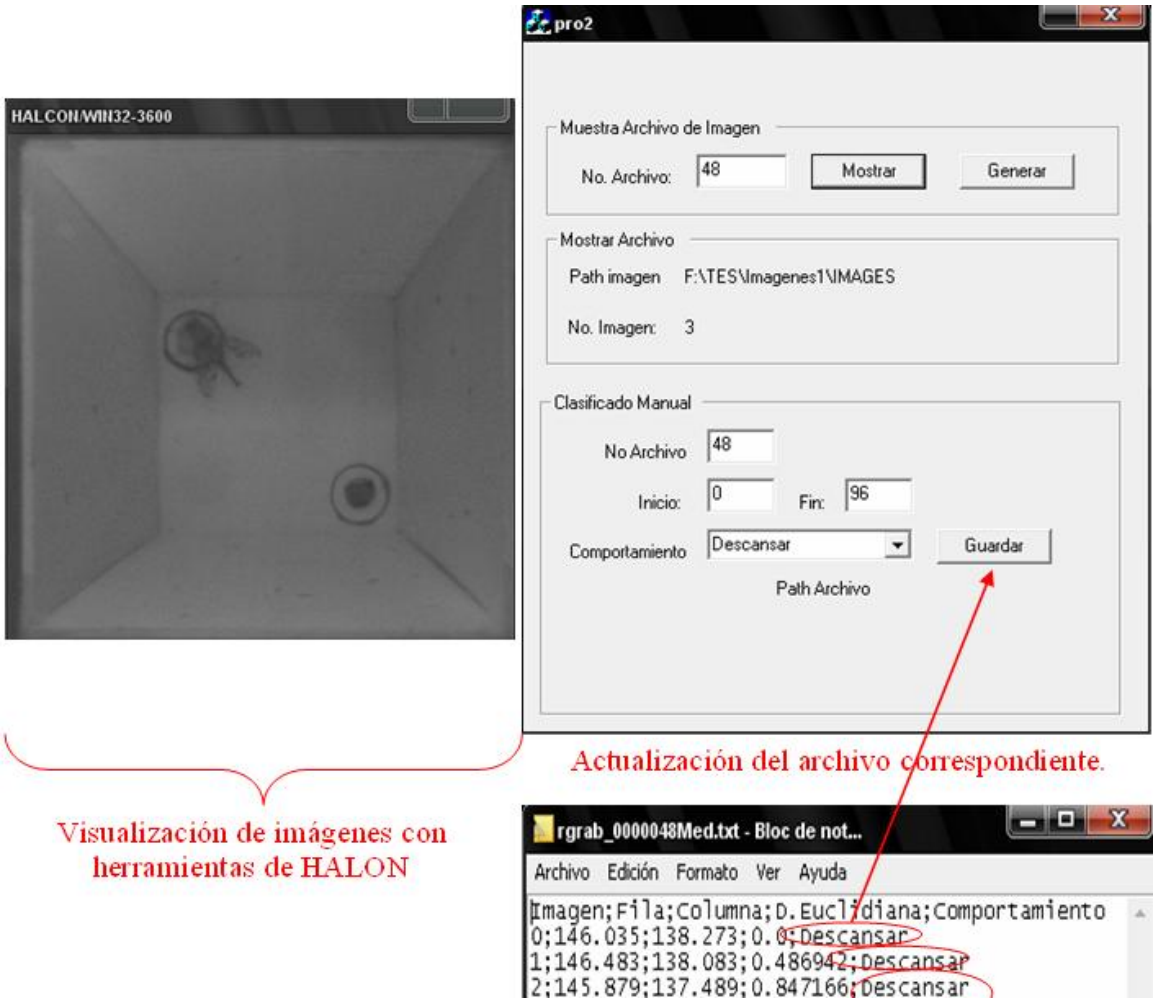


Figura 3.14: Proceso de clasificación manual actualizando los archivos de características.

4, es posible verificar que la única característica en la cual es basado este proyecto es la distancia euclidiana. En primer lugar se tiene la distancia euclidiana obtenida del centroide de la región segmentada de la mosca en un tiempo t con un tiempo *t+1*, estas distancia euclidianas están presentes en el algoritmo representadas como D. Euclidiana y D.EuclidianaIJ correspondientes a la imagen de entrenamiento y a la imagen a ser clasificada por KNN respectivamente. Por otra parte el cálculo euclidiano que se realiza sobre estas distancias es la base de KNN ya que calcula esta distancia para clasificar la nueva instancia con el valor mínimo de este cálculo, además de ser la clase que aparece mayor cantidad de veces en un vector de K clases.

Algorithm 3: Obtención de características de imágenes representadas en líneas de archivos de clasificación manual.

Data: $A = \{A_1, A_2, ...A_{64}\}$ // conjunto de archivos clasificados manualmente.
Result: $A' =$
$\{A'_3 2, A'_3 3, ...A'_{64}\}$ // subconjunto de archivos clasificados con KNN.
$enteroK = 7$
begin
  for $i = 32; i \leq 64; i++$ **do**
    archivo A = LecturaArchivo(A[i])
    for $j = 0; j \leq 300; j++$ **do**
      decimal D. Euclidiana = lectura de columna(3,A)
      clase C = lectura de columna(4,A)
      clasificación(D. Euclidiana, C, K)
    end
    actualiza archivo(archivoI,claseKNN)
  end
end

Este algoritmo es implementado obteniendo como resultado un conjunto de archivos de 32 elementos que se actualizaron con una nueva columna llamada KNN como lo muestra la figura 3.6.

Como se ha mencionado con anterioridad, el algoritmo de Viterbi necesita observaciones y un modelo oculto de Markov para obtener resultados de suavizado de tales observaciones. Las observaciones a ser suavizadas son parte de los archivos generados por KNN y la otra parte de estos archivos son utilizados para formar el modelo oculto de Markov. Enseguida se describe la forma en que el único modelo oculto es obtenido y la manera en como se construye.

Algorithm 4: Algoritmo de clasificación con KNN.

```
Data: decimal D. Euclidiana, clase C, entero K
Result: clase_nueva
begin
    for i = 0; i ≤ 32; i++ do
        archivo A = LecturaArchivo(A[i])
        for j = 0; j ≤ 300; j++ do
            línea l = lectura de línea(j,A)
            decimal D. EuclidianaIJ = lectura de columna(4,A)
            vector euclidiano[i][j] =
            calcula distancia euclidiana(D. EuclidianaIJ, D. Euclidiana)
        end
        Ordenar vector eclidiano[i][j] en orden ascendente
        clases[] = Obtener los K casos mas cercanos a C del vector euclidiano[ ][ ]
        claseKNN = clase mas frecuente de clases []
        actualiza archivo(archivoI,claseKNN)
    end
end
```

3.4.2. Obtención del modelo oculto de Markov

Para obtener el modelo oculto de Markov es necesario encontrar la matriz de transición de estados, la matriz de emisión de estados y el vector de probabilidad inicial de presencia de los estados. Para realizar esta tarea se necesita un conjunto de entrenamiento el cual contenga la clasificación manual y la clasificación con KNN. El conjunto clasificado con KNN en la sección anterior consta de 32 elementos, en esta sección este conjunto de archivos es dividido en dos subconjuntos de 16 archivos cada uno. El primer subconjunto es utilizado en esta sección para obtener el modelo oculto de Makov, es decir, es ocupado para obtener la matriz de transición de estados, la matriz de emisión de estados y el vector de probabilidad inicial de presencia de los estados.

Matriz de transición de estados

La matriz de transición de estados es obtenida de una sola columna de cada uno de los archivos del subconjunto de 16 archivos destinado para este propósito. La columna que es utilizada es la obtenida en la clasificación manual, ya que de esta forma se tiene la probabilidad de aparición de cada uno de los comportamientos que la mosca tiene realmente.

Se tienen 3 comportamientos por lo cual se tienen 3 x 3 elementos de la matriz

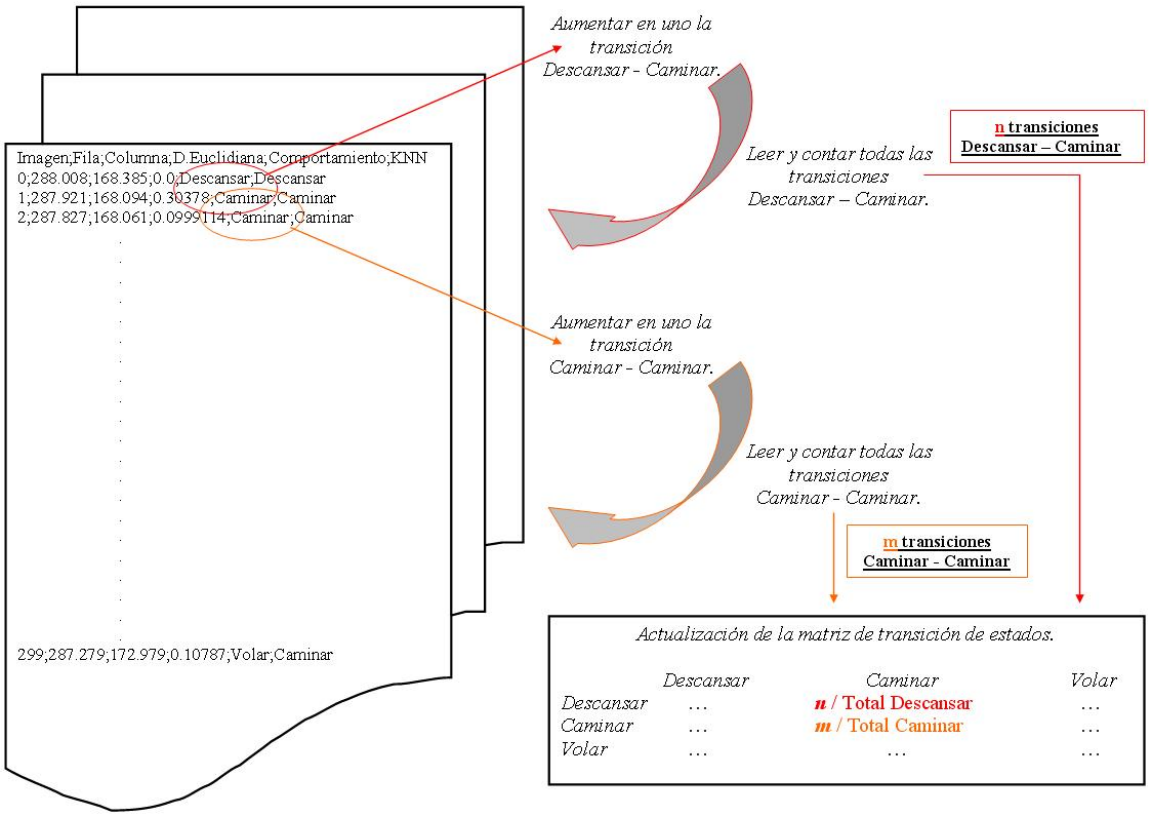


Figura 3.15: Proceso de obtención de la matriz de transición de estados.

de transición de estados, esto es por que para cada una de las clases se obtiene la probabilidad de transición al siguiente estado. Por ejemplo, si se está en el estado ***Descansar*** y se verifica que en la siguiente imagen de la secuencia pasa al estado ***Volar*** entonces se incrementa en uno la transición ***Descansar*** a ***Volar***. Los valores de la matriz de transición deben ser menores o iguales a 1 lo cual representa la probabilidad de pasar de un estado a otro, por lo cual el valor final obtenido para cada transición se divide entre el total de veces que aparece el estado en la columna. En la figura 3.15 se visualiza la manera en como se obtienen estos valores de esta matriz.

Matriz de emisión de estados

De la misma forma que la matriz de transición de estados es obtenida, se genera esta otra matriz de emisión de estados. La diferencia radica en la fuente de datos ocupados para formar esta matriz. Esta matriz es creada tomando los datos de dos

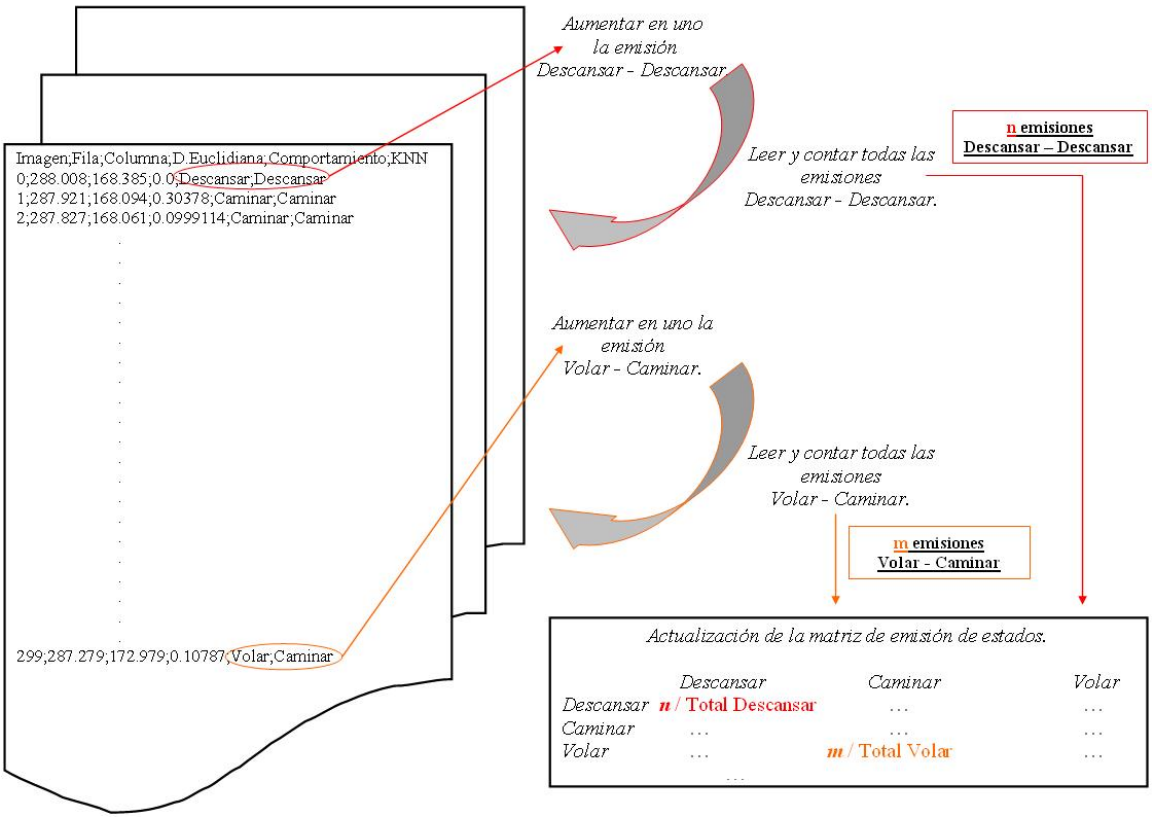


Figura 3.16: Proceso de obtención de la matriz de emisión de estados.

columnas de los archivos para este propósito, las columnas que son ocupas son la de clasificación manual y clasificación con KNN. De esta forma se tiene la probabilidad de aparición de cada uno de los comportamientos que la mosca tiene realmente con el comportamiento que ha observado el algoritmo KNN.

Se tienen 3 comportamientos por lo cual se tienen 3 x 3 elementos de la matriz de emisión de estados, esto es por que para cada una de las clases se obtiene la probabilidad de emisión al siguiente estado. Por ejemplo, si de forma manual se coloca un estado ***Descansar*** y el algoritmo KNN clasifica esta imagen con un estado ***Volar*** entonces se incrementa en uno la emisión ***Descansar*** a ***Volar***. Los valores de la matriz de emisión deben ser menores o iguales a 1 lo cual representa la probabilidad de pasar de un estado a otro, por lo cual el valor final obtenido para cada transición se divide entre el total de veces que aparece el estado en la columna de clasificación manual. En la figura 3.16 se visualiza la manera en como se obtienen estos valores de esta matriz.

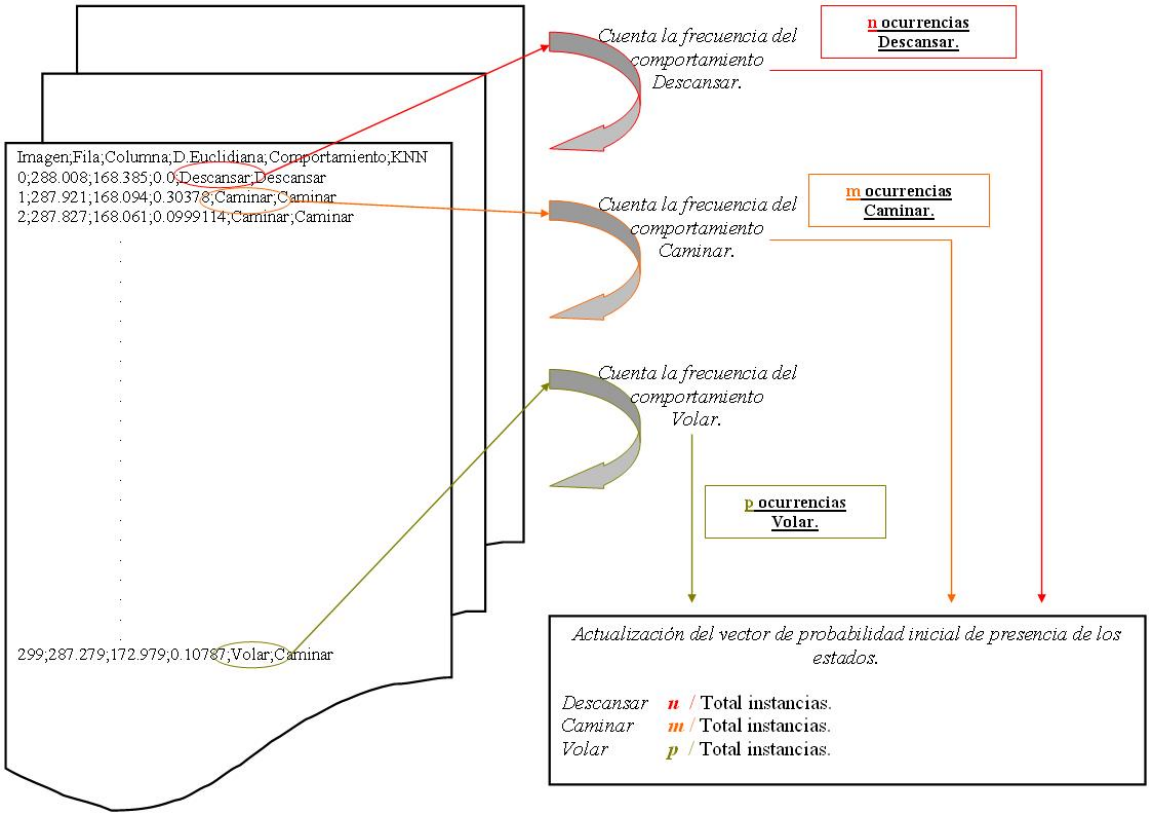


Figura 3.17: Proceso de obtención del vector de probabilidad inicial de presencia de los estados.

Vector de probabilidad inicial de presencia de los estados

Esta vector se genera apartir de la columna de comportamiento manual ya que se tienen las probabilidades reales de ocurrencia de cada uno de estos estados o comportamientos.

Se tienen 3 comportamientos por lo cual se tienen 3 elementos del vector, esto es por que para cada una de las clases se obtiene la probabilidad de ocurrencia de cada estado. Por ejemplo, si aparecen 10 ocurrencias de que la mosca está en un estado ***Volar*** y se tienen 3000 imágenes entonces la probabilidad de que vuele la mosca es muy mínima, en este caso sería del .3 % de probabilidad que la mosca vuele. De esta forma se obtiene este vector de probabilidad inicial de presencia de estados. En la figura 3.17 se visualiza la manera en como se obtienen los valores de este vector.

3.4.3. Clasificación con Viterbi

El modelo oculto de Markov y las observaciones obtenidas en los procesos anteriores son los parámetros necesarios para el algoritmo de Viterbi. Este algoritmo reclasifica las observaciones utilizando el HMM encotnrado maximizando la probabilidad de las observaciones dependiendo del modelo basándose en la ecuación 3.4.1. El algoritmo realiza una búsqueda de manera recursiva de la secuencia de observación más probable dependiendo del modelo oculto de Markov. Para este proyecto de tesis solo se ucupa una librería que tiene implementado el algoritmo de Viterbi el cual recibe como parámetros dos clases, observaciones y modelo oculto de Markov.

$$\Pi_{i=1,n} P(w_i|t_i)P(t_i|t_{i-1}) \tag{3.4.1}$$

El algoritmo utilizado para realizar esta reclasificación de estados se presenta en el algoritmo 5, se tienen 4 parámetros de entrada de los cuales los primeros 3 son parámetros del modelo oculto de Markov y el parámetros restante representa el vector obtenido por la clasificación KNN. El resultado final del algoritmo Viterbi es un nuevo vector de comportamientos representados en números enteros los cuales se mapean a su correspondiente valor del comportamiento de la mosca. Cabe mencionar que la librería ocupada para este proyecto es llamada JAHMM implementada en JAVA por lo que esta última parte del proyecto también es implementada en JAVA. Las clases ocupadas de esta librería son dos las cuales son llamadas Hmm y ViterbiCalculator, la instancia de la primera clase se utiliza para formar el modelo oculto de Markov pasando como parámetros, a su respectivo constructor, los primeros tres parámetros ya mencionados. La siguiente instancia de la clase ViterbiCalculator es generada para la reclasificación de los estados, esta clase recibe en uno de sus constructores el modelo y las observaciones. Esta implementación del algoritmo trabaja con números enteros consecutivos positivos en las observaciones, por lo cual se mapean los nombres de los estados o comportamientos a sus respectivos números enteros consecutivos positivos. Por último se hace referencia al método stateSequence(); presente en la clase ViterbiCalculator el cual trabaja con el algoritmo de Viterbi y regresa como reultado un vector de números enteros consecutivos positivos que representan la reclasificación de los comportamientos, así que se mapean nuevamente los valores enteros a sus respectivas cadenas de caracteres correspondientes a cada uno de los estados. Por último se actualizan los archivos destinados para esta sección, los cuales se presentan en la figura 3.8, obteniendo así os resultados finales y listos para obtener la presición tanto del algoritmo de KNN y la presición del algoritmo de Viterbi ocupando el modelo oculto de Markov.

Algorithm 5: Algoritmo de clasificación con Viterbi utilizando el HMM y las observaciones obtenidas con KNN.

Data: matriz de transición Mt, matriz de emisión Me,
vector inicial Pi, vector de observaiones Obs
Result: vector de reclasificación de estados Vr
begin
  Crear instancia de la clase HMM hmm = new Hmm(Pi,Mt,Me)
  Encuentra todos los comportamientos diferentes presentes
  en el vector Obs y almacenalos en el vector Comport.
  for $i = 0; i \leq longitud\_de\_Obs; i++$ **do**
    Busca la posición k en Comport tal que Comport[k] == Obs[i]
    Almacena k en el vector obsVect
  end
  Crear instancia de la clase ViterbiCalculator vit(obsVect,hmm)
  Obtener los valores en vector reclasificación = vit.stateSequence()
  for $i = 0; i \leq longitud\_de\_reclasificacin; i++$ **do**
    entero estado = reclasificación[i]
    Almacena Corpot[estado] en vectorReclasificado[i]
  end
  Actualizar archivos de Viterbi utilizando el vector vectorReclasificación
end

Capítulo 4

Resultados

Los resultados finales son obtenidos del conjunto de archivos clasificados con el algoritmo de Viterbi. Sin embargo se tiene resultados parciales obtenidos de la clasificación con KNN y del conjunto con el cual se genera el modelo oculto de Markov, los cuales son indispensables para los resultados finales. Teniendo en cuenta lo anterior, se presenta a continuación los resultados parciales tanto del conjunto clasificado con KNN y enseguida los resultados del conjunto para generar el modelo oculto de Markov. Cabe mencionar que para cada uno de los resultados parciales incluyendo los resultados finales, están basados en la clasificación manual es decir, cada clasificación realizada tanto por el algoritmo de KNN como el algoritmo de Viterbi, es comparada con la clasificación manual de tal forma que si estas dos clasificaciones son iguales, entonces se deduce que la clasificación automática es correcta para el algoritmo de clasificación automática correspondiente. El procedimiento realizado para obtener esta comparación está compuesto de un ciclo el cual va deduciendo la igualdad entre estados de dos columnas diferentes de cada archivo que pertenece al conjunto de archivos clasificados con KNN o con Viterbi. El procedimiento 6 se ocupa para obtener un vector con la cantidad de elementos que son clasificados correctamente de forma automática.

Por otra parte es necesario obtener la precisión de cada uno de los tres estados clasificados de forma automática, por lo cual es necesario encontrar los verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos. La definición de cada uno de estos términos son las siguientes:

1. Verdaderos positivos: El conjunto de imágenes que, tanto el sistema como manualmente, son clasificados bajo el estado E_i.

2. Verdaderos negativos: El conjunto de imágenes que, tanto el sistema como manualmente, no son clasificados bajo el estado E_i.

Algorithm 6: Procedimiento para obtener el número de clasificaciones correctas.

```
Data: vectorClasificacionManalCM}
Data: vectorClasificacionAutomticaCA}
Result: vectordeClasificacinCorrectaCC}
begin
    for enteroi = 0; i ≤ longituddeCM; i++ do
        if CM[i] == CA[i] then
            if CM[i] == Descansar then
                CC[0]++
            end
            if CM[i] == Caminar then
                CC[1]++
            end
            if CM[i] == Volar then
                CC[2]++
            end
        end
    end
end
```

3. Falsos positivos: El conjunto de imágenes que el sistema clasifica bajo un estado E_i, pero manualmente se indica lo contrario.

4. Falsos negativos: El conjunto de imágenes que el sistema NO clasifica bajo un estado E_i, pero manualmente se indica lo contrario, que debían ser clasificados bajo el estado E_i.

La precisión es calculada con la ecuación 4.0.1 y esta es ocupada para cada uno de los estados.

$$Presicin = Verdaderospositivos/Verdaderospositivos + Falsospositivos. \quad (4.0.1)$$

4.1. Clasificador K-Nearest Neighbor

Como se he mencionado anteriormente el algoritmo de clasificación supervisado KNN necesita el conjunto de entrenamiento para trabajar sobre el conjunto a ser clasificado. Estos dos conjuntos son clasificados manualmente con el objeto de poder obtener los resultados, se utiliza el procedimiento 6 para obtener las clasificaciones correctas del algoritmo KNN. La precisión general de este conjunto clasificado con KNN es de 65.02 % y es obtenido del promedio de la precisión de cada uno de los estados como es posible verificarlo en la tabla 4.1.

	Descansar	Caminar	Volar
Verdaderos positivos	752	307	19
Falsos positivos	36	370	16
Verdaderos negativos	337	776	1465
Falsos negativos	337	47	0
PRECISIÓN	95.43 %	45.34 %	%54.28

Tabla 4.1: Resultados obtenidos del conjunto clasificado con el algoritmo KNN.

4.2. Modelo oculto de Markov

El modelo oculto de Markov se define mediante tres tres parámetros, matriz de transición de estados, matriz de emisión de estados y vector de probabilidad inicial de presencia de los estados. La tabla 4.2 representa el vector inicial de presencia de los estados, la tabla 4.3 es la matriz de transición de estados, y la tabla 4.4 representa la matriz de emisión de estados. Cada uno de estos vectores se obtienen del conjunto de archivos para obtener el HMM.

Descansar	0.8673333333333333
Caminar	0.12688888888888888
Volar	0.0024444444444444444

Tabla 4.2: Valores de probabilidad inicial de presencia de los estados.

	Descansar	Caminar	Volar
Descansar	0.9992313604919293	6.405329233922624E-4	1.2810658467845247E-4
Caminar	0.005253940455341506	0.989492119089317	0.005253940455341506
Volar	0.13636363636363635	0.18181818181818182	0.6818181818181818

Tabla 4.3: Matriz de transición de estados.

4.3. Resultados del Algoritmo K-Nearest Neighbor con respecto al algoritmo de Viterbi utilizando el modelo Oculto de Markov.

Para el conjunto clasificado con el algoritmo de KNN y sus subconjuntos se obtuvieron resultados parciales. Esto fue posible debido a la clasificación manual que se realiza desde el inicio del procedimiento. En la tabla 4.5 se muestra la matriz de confusión referente al algoritmo KNN. Por otro lado se tiene en la tabla 4.6 la matriz de confusión para las observaciones y en la tabla 4.7 la matriz de confusión para la reclasificación utilizando el modelo oculto de Markov. Por último la tabla 4.8 representa la matriz de confusión para el conjunto de entrenamiento del modelo oculto de Markov, es decir del conjunto donde se obtiene la matriz de transición de estados, la matriz de emisión de estados y el vector de probabilidad de presencia de estados.

Los 6 experimentos que se reealizan es con el objetivo de obtener una mayor precisión del sistema. En la tabla 4.9 se muestra la comparación del algoritmo KNN entre los diferentes experimentos dependiendo del primer comportamiento: ***Descansar***, de la misma forma la tabla 4.10 muestra la comparación del algoritmo de KNN dependiendo del segundo comportamiento: ***Caminar***. Por último se tiene la tabla 4.11 que muestra la comparación del algoritmo de KNN dependiendo del tercer comportamiento: ***Volar***. Por otro lado y de forma análoga se tienen las tablas 4.12 4.13 y 4.14 que representan la comparación de la reclasificación utilizando el modelo oculto de Markov dependiendo del comportamiento ***Descansar***, ***Caminar*** y ***Volar*** respectivamente.

Por último en la tabla 4.15 se tienen la comparación de las precisiones generales del sistema entre el algoritmo KNN y el HMM para la clasificación de los diferentes

	Descansar	Caminar	Volar
Descansar	0.9111564973193771	0.08884350268062292	0.0
Caminar	0.5856643356643356	0.4143356643356643	0.0
Volar	0.36363636363636365	0.2727272727272727	0.6818181818181818

Tabla 4.4: Matriz de emisión de estados.

	Descansar	Caminar	Volar
Descansar	9460	690	3
Caminar	1234	853	0
Volar	15	18	27

Tabla 4.5: Matriz de confusión para el algoritmo KNN.

experimentos realizados.

4.3.1. Discusión de Resultados

Con los resultados presentes en la tabla 4.15 se deduce que el experimento número 6 tiene una posición mayor. También se puede notar que un modelo oculto de Markov bien obtenido y utilizándolo de la manera correcta, puede aumentar la precisión del sistema. Es importante observar que el clasificador KNN no es muy preciso para resolver este tipo de problemáticas, sin embargo, construyendo un modelo que depende de este clasificador la precisión aumenta aplicando un algoritmo como el de Viterbi el cual maximiza la probabilidad de las observaciones obtenidas.

	Descansar	Caminar	Volar
Descansar	2171	145	3
Caminar	562	381	0
Volar	8	11	19

Tabla 4.6: Matriz de confusión para las observaciones.

	Descansar	Caminar	Volar
Descansar	2095	220	4
Caminar	142	800	1
Volar	5	13	20

Tabla 4.7: Matriz de confusión para la reclasificación de las observaciones utilizando el HMM.

	Descansar	Caminar	Volar
Descansar	7289	545	0
Caminar	672	472	0
Volar	7	7	8

Tabla 4.8: Matriz de confusión para el conjunto de entrenamiento del HMM.

	Opc 1	Opc 2	Opc 3	Opc 4	Opc 5	Opc 6
Verdaderos Positivos	2132	2205	2184	3461	3418	2171
Verdaderos Negativos	411	308	297	742	697	411c
Falsos Positivos	187	114	135	314	357	148
Falsos Negativos	570	673	684	883	928	570
PRECISIÓN %	91.94	95.08	94.18	91.68	90.54	93.62

Tabla 4.9: Comparación del algoritmo de KNN con los diferentes experimentos referentes al comportamiento ***Descansar***.

	Opc 1	Opc 2	Opc 3	Opc 4	Opc 5	Opc 6
Verdaderos Positivos	381	280	267	702	658	381
Verdaderos Negativos	2162	2237	2218	3499	3460	2201
Falsos Positivos	562	663	676	872	916	562
Falsos Negativos	195	120	139	327	366	156
PRECISIÓN %	40.40	29.69	28.31	44.60	41.80	40.40

Tabla 4.10: Comparación del algoritmo de KNN con los diferentes experimentos referentes al comportamiento ***Caminar***.

	Opc 1	Opc 2	Opc 3	Opc 4	Opc 5	Opc 6
Verdaderos Positivos	19	19	19	23	23	19
Verdaderos Negativos	3259	3259	3255	5345	5342	3259
Falsos Positivos	19	19	19	28	28	19
Falsos Negativos	3	3	7	4	7	3
PRECISIÓN %	50.00	50.00	50.00	45.10	45.10	50.00

Tabla 4.11: Comparación del algoritmo de KNN con los diferentes experimentos referentes al comportamiento ***Volar***.

	Opc 1	Opc 2	Opc 3	Opc 4	Opc 5	Opc 6
Verdaderos Positivos	2111	2081	2164	3555	3567	2095
Verdaderos Negativos	821	784	770	1428	1409	834
Falsos Positivos	208	238	155	220	208	224
Falsos Negativos	160	197	211	197	216	147
PRECISIÓN %	91.03	89.74	93.32	94.17	94.49	90.34

Tabla 4.12: Comparación de la reclasificación utilizando el HMM con los diferentes experimentos referentes al comportamiento ***Descansar***.

	Opc 1	Opc 2	Opc 3	Opc 4	Opc 5	Opc 6
Verdaderos Positivos	787	750	729	1379	1355	800
Verdaderos Negativos	2139	2102	2182	3586	3590	2124
Falsos Positivos	156	193	214	195	219	143
Falsos Negativos	218	255	175	240	236	233
PRECISIÓN %	83.46	79.53	77.31	87.61	86.09	84.84

Tabla 4.13: Comparación de la reclasificación utilizando el HMM con los diferentes experimentos referentes al comportamiento ***Caminar***.

	Opc 1	Opc 2	Opc 3	Opc 4	Opc 5	Opc 6
Verdaderos Positivos	20	14	12	23	15	20
Verdaderos Negativos	3258	3259	3253	5343	5338	3257
Falsos Positivos	18	24	26	28	36	18
Falsos Negativos	4	3	9	6	11	5
PRECISIÓN %	52.63	36.84	31.58	45.10	29.41	52.63

Tabla 4.14: Comparación de la reclasificación utilizando el HMM con los diferentes experimentos referentes al comportamiento ***Volar***.

Precisión General	Opc 1	Opc 2	Opc 3	Opc 4	Opc 5	Opc 6
Clasificación con KNN %	59.09	56.29	55.84	62.10	59.25	59.68
Clasificación de Observaciones (KNN) %	60.78	58.26	57.50	75.63	59.15	61.34
Reclasificación de observaciones (HMM) %	75.71	68.70	67.40	60.46	70.00	75.94

Tabla 4.15: Comparación de precisiones generales en los difentes experimentos.

Capítulo 5

Conclusiones

El problema de determinar los comportamientos de la mosca de la fruta radica en la determinación de sus movimientos y posiciones que toma en cada imagen de la secuencia de imágenes, por ello fue necesario primeramente obtener estas posiciones con el objeto de realizar un cálculo con estas posiciones y así determinar la velocidad de movimiento de la mosca y de esta forma deducir el posible comportamiento que estaba teniendo. Partes de la secuencia es difícil determinar su comportamiento debido a que se observa que a mosca no tiene un determinado comportamiento, es decir, camina alrededor de su propio eje pero no hacia una determinada dirección lo que causa error en el sistema. Este error es determinado por la constante variación del centroide en esta región. El algoritmo de KNN incurre en este tipo de errores frecuentemente, sin embargo utilizando el algoritmo de Viterbi y el modelo oculto de Markov es posible corregir varias de estos errores y de esta forma elevar la precisión del sistema.

El algoritmo de KNN no es suficiente para resolver la problemática de este proyecto, sin embargo si se crea un modelo matemático como un modelo oculto de Markov se puede aumentar la precisión de clasificación de los comportamientos. Por otro lado, la característica de velocidad que determina el comportamiento no es suficiente para poder resolver una problemática de este tipo. Es necesario tener más características en el algoritmo de KNN que determinen los comportamientos con un error mínimo.

Los modelos ocultos de Markov son una buena solución para obtener más información de un experimento basado en probabilidades de transición de estados, ya que de esta manera se pueden deducir algunas características esenciales del proyecto, por ejemplo, en este proyecto de tesis la probabilidad para que la mosca del mediterráneo pase de un estado Volar a un estado Descansar es menor que la probabilidad que existe cuando la mosca pasa del estado Volar a Caminar.

De manera general se concluye que teniendo una representación matemática que

modele un problema de la vida real, entonces es posible obtener mejores resultados a la hora de resolver esta problemática. En este caso, el modelo oculto de Markov, representa bien el problema de la clasificación de los comportamientos de la mosca y con ayuda del algoritmo de Viterbi es posible suavizar las observaciones obtenidas por el algoritmo de KNN.

5.1. Trabajo Futuro

Para aumentar la precisión del sistema será necesario aumentar las características. El primer trabajo a futuro será aumentar las características para la clasificación del algoritmo de KNN, estas serán características contextuales como por ejemplo el sexo de la mosca no la edad, es bien conocido que una mosca joven es más activa que una mosca vieja.

Otro trabajo a futuro será maximizar los parámetros del modelo oculto de Markov para obtener una reclasificación con mayor precisión. En esta parte se utilizará el algoritmo de Baum Welch descrito en el capitulo 2 en la sección de ¿Cómo ajustar los parámetros del modelo $\lambda = (A, B, \Pi)$ para maximizar $P(O|\lambda)$?. Con esta obtención de parámetros con mayor probabilidad se tendrá una estimación mayor para esta reclasificación.

Por otro lado se construirá una interfaz para mostrar los resultados tanto de las observaciones como de la reclasificación de estas. generando las imágenes de inmediato junto con la clasificación automática.

Referencias

[1] Tucker Balch Adam Feldman. Modeling honey bee behavior for recognition using human trainable models. Georgia Institute of Technology, Atlanta, Georgia 30332, USA.

[2] APHIS Protección de Plantas y Cuarentenas. La mosca del mediterráneo. Departamento de Agricultura de los Estados Unidos Deciembre 2002.

[3] Manuela Veloso Tucker Balch, Zia Khan. Automatically tracking and analyzing the behavior of live insect colonies. Montréal Quebec, Canada, 1 Junio 2001.

[4] Nuria Oliver Alex P. Pentland. Graphical models for driver behavior recognition in a smartcar. Massachusetts Institute of Technology.

[5] Nam T. Nguyen Svetha Venkatesh Geoff West Hung H. Bui. Hierarchical monitoring of people's behaviors in complex environments using multiple cameras. School of Computing, Curtin University of Technology GPO Box U1987 Perth, 6845 Western Australia 2002.

[6] IEEE Lawrence R. Rabiner, Fellow. A tutorial on hidden markov models and selected applications in speech recognition. IEEE, Vol 77, No. 2, February 1989.

[7] Valeriy A. Naumov Gely P. Basharin, Amy N. Langville. The life and work of a.a. markov. 30 December 2003.

[8] J.A. Eagon L.E. Baum. An inequality with applications to statistical estimation for probabilistic functions of markov processes and to amodel for ecology. Bulletin of the American Mathematical Society, May 1967, Volume 73, pp. 360-363.

[9] Dani Goldberg. Evaluating the dynamics of mobile robotic systems. October 2000.