



Universidad
Rey Juan Carlos

ESCUELA SUPERIOR DE INGENIERÍA INFORMÁTICA

INGENIERÍA TÉCNICA EN INFORMÁTICA DE SISTEMAS

Curso Académico 2011 / 2012

Proyecto Fin de Carrera

**SISTEMA BIO - INSPIRADO BASADO EN AER
APLICADO A AUTOMOCIÓN**

Autor: Manuel González Blanco.

Tutora: Cristina Conde Vilda.



Agradecimientos

Me gustaría dedicar y agradecer este proyecto fin de carrera a todas las personas que me han apoyado y animado, ya que significa el fin de una travesía dura y difícil.

Quiero agradecerles a mi familia, en especial a mis padres, mi abuela y hermano, porque sin ellos, sin su comprensión y sobre todo, su apoyo moral, este objetivo de mi vida que desde pequeño he querido realizar, no se hubiera cumplido jamás. Gracias.

Agradecer a Lorena por todos estos años de apoyo, ya que vio tanto el inicio de este viaje como su finalización. Compartiendo muchas preocupaciones, como muchas alegrías y satisfacciones que esta carrera me ha dado.

Gracias a mi tutora Cristina, por brindarme esta oportunidad y plena confianza, como también su tiempo, dedicación y paciencia.

Gracias a todo el grupo de investigación FRAV, a Oscar, Daniela, Enrique, Eduardo, Isaac y Ángel, por todo el apoyo, ayuda y el tiempo compartido con ellos, creando un ambiente muy especial, haciéndome sentir uno más.

Gracias a todos por haberme ayudado a realizar este sueño.





Resumen

En este proyecto se ha trabajado en el diseño e implementación de un sistema bio-inspirado basado en eventos y su adaptación a cámaras comerciales, evitando unos de los principales problemas de este tipo de sistemas.

Por otro lado, se ha aplicado el sistema a un entorno de automoción mediante la utilización de un simulador altamente inmersivo. De este modo se ha evaluado el rendimiento y adecuación de los sistemas basados en AER para la medición de la actividad de las manos de un conductor.

Hasta ahora solo estaba disponible el estudio basado en eventos AER disponiendo de una cámara prototipo especial fotosensible llamada Retina (*capta los cambios de luminosidad producidos por o sobre el objeto*) con un coste elevado y no comercializada, con un hardware específico requerido.

Gracias a este software implementado, se puede realizar un estudio detallado con videos grabados con cámaras comerciales, ya que este sistema emula el funcionamiento de la cámara Retina.

Este proyecto ha sido desarrollado para un entorno real, en específico, en un contexto de automoción, donde se pretende desarrollar un demostrador barato y compacto, capaz de realizar determinadas tareas de reconocimiento a alta velocidad para el proyecto **VULCANO** (*Ref: TEC2009-10639-C04-04*).

Su entidad financiadora es el **Ministerio de ciencia e Innovación** y han participado las siguientes entidades: *Universidad Rey Juan Carlos, Universidad de Cartagena, Universidad de Sevilla e Instituto Microelectrónica de Sevilla (CSIC)*.

A lo largo del proyecto se explica que mecanismos y técnicas se han llevado a cabo para cumplir el objetivo, desarrollar un sistema bio-inspirado que imita la visión humana y demostrar que este sistema basado en eventos AER, ofrece mejoras respecto al sistema tradicional basado en fotogramas.



**PROYECTO VULCANO: VISIÓN ULTRA-RÁPIDA POR EVENTOS Y SIN FOTOGRAMAS.
APLICACIÓN A AUTOMOCIÓN Y ROBÓTICA COGNITIVA.**

Referencia **TEC2009-10639-C04-04**

Fig. 1. Proyecto VULCANO



Índice General

AGRADECIMIENTOS	3
RESUMEN	5
ÍNDICE GENERAL	6
ÍNDICE DE FIGURAS	7
ÍNDICE DE TABLAS	9
1 INTRODUCCIÓN	10
1.1 REPRESENTACIÓN VISUAL BASADA EN FOTOGRAMAS	10
1.2 REPRESENTACIÓN VISUAL BASADA EN EVENTOS	11
1.3 VENTAJA DE UN SISTEMA BASADO EN EVENTOS	11
2 OBJETIVOS	12
3 ESTADO DEL ARTE	13
3.1 SISTEMA NERVIOSO: NEURONAS.	13
3.2 SISTEMAS ARTIFICIALES NEURO-INSPIRADOS	14
3.3 SISTEMA DE EVENTOS AER.....	14
3.4 ARQUITECTURA HARDWARE AER.....	17
3.5 ARQUITECTURA SOFTWARE AER.....	18
4 ENTORNO DE DESARROLLO	20
4.1 ESTRUCTURA DE LA APLICACIÓN	21
4.2 REQUISITOS.....	23
5 EXTRACCIÓN DE FRAMES Y SELECTOR DE REGIONES DE INTERÉS (ROI)	24
6 TÉCNICAS DE EXTRACCIÓN DE MOVIMIENTO	26
6.1 MEDIA (MEAN) Y MODA (MODE).....	26
6.2 RESTA DE FRAMES (FRAME DIFFERENCE)	27
6.3 REDUCCIÓN DE RUIDO Y REDUNDANCIA.....	27
7 MÉTODOS DE CODIFICACIÓN EN EVENTOS AER	29
7.1 INTRODUCCIÓN	29
7.2 TIMESTAMP	31
7.3 MÉTODOS DE GENERACIÓN DE EVENTOS	32
7.3.1 Método Scan	32
7.3.2 Método Uniform	32
7.3.3 Método Random	33
7.3.3.1 Random LFSR	33
7.3.3.2 Random Square	34
7.3.4 Método Exhaustive	35
8 ARQUITECTURA DEL SISTEMA: SINÁPTICA Y NEURONAL	37
8.1 ARQUITECTURA SINÁPTICA.....	37
8.2 ARQUITECTURA NEURONAL.....	38



9	RESULTADOS	- 39 -
9.1	CONSIDERACIONES PREVIAS	- 39 -
9.2	GENERADOR DE ESTADÍSTICAS	- 40 -
9.3	EVALUACIÓN DE ALGORITMOS	- 41 -
9.4	TABLAS DE RESULTADOS Y GRÁFICAS	- 43 -
9.4.1	Resultados obtenidos de escalar a 4 eventos AER por píxel	- 45 -
9.4.2	Resultados obtenidos de los 4 primeros eventos AER por píxel	- 50 -
10	CONCLUSIONES	- 55 -
	BIBLIOGRAFÍA	- 56 -
	ANEXO A	- 58 -

Índice de Figuras

Fig. 1.	Proyecto VULCANO	- 5 -
Fig. 2.	Descripción del Procesamiento de imágenes basado en fotogramas	- 10 -
Fig. 3.	Comparación ilustrativa entre sensado y procesamiento de imagen	- 11 -
Fig. 4.	Estructura de la Neurona	- 13 -
Fig. 5.	Esquema de representación eventos AER	- 15 -
Fig. 6.	Imágenes creadas colectando eventos de una Retina AER	- 15 -
Fig. 7.	Estructura multicapas de la corteza cerebral (Campos Proyectivos)	- 16 -
Fig. 8.	Esquema hardware del proceso AER	- 17 -
Fig. 9.	Izq: Ventana inicial JAER. Der: Reproducción de un fichero .aerdat aplicando dos filtros	- 19 -
Fig. 10.	Diagrama de flujo del proceso de la aplicación	- 22 -
Fig. 11.	Selector de ROIs	- 24 -
Fig. 12.	Fichero de salida del selector, con las posiciones de las ROIs	- 24 -
Fig. 13.	Técnica <i>background subtraction</i> mediante la Moda	- 26 -
Fig. 14.	Imagen resultado con ruido	- 28 -
Fig. 15.	Imagen resultado sin ruido	- 28 -
Fig. 16.	Módulo de escalado y umbralizado de la aplicación	- 28 -
Fig. 17.	Cadena completa de sensado, generación AER, transmisión por bus AER y recepción AER	- 29 -
Fig. 18.	Estructura LFSR + Contador 2 bits para generación pseudo-random de posiciones	- 34 -
Fig. 19.	Estructura LFSR-8 + LFSR-14 para generación pseudo-random de posiciones	- 35 -
Fig. 20.	Ejemplo de distribución de eventos del método Exhaustive	- 36 -
Fig. 21.	Distribución en <i>slices</i> del método Exhaustive	- 36 -



Fig. 22. Diseño de la arquitectura Sináptica	- 37 -
Fig. 23. Diseño de la arquitectura Neuronal.....	- 38 -
Fig. 24. Módulo encargado de las estadísticas	- 40 -
Fig. 25. Imagen ejemplo para codificar	- 41 -
Fig. 26. Fotograma utilizado en los resultados.....	- 43 -
Fig. 27. ROIs definidas para el resultado	- 44 -
Fig. 28. Eval. de la emisión y detección de la Act. Manos. Sináp. y FD.....	- 45 -
Fig. 29. Eval. de emisión y detección de la Act. Manos. Sináp. y MEAN.....	- 45 -
Fig. 30. Eval. de emisión y detección de la Act. Manos. Sináp. y MODE	- 45 -
Fig. 31. Eval. de la emisión y detección de la Act. Manos. Neuro. y FD	- 46 -
Fig. 32. Eval. de emisión y detección de la Act. Manos. Neuro. y MEAN	- 46 -
Fig. 33. Eval. de emisión y detección de la Act. Manos. Neuro. y MODE.....	- 46 -
Fig. 34. Evaluación y comparativa de tamaño frente al JPG. Sináptica y FD	- 48 -
Fig. 35. Evaluación y comparativa de tamaño frente al JPG. Neuronal y FD.....	- 48 -
Fig. 36. Evaluación y comparativa de tamaño frente al JPG. Sináptica y MEAN	- 48 -
Fig. 37. Evaluación y comparativa de tamaño frente al JPG. Neuronal y MEAN.....	- 48 -
Fig. 38. Evaluación y comparativa de tamaño frente al JPG. Sináptica y MODE.....	- 48 -
Fig. 39. Evaluación y comparativa de tamaño frente al JPG. Neuronal y MODE.....	- 48 -
Fig. 40. Eval. de la emisión y detección de la Act. Manos. Sináp. y FD.....	- 50 -
Fig. 41. Eval. de emisión y detección de la Act. Manos. Sináp. y MEAN	- 50 -
Fig. 42. Eval. de emisión y detección de la Act. Manos. Sináp. y MODE	- 50 -
Fig. 43. Eval. de la emisión y detección de la Act. Manos. Neuro. y FD	- 51 -
Fig. 44. Eval. de emisión y detección de la Act. Manos. Neuro. y MODE.....	- 51 -
Fig. 45. Eval. de emisión y detección de la Act. Manos. Neuro. y MODE.....	- 51 -
Fig. 46. Evaluación y comparativa de tamaño frente al JPG. Sináptica y FD	- 53 -
Fig. 47. Evaluación y comparativa de tamaño frente al JPG. Neuronal y FD.....	- 53 -
Fig. 48. Evaluación y comparativa de tamaño frente al JPG. Sináptica y MEAN	- 53 -
Fig. 49. Evaluación y comparativa de tamaño frente al JPG. Neuronal y MEAN.....	- 53 -
Fig. 50. Evaluación y comparativa de tamaño frente al JPG. Sináptica y MODE.....	- 53 -
Fig. 51. Evaluación y comparativa de tamaño frente al JPG. Neuronal y MODE.....	- 53 -



Índice de Tablas

Tabla 1. Dirección AER (estándar) emitida por Retina DVS128.....	- 18 -
Tabla 2. Equipos de tests de pruebas.....	- 23 -
Tabla 3. Reconstrucción de eventos AER	- 41 -
Tabla 4. Porcentajes de acierto	- 42 -
Tabla 5. Eval. de la emisión y detección de la Act. Manos. Sináp. y FD	- 45 -
Tabla 6. Eval. de emisión y detección de la Act. Manos. Sináp. y MEAN	- 45 -
Tabla 7. Eval. de emisión y detección de la Act. Manos. Sináp. y MODE	- 45 -
Tabla 8. Eval. de la emisión y detección de la Act. Manos. Neuro. y FD	- 46 -
Tabla 9. Eval. de emisión y detección de la Act. Manos. Neuro. y MEAN	- 46 -
Tabla 10. Eval. de emisión y detección de la Act. Manos. Neuro. y MODE.....	- 46 -
Tabla 11. Evaluación del tamaño de los datos emitidos para detectar el 50% o el 100% de la actividad en la ROI. Sináptico	- 47 -
Tabla 12. Comparativa del sistema propuesto frente al JPG. Sináptico	- 47 -
Tabla 13. Evaluación del tamaño de los datos emitidos para detectar el 50% o el 100% de la actividad en la ROI. Neuronal	- 47 -
Tabla 14. Comparativa del sistema propuesto frente al JPG. Neuronal	- 47 -
Tabla 15. Eval. de la emisión y detección de la Act. Manos. Sináp. y FD	- 50 -
Tabla 16. Eval. de emisión y detección de la Act. Manos. Sináp. Y MEAN	- 50 -
Tabla 17. Eval. de emisión y detección de la Act. Manos. Sináp. y MODE.....	- 50 -
Tabla 18. Eval. de la emisión y detección de la Act. Manos. Neuro. y FD	- 51 -
Tabla 19. Eval. de emisión y detección de la Act. Manos. Neuro. y MODE.....	- 51 -
Tabla 20. Eval. de emisión y detección de la Act. Manos. Neuro. y MODE.....	- 51 -
Tabla 21. Evaluación del tamaño de los datos emitidos para detectar el 50% o el 100% de la actividad en la ROI. Sináptico	- 52 -
Tabla 22. Comparativa del sistema propuesto frente al JPG. Neuronal	- 52 -
Tabla 23. Evaluación del tamaño de los datos emitidos para detectar el 50% o el 100% de la actividad en la ROI. Neuronal	- 52 -
Tabla 24. Comparativa del sistema propuesto frente al JPG. Neuronal	- 52 -

Capítulo 1

1 Introducción

A la hora de hacer procesamiento de visión en tiempo real, los sistemas basados en fotogramas tienen importantes limitaciones. El principal motivo es la propia naturaleza secuencial de sensado y procesado fotograma a fotograma. Si comparamos los sistemas de procesamiento de imágenes con la forma de llevar a cabo las mismas tareas por parte del cerebro humano, podemos sacar como conclusión de que cada neurona individualmente, es mucho más lenta que cualquier ordenador a la hora de hacer operaciones sencillas, pero aun así, el cerebro es mucho más eficiente gracias a su modo de funcionamiento basado en el paralelismo.

1.1 Representación visual basada en Fotogramas

La imagen en el mundo real es continua tanto en el espacio como en el tiempo, así que el primer paso es capturar imágenes estáticas a intervalos regulares. Cada uno de estas imágenes es un *fotograma* o *frame*. De este modo, se capturan fotogramas a intervalos regulares y al reproducirlos todos seguidos producen para el ojo humano la sensación de movimiento. Para ello es fundamental una buena elección del tiempo de muestreo T_s . De forma habitual se usa la frecuencia de fotograma $f_s = \frac{1}{T_s}$, expresada en fotogramas por segundos (fps).

Cuando se trata de utilizar estos fotogramas como entradas de un sistema de procesamiento, nos encontramos una serie de limitaciones [1]. Se pierde la información de cualquier cambio que ocurra entre dos instantes de muestreo, haciendo imposible la aplicación a un sistema que se mueva a una velocidad mayor que el propio tiempo de muestreo. Por otra parte, en el caso de que la imagen no cambie entre dos instantes o pertenezca invariable incluso, el sistema va a capturar y procesar un fotograma completo, aunque no se obtenga ninguna información. Así pues, los sistemas de procesamiento basados en frames no sirven para aplicaciones de alta velocidad e incluso son enormemente ineficientes para aplicaciones de baja velocidad (ya que hay que procesar la imagen completa en cada instante de muestreo).

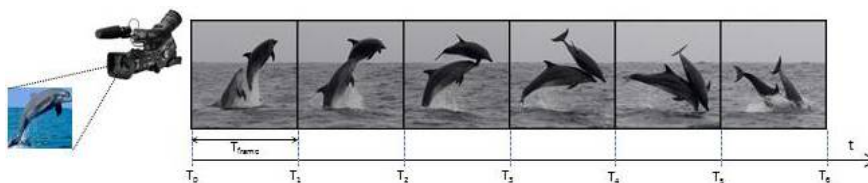


Fig. 2. Descripción del Procesamiento de imágenes basado en fotogramas.

1.2 Representación visual basada en Eventos

Los cerebros biológicos no procesan la visión fotograma a fotograma, sino que la información se codifica en forma de impulsos nerviosos (“*spikes*” o *eventos*). En la retina del ojo, cada célula fotorreptora envía un pulso o evento al córtex cerebral cuando su nivel de actividad alcanza un umbral, de modo que la información se transmite conforme se produce, sin esperar que llegue un instante de muestreo.

Mientras que las representaciones clásicas de imágenes por fotogramas tienen un carácter secuencial, los sistemas biológicos se basan en la ejecución en paralelo de muchas actividades diferentes. Para implementar este paralelismo, cada neurona envía pulsos de información a otras muchas neuronas simultáneamente.

1.3 Ventaja de un sistema basado en Eventos

La gran ventaja inherente a los sistemas de procesamiento basados en eventos se encuentra en el hecho de que la información más relevante se envía en primer lugar. Una posibilidad es que la información se codifique en el orden en el que se producen los eventos, es decir, la neurona que envía un evento en primer lugar es la más activa, lo que significa que su entrada es la más intensa. De este modo, los primeros pulsos enviados codifican el fragmento más importante de la información, lo que implica que se puede hacer un procesamiento aproximado en un intervalo de tiempo realmente pequeño, tomando sólo unos pocos eventos iniciales. Esto es algo que el procesamiento por fotogramas no permite, ya que siempre procesa imágenes completas.

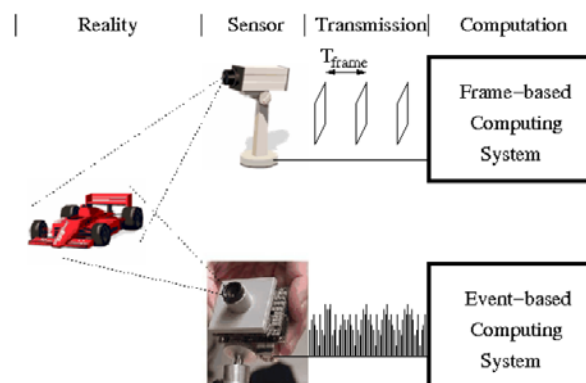


Fig. 3. Comparación ilustrativa entre sensado y procesamiento de imagen basado en fotogramas y basado en eventos.



Capítulo 2

2 Objetivos

Se ha trabajado en este proyecto, el entorno de los sistemas biológicamente inspirados (*imitando la estructura y codificación de información del cerebro*) basados en eventos, comparándolos con los sistemas convencionales basados en fotogramas o frames.

El objetivo principal, es la creación de una herramienta que permita codificar los fotogramas que componen un video digital tomados con cámaras comerciales, en eventos AER ya que hasta ahora, los sistemas AER sólo podían adquirir la información visual mediante sensores específicos denominadas “*Retinas Artificiales*”.

Esta cámara tiene un coste muy elevado y requiere un hardware específico, debido a que es un prototipo no comercializado y se encuentra en el ámbito de investigación.

Se ha diseñado e implementado un módulo “**AER-Imager**” capaz de emular el sistema AER con cámaras comerciales. La principal ventaja, es poder aprovechar las características propias que ofrece el sistema AER, debido al estudio sobre una plataforma software. Esto conlleva a una reducción del coste.

Por otro lado, se ha dado un paso más en la incorporación de la tecnología AER, que hasta ahora estaba centrada en aplicaciones teóricas o de laboratorio, a una aplicación más orientada en el mundo real. Para ello se ha trabajado en el diseño e implementación de un sistema basado en AER en un entorno real de automoción.

Se ha diseñado e implementado un sistema de medición de la actividad de las manos de un conductor, en las distintas zonas de interés definidas por el usuario. Se ha trabajado con conductores profesionales y en un entorno de simulación de una cabina de camión altamente inmersivo.

Además se han propuesto y estudiado dos arquitecturas del sistema: *Sináptica* y *Neuronal* orientadas a un entorno real de automoción, y se ha evaluado su rendimiento, acompañado del tamaño de ocupación en disco de la comprensión habitual de imágenes en formato JPG frente a eventos AER.

Capítulo 3

3 Estado del Arte

3.1 Sistema nervioso: Neuronas.

Las neuronas son un tipo de células del sistema nervioso cuya principal característica es la excitabilidad eléctrica de su membrana plasmática; están especializadas en la recepción de estímulos y conducción del impulso nervioso, entre ellas o con otros tipos celulares.

Las neuronas presentan unas características morfológicas típicas que sustentan sus funciones: un cuerpo celular llamado *soma* o *pericarion*, una o varias prolongaciones cortas que generalmente transmiten impulsos hacia el soma celular, denominadas *dendritas*; y una prolongación larga, denominada *axón* o *cilindroeje*, que conduce los impulsos desde el soma hacia otra neurona u órgano diana.

Las neuronas tienen la capacidad de comunicarse con precisión, rapidez y a larga distancia con otras células, ya sean nerviosas, musculares o glandulares. A través de las neuronas se transmiten señales eléctricas denominadas **impulsos nerviosos**. Estos impulsos nerviosos viajan por toda la neurona comenzando por las dendritas, y pasa por toda la neurona hasta llegar a los botones terminales, que pueden conectar con otra neurona, fibras musculares o glándulas. La conexión entre una neurona y otra se denomina *sinapsis*.

El impulso nervioso se transmite a través de las dendritas y el axón. La velocidad de transmisión del impulso nervioso, depende fundamentalmente de la velocidad de conducción del axón (la cual depende a su vez del diámetro del axón). El axón lleva el impulso a una sola dirección y el impulso es transmitido de un espacio a otro. Las dendritas son las fibras nerviosas de una neurona, que reciben los impulsos provenientes desde otras neuronas. Los espacios entre un axón y una dendrita se denominan espacio sináptico. Las velocidades de conducción pueden alcanzar hasta 120 m/s.

Se estima que cada cerebro humano posee en torno a 10^{11} neuronas, es decir, unos cien mil millones y 10^{15} interconexiones o sinapsis.

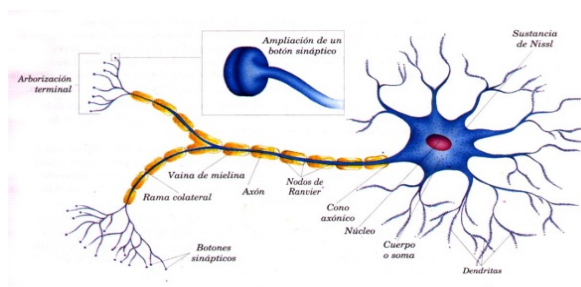


Fig. 4. Estructura de la Neurona



3.2 Sistemas artificiales Neuro –Inspirados

Durante los últimos tiempos, la capacidad de procesamiento de los sistemas informáticos se ha incrementado hasta alcanzar cotas que hasta hace poco resultaban inimaginables. Esto ha permitido diseñar sistemas artificiales capaces de llevar a cabo tareas cada vez más complejas. Sin embargo, en el ámbito de la percepción sensorial ponen de relieve una importante limitación a pesar de trabajar con unidades básicas de procesamiento mucho más rápidas que las neuronas (los sistemas electrónicos actuales tienen tiempos característicos del orden de los nanosegundos, mientras que las neuronas trabajan con milisegundos, lo que supone una diferencia de seis órdenes de magnitud), los sistemas artificiales no consiguen llevar a cabo este tipo de tareas a la misma velocidad que los sistemas biológicos.

El principal motivo de la ventaja de los sistemas biológicos frente a los artificiales está en su arquitectura, masivamente paralela, adaptativa, altamente interconectada y tolerante al ruido del entorno. Los esquemas y métodos de codificar la información sensorial, así como también las técnicas de procesamiento, el cerebro sigue una estrategia de procesamiento completamente diferente.

Los sistemas de visión artificial convencionales están basados en la captura y procesado de secuencias de fotogramas. Así, obtienen una secuencia de imágenes, cada una de las cuales es procesada para extraer algún tipo de característica. Sin embargo, los sistemas biológicos no se basan en el procesamiento de fotogramas, sino en eventos¹.

3.3 Sistema de eventos AER

AER (Address Event Representation) nace en los laboratorios de Caltech en torno a 1991 como un simple protocolo asíncrono de comunicación inter-chip para replicar de manera continua en el tiempo el estado de un array de neuronas de un chip en otro chip [2]. Cada neurona (o píxel) codifica su estado como la frecuencia de una secuencia de impulsos (eventos). Los impulsos generados asincrónicamente por un array de píxeles se arbitran y transmiten a un bus digital inter-chip de alta velocidad [3]. En dicho bus aparece la *dirección* del píxel que originó el evento. En el bus inter-chip AER cada evento asíncrono se transmite en pocos nano segundos (15ns en los chips más rápidos reportados actualmente). Sin embargo, la frecuencia máxima con que un píxel genera eventos es del orden del KHz o inferior. Esto permite multiplexar en el tiempo la actividad de un elevado número de píxeles. Por otro lado, píxeles inactivos o poco activos apenas consumen ancho de banda de comunicación, por lo que solamente se transmite información relevante. Así mismo, los píxeles más activos (más relevantes) envían su información antes.

¹ Se denomina eventos a los impulsos eléctricos generados por la retina y enviados al córtex cerebral cada vez que su nivel de actividad alcanza un determinado umbral (este nivel de actividad se corresponde con la intensidad, contraste, movimiento, etc. de la imagen)

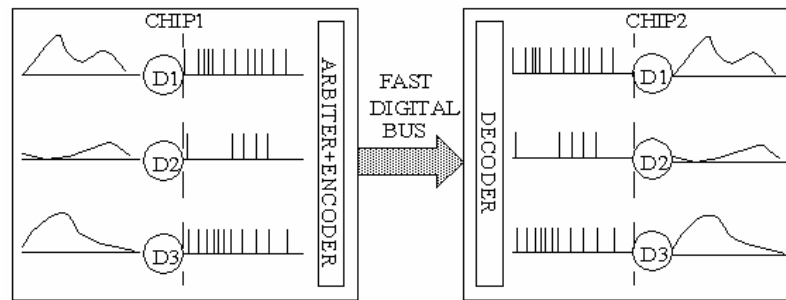


Fig. 5. Esquema de representación eventos AER.

Los primeros trabajos sobre AER, e incluso la mayor parte de los que se reportan actualmente, se centran en sensores AER, tanto visuales, auditivos, como de otras naturalezas. En general, los sensores AER se dotan de algún pre-procesamiento de la información para así concentrarse en información relevante. Por ejemplo, en retinas AER de movimiento cada píxel calcula la derivada temporal de la intensidad de luz sensada. En retinas de contraste cada píxel calcula el contraste espacial instantáneo con los vecinos. Otro aspecto de muy elevado interés en los sistemas de visión AER es el siguiente: la ausencia de fotogramas [4].

Las retinas AER no generan secuencias de imágenes estáticas. En los sensores AER cada píxel es independiente y transmite sus eventos (de movimiento, contraste, luz, etc) cuando los detecta. De esta manera el flujo de información que sale de un chip AER es un continuo en el tiempo de las coordenadas espaciales de los píxeles generadores de eventos. Este aspecto hace a los sensores de visión AER mucho más rápidos que las cámaras convencionales basadas en secuencias de fotogramas, puesto que no hay que esperar a capturar un fotograma completo (típicamente 30-40ms) para disponer de la imagen y empezar a procesarla. Por otro lado, en los sensores AER se elimina una gran cantidad de información redundante que en los sistemas tradicionales de fotogramas se transmite una y otra vez fotograma tras fotograma.

A modo ilustrativo, la Fig. 6 muestra imágenes compuestas a partir del flujo AER de la retina de movimiento. Para crear estas imágenes 2D se colectan los eventos del sensor AER durante un tiempo y se codifica en escala de gris el número de eventos recibidos por píxel. De esta manera se puede visualizar en un sistema de video convencional por fotogramas.

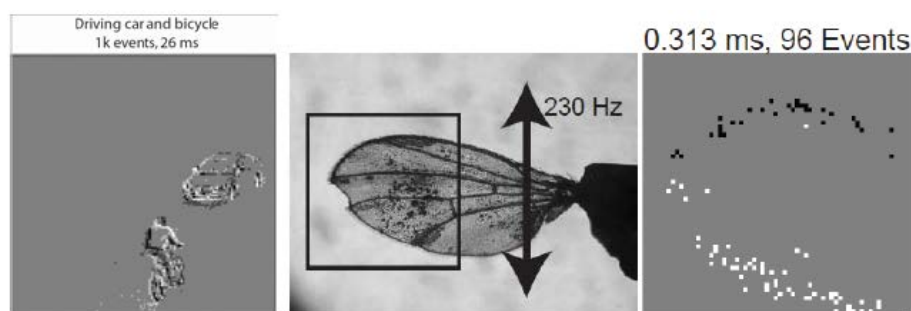


Fig. 6. Imágenes creadas colectando eventos de una Retina AER



Sin embargo, en un sistema de visión AER compuesto de sensor más procesadores neurocorticales, nunca se pasaría a fotogramas. La idea es transmitir y procesar eventos de manera continua en el tiempo. Esto se ilustra en la Fig. 7 la salida de una retina se conecta, mediante “campos proyectivos”, a una secuencia de capas neurocorticales (entre 4 a 10, por ejemplo el cerebro humano tiene unas 10). Los campos proyectivos calculan en realidad convoluciones espaciales. Así, en cada capa se calcula un determinado número de convoluciones espaciales en paralelo. Las salidas de estas convoluciones se combinan en el siguiente nivel y se vuelven a calcular otras. De esta manera en las primeras capas se extraen rasgos simples (bordes, orientaciones, intersecciones, etc), que se combinan en el siguiente nivel para detectar rasgos o formas, que a su vez se combinan en los siguientes niveles hasta detectar objetos concretos independientemente de su tamaño y orientación. Al no calcular las convoluciones por imágenes, sino evento a evento (al igual que en el cerebro), se puede hacer reconocimiento de objetos a muy alta velocidad; si la retina ve un objeto generará una serie de eventos espacial y temporalmente correlacionados, que en el siguiente nivel disparan los segmentos que lo componen, que a su vez disparan en el siguiente las formas que lo componen, hasta que en el último nivel dispara la categoría del objeto. Así, los procesadores AER neurocorticales son chips convolucionadores AER que se interconectan por enlaces AER imitando la estructura por capas de la corteza cerebral.

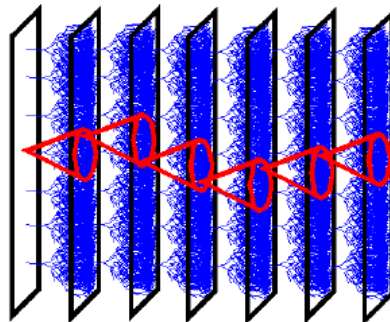


Fig. 7. Estructura multicapas de la corteza cerebral (Campos Proyectivos)

3.4 Arquitectura Hardware AER

La implementación en hardware del sistema bio-inspirado se ilustra en la Fig. 8 donde se muestra un esquema del proceso que sigue. El sistema está compuesto por una cámara **Retina DVS128** cuyo sensor es una matriz de 128x128 píxeles [5], donde emite una dirección AER de 16 bits (pos.X, pos.Y, polaridad) del píxel que ha emitido un impulso². La retina puede estar conectada de forma directa al Pc, pero no es recomendable ya que es USB 1.0 y se pierde gran número de eventos, lo que implica información [6]. Para paliar este problema, se ha creado la **UsbAERmini2** [7], conectada mediante puerto IDE a la cámara y en el que recibe los eventos. Almacena los eventos en una *FIFO*, ya que la Retina emite gran cantidad de eventos a alta velocidad y se produce “el cuello de botella” en el Pc por la naturaleza del protocolo USB. Además cuando el protocolo *handshake* lo permita [8], envía eventos al Pc con una estampa de tiempo o Timestamp, permitiendo al software **jAER** pueda interpretar dichos eventos. Opcionalmente se pueden conectar una o varias placas **UsbAER** (simulan las capas neurocorticales de la corteza cerebral), donde cada una de ellas, hacen un procesamiento de los eventos, generalmente convoluciones (detección de bordes, de objetos con patrones de figuras conocidas, etc).

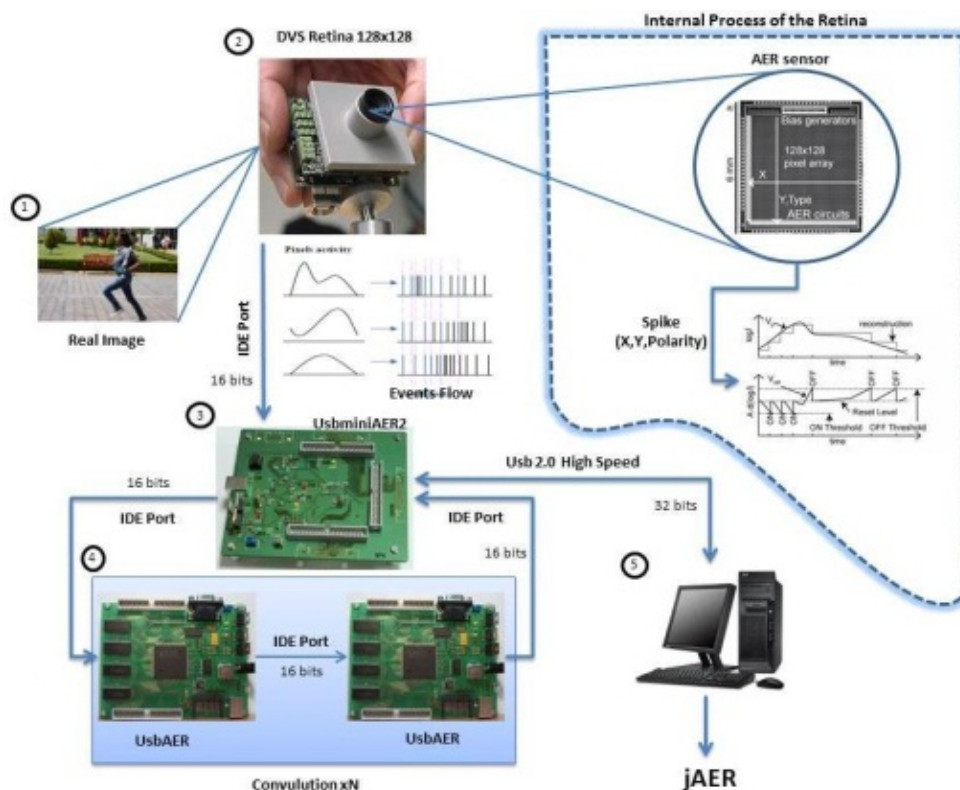


Fig. 8. Esquema hardware del proceso AER

² $|\Delta \log I| > T$ Capta la variación de luminosidad (I) de la escena u objeto. Realiza constantemente el logaritmo de la luminosidad respecto al tiempo. Cuando alcanza un cierto umbral (T) (configurable en las *biases* de la Retina) emite un impulso (positivo o negativo, dependiendo del umbral alcanzado) y resetea el umbral. Por ello desplazando el objeto o variando su luminosidad, la Retina capta dichos cambios, emitiendo impulsos.



3.5 Arquitectura Software AER

jAER es un proyecto libre implementado en Java [9], se compone aproximadamente de 500 clases, y permite la monitorización y procesamiento de eventos a tiempo real en el Pc conectando una interfaz hardware como UsbAERmini2, UsbAERmapper y la retina DVS128 o bien, de forma *offline* con un fichero de eventos previamente grabado en formato binario **.aerdat**, entre otras muchas funciones. Al ser proyecto libre, se encuentra en la forja de código [SourceForge](#).

Ha sido creado por *Institute for Neuroinformatics (INI)* de la Universidad de Zurich (Suiza). Las principales entidades colaboradoras de este proyecto son Universidad de Sevilla y ETH de Zurich entre otras.

La estructura principal del jAER se divide en cuatro grandes bloques:

- **Filtros:** son procesamiento de eventos a tiempo real, la retina envía un paquete de eventos con la dirección del píxel que emitió el impulso. Al aplicar un filtro, captura el paquete que ha llegado, procesa los eventos realizando unas operaciones matemáticas y devuelve el paquete ya procesado. Transcurre todo ello a tiempo real. Una característica importante de los filtros que son muy parametrizables [10]. Hay distintos filtros creados por distintas entidades, los más destacables son *Background Activity Filter* (Elimina el ruido de fondo que captura la retina), *Rotate Filter* (Invierte las posiciones de representación por pantalla), *Rectangular Cluster Tracker* (Seguimiento de objetos), etc.
- **AE Chips:** Hay una gran lista de chips creados por distintas entidades y para distintas finalidades. Lo que diferencia una clase chip de otra, es la forma de tratar y leer los eventos. La clase chip estándar es DVS128 donde el direccionamiento AER es de 16 bits [11]. Los distintos chips leen de distinta forma el direccionamiento, bien por que el problema lo requiere (se necesita ocupar el bit N.C, por ejemplo para retinas en estéreo), el diseño de las placas ha sido modificado, se quiere tratar direcciones en vez de 16 bits a 32 bits, etc.

N.C	POS. Y	POS. X	POLARITY
1 bit	7 bits	7 bits	1 bit

← 16 bits →

Tabla 1. Dirección AER (estándar) emitida por Retina DVS128

- Grabación de eventos: Además de monitorizar los eventos generados por los distintos sistemas basados en AER, puede almacenarlos en un fichero binario .aerdat o.dat. Es interesante esta idea, si se quiere guardar lo que está capturando la retina para posteriormente procesar o reproducir dichos eventos de forma “*offline*”, permitiendo trabajar con el fichero sin disponer de la cámara Retina y todo el sistema hardware.
- Reproductor del fichero .aerdat: Dispone de botones para el control de reproducción (play, fw, rw, pause, ...) y cajas de texto donde se muestra el tiempo en microsegundos y el número de eventos que se está leyendo. Se puede acotar la reproducción por número de eventos, por franja de tiempo (expresado en microsegundos) o bien según se captura, reproduce en vivo, con la opción *real time playback*. Dispone de la opción de grabar la monitorización de los eventos en una secuencia de frames.

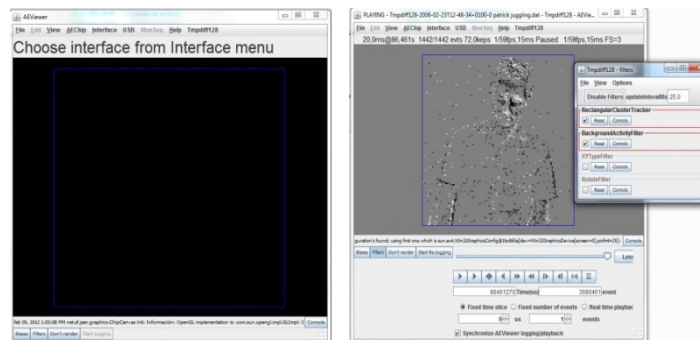


Fig. 9. Izq: Ventana inicial jAER. Der: Reproducción de un fichero .aerdat aplicando dos filtros.



Capítulo 4

4 Entorno de desarrollo

La cámara Retina es una cámara prototipo, es decir, no está comercializada y se encuentra en laboratorios de investigación para su estudio y experimentación.

Es un prototipo de un coste elevado ya que no está comercializada.

Surge por ello la necesidad de poder monitorizar o procesar eventos, a través del jAER, procedentes de una fuente distinta a la cámara Retina por motivos ya citados; por ejemplo, un vídeo o secuencia de imágenes tomadas por una cámara comercial ya que hasta ahora, no se podía operar con dichos eventos AER, dentro de la interfaz jAER si no eran provenientes de la cámara Retina.

Por este motivo, se ha diseñado una aplicación que sea capaz de adaptar cualquier fuente de entrada proveniente de una cámara común, a los requisitos del jAER, con el fin de poder tratar los eventos extraídos de dicha fuente, como sí procedieran de la cámara Retina.

Esta aplicación diseñada es de gran utilidad en un entorno real ya que, se dispone de cámaras convencionales y sustituir estas por cámaras Retina, tendría un coste demasiado elevado por eso, la mejor opción es adaptar las restricciones que impone un entorno real y emular el funcionamiento de la cámara Retina.

La aplicación ha sido diseñada e implementada en el lenguaje de programación **MATLAB** (**MA**trix **LAB**oratory).

Entre sus prestaciones básicas se hallan: la manipulación de matrices, la representación de datos y funciones, la implementación de algoritmos, la creación de interfaces de usuario (GUI) y la comunicación con programas en otros lenguajes y con otros dispositivos hardware.

El paquete Matlab dispone de dos herramientas adicionales que expanden sus prestaciones, *Simulink* (plataforma de simulación multidominio) y *GUIDE* (editor de interfaces de usuario - GUI). Además, se pueden ampliar las capacidades de Matlab con las cajas de herramientas (*toolboxes*); y las de Simulink con los paquetes de bloques (*blocksets*).

Para ejecutar rápidamente cálculos matriciales y vectoriales complejos, utiliza bibliotecas optimizadas para el procesador. Para cálculos escalares de aplicación general, genera instrucciones en código máquina utilizando su tecnología JIT (Just-In-Time).

Por estos motivos se ha seleccionado este lenguaje para desarrollar el proyecto, ya que gran parte de la información que se va a tratar son imágenes, y las imágenes son matrices cuyos valores indican los distintos niveles de intensidad del píxel.



Como el propio nombre del lenguaje indica, Matlab para operaciones con matrices tiene un alto rendimiento.

Se ha utilizado para la implementación de la aplicación funciones que proporciona el toolbox de procesamiento de imagen o *Image Processing Toolbox*, como por ejemplo lectura (*imread*) o escritura (*imwrite*) de imágenes entre muchas otras.

4.1 Estructura de la Aplicación

En los últimos años, se ha puesto de manifiesto el gran potencial de la tecnología **AER** (*Address Event Representation*) para sensor y procesar visión a muy alta velocidad, así como para actuar sobre los subsistemas de percepción y accionamiento en el ámbito de la Neuro-robótica. En visión convencional, una cámara de video capta secuencias de fotogramas, cada uno de los cuales debe ser tratado por sofisticados algoritmos si se quieren realizar tareas procesamiento y reconocimiento automático (*en automoción, robótica, etc*).

AER se basa en un concepto diferente, imitando la estructura y codificación de información del cerebro. En AER cada pixel del sensor emite eventos de información cuando “*capta*” un determinado nivel de alguna propiedad visual (*movimiento, contraste, luminosidad, ...*). De esta manera, la salida del sensor es un flujo continuo de información (espacial y temporal) que no está restringida a fotogramas discretos. Este flujo de información visual se lleva a una estructura jerárquica, que imita la corteza cerebral, y que va extrayendo información relevante de una manera continua y paralela, evento tras evento, sin esperar a fotogramas.

El objetivo principal de este proyecto es la creación de una herramienta que permite codificar los fotogramas que componen un video digital tomados con una cámara comercial en eventos AER, para estudiar la actividad de las manos de un conductor sobre el volante u otras zonas de interés.

Hasta ahora los sistemas AER sólo podían adquirir la información visual mediante sensores específicos llamados “**Retinas Artificiales**”. Esta cámara tiene un coste muy elevado ya que es un prototipo no comercializado y se encuentra en el ámbito de investigación. Dicha cámara requiere de un hardware específico.

El problema surge cuando no se dispone de esta cámara o se dispone de videos ya grabados con una cámara comercial estándar y se quieren codificar a eventos AER para su estudio.

Para llevar a cabo este objetivo se ha diseñado una aplicación dividida en módulos, como se ilustra en la Fig. 10, cada uno desarrolla una tarea específica y en el que a continuación se explicará con más detalle.

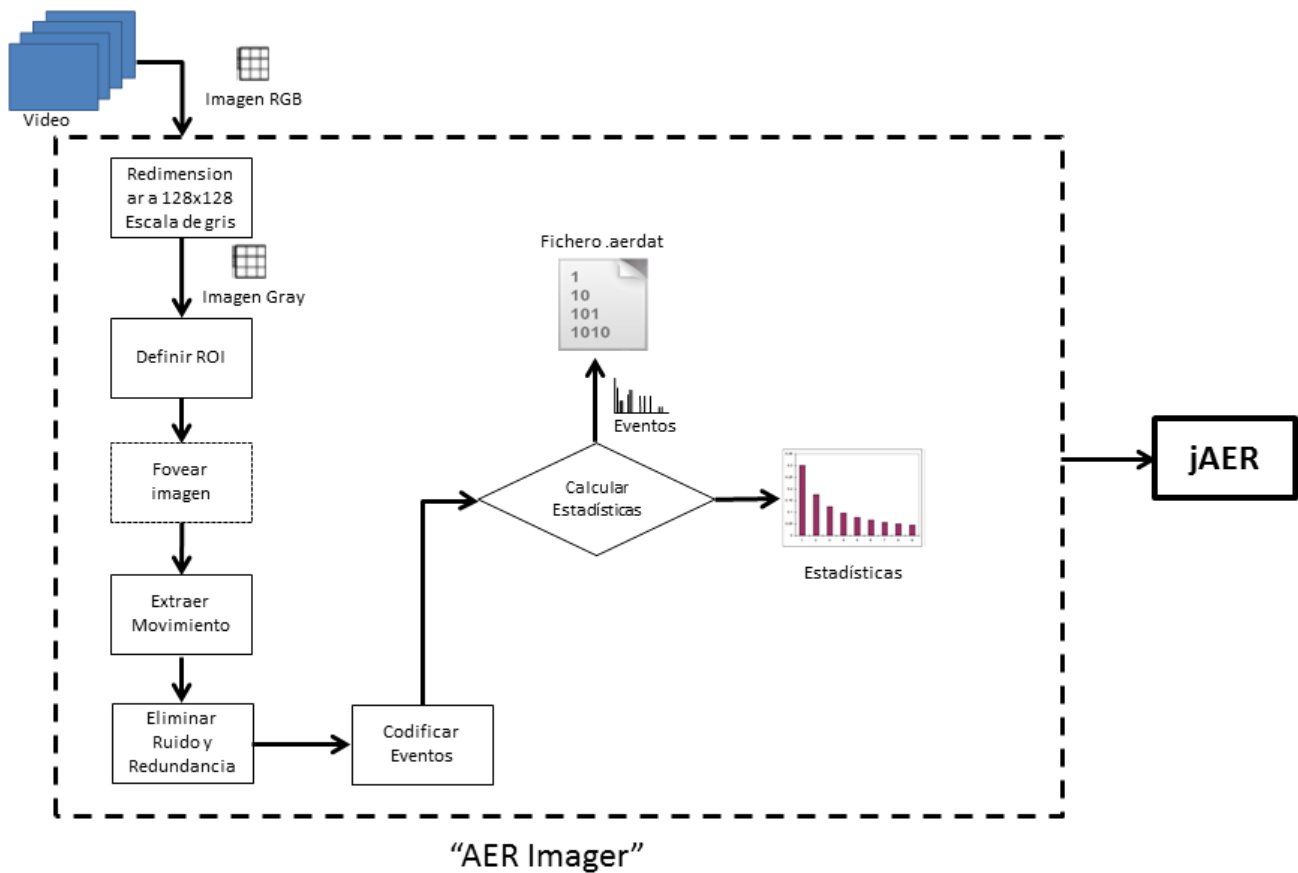


Fig. 10. Diagrama de flujo del proceso de la aplicación

Se presenta la opción de representar la imagen de manera normal o foveada[12]. Esta última se ha desarrollado ya que es una técnica bio-inspirada, y el objetivo de este proyecto es crear un sistema bio-inspirado. Se imita la forma en el que el ser humano visualiza el escenario, centrando toda atención en un punto y obviando el resto de la imagen. Por ello, al fovear una imagen, toda ella aparece difuminada excepto la zona que consideramos importante, quedando dicha zona intacta. A esta zona se le denomina fóvea[13].

El módulo de selección de regiones de interés tiene el objetivo de imitar el funcionamiento neurológico del humano, fijando los focos de atención de la escena. Permite al usuario definir mediante figuras elípticas o rectangulares las zonas del escenario que son consideradas de alto interés, para posteriormente realizar operaciones y estadísticas en dichas zonas.

Además se definen tres técnicas de extracción de movimiento. Se extrae el movimiento del video digital, ya que la información relevante es el cambio sufrido entre fotogramas, el movimiento producido dentro de la escena. Posteriormente se codifica dichos fotogramas a eventos AER mediante varios algoritmos que se detallarán adelante.



La aplicación tiene en cuenta las restricciones que impone el sistema AER, número máximo de eventos AER emitidos por píxel, es decir, como máximo se emiten cuatro eventos de un mismo píxel por el bus AER con el fin de reducir la redundancia y no sobrecargar el bus. Por lo tanto, una dirección de un píxel cuyo valor de intensidad sea elevado, como máximo aparecerá en el vector final de eventos codificados, o bien en el bus AER, cuatro veces dicha dirección.

El módulo de estadísticas es el encargado de la medición de la actividad de las manos del conductor en las zonas de interés.

Todos estos datos están reflejados en tablas Excel generadas de forma automática por la aplicación.

4.2 Requisitos

La aplicación ha sido desarrollada y probada en dos ordenadores con las siguientes características:

	Pc portátil Sony Vaio Fw-21m	Pc Sobremesa Dell
Procesador	Intel Core 2 Duo 2.40GHz	Intel Core i5 2400 3.10GHz
Memoria RAM	4Gb (2 x 2Gb)	4Gb (2 x 2Gb)
Matlab	v7.12.0 R2011a	v7.12.0 R2011a
Sistema Operativo	Windows 7 Ultimate 32 bits	Windows Professional 32 bits

Tabla 2. Equipos de tests de pruebas

Los métodos y datos almacenados durante la ejecución de la aplicación, por tanto son datos que residen en memoria, han sido totalmente optimizados para aprovechar el menor porcentaje de uso de memoria RAM, ajustando la reserva de espacio al menor tipo siempre que fuera posible (Ej. *uint8*).

Se recomienda si se desea transformar videos o una carpeta que contiene una secuencia de imágenes, a un tamaño superior a lo permitido por el software jAER (128x128 px.) o bien, si el número de imágenes o fotogramas que componen el vídeo es elevado (superior a 5000 frames), que se utilice un sistema operativo de 64 bits, ya que la gestión de memoria varía entre las dos versiones, pudiendo obtener más espacio en memoria si la versión es de 64 bits.

El rendimiento en lo que tiempo se refiere no varía de una versión a otra, únicamente en almacenamiento temporal de datos en memoria.

Capítulo 5

5 Extracción de frames y selector de Regiones de Interés (ROI)

Como ya se ha visto, la Retina tiene unas limitaciones hardware implícitas, como por ejemplo el sensor de tamaño 128x128 px. El software jAER tiene en cuenta estas limitaciones a la hora de monitorizar los eventos llegados de la cámara o bien grabados previamente por ello, la aplicación diseñada también debe tener en cuenta estos detalles y se ha de tratar la entrada del sistema para que cumpla con estos requisitos.

La entrada de la aplicación será un vídeo digital formado por una secuencia de frames o imágenes estáticas en color (RGB) o en blanco y negro (escala de grises), con una resolución que varía según el vídeo que se tome como entrada (comúnmente 480x640 píxeles).

La idea es transformar los frames en eventos AER para ello, habrá que extraer los fotogramas que componen el video y transformar cada uno en escala de grises, además de redimensionar el tamaño del frame a 128x128 en caso de que fuera necesario.

El objetivo principal por el que se ha diseñado y utilizado un seleccionador de regiones de interés es imitar la visión que realizada el ojo humano. Nuestra visión se centra en la parte que consideramos importante de todo el escenario que visualizamos, obviando el resto. En un entorno *bio-inspirado* esto se traduce en **regiones de interés**, por lo que seleccionando *zonas* dentro de la imagen que consideramos importante según el contexto, en este proyecto a automoción, imitamos el funcionamiento biológico del ojo humano.

Como se ilustra en la Fig. 11 se dispone de un programa desarrollado en C++, para que el usuario pueda seleccionar la o las regiones de interés (ROIs) dónde se estudiará la actividad en dicha zona.

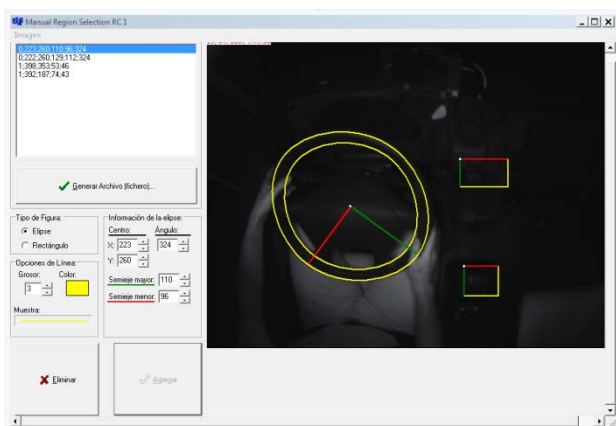


Fig. 11. Selector de ROIs

4
0;223;260;110;96;324
0;222;260;129;112;324
1;398;353;46;53;0
1;392;187;43;74;0

Fig. 12. Fichero de salida del selector, con las posiciones de las ROIs



Este programa al iniciarse carga el primer fotograma del video (dispone de la opción de cargar cualquier otra imagen) y permite seleccionar mediante el trazado de elipses o cuadrados, la región o regiones que se pretenden estudiar. Posteriormente genera un fichero de texto Fig. 12 que contiene el número de regiones descritas y las posiciones en píxeles.

El propósito principal es estudiar la actividad de movimiento que se concentra en la región seleccionada, para posteriormente a la hora de realizar las estadísticas, la aplicación muestre dicha actividad en forma de tablas y gráficas.

Capítulo 6

6 Técnicas de extracción de movimiento

Previamente se ha explicado el funcionamiento que sigue la cámara Retina, captura la información relevante de la escena u objeto que se está grabando, es decir, obtiene los cambios de luminosidad y esto nos produce la sensación de que captura el “movimiento”. Por ello necesitamos recurrir a estas técnicas para simplificar el problema y así poder emular la captura que realiza la Retina.

El objetivo es crear un conjunto de frames que contienen únicamente los objetos móviles. Se ha elegido la resta de fondo como técnica de procesamiento de imágenes. Esta técnica es óptima en grabaciones con fondo invariable como es nuestro caso, es decir, con una cámara fija siempre en el mismo punto, y es la mejor opción para aplicar en un sitio con mucho movimiento. En caso de que la cámara fuera móvil, se debería realizar un procesado más complejo de la escena.

El algoritmo de la resta de fondo se basa en la obtención de una imagen que representa el fondo de la escena a partir de N frames de un vídeo. Se calcula la media o moda de los píxeles de los N frames con el fin de dejar en la imagen de fondo únicamente los objetos que no se han movido en la secuencia de vídeo. Posteriormente se efectúa una resta píxel a píxel entre el frame actual y la imagen de fondo para obtener los objetos móviles.

6.1 Media (Mean) y Moda (Mode)

La imagen de fondo debe ser actual, por tanto su cálculo debe ser constante [14]. La realización de un cálculo una única vez no ofrecería buenos resultados y por lo tanto debe ser calculada dinámicamente. La imagen de fondo se recalculará completamente y deberá tener una transición suave, es decir, cada cierto tiempo se irán incluyendo nuevos frames (llamado también *intervalo* de actualización del *buffer*). La implementación del buffer que contiene el conjunto de imágenes para realizar la imagen fondo es de tipo FIFO (*first in first out*), de este modo se mantendrá un estado actual de la imagen de fondo, haciendo frente a cambios en la iluminación o posibles cambios en el escenario de grabación. El proceso se refleja en la Fig. 13



Fig. 13. Técnica *background subtraction* mediante la Moda



El proceso para las dos técnicas es el mismo, la diferencia que presentan es la operación que realizan para obtener dicho fondo, y ventajas e inconvenientes de las implementaciones [15].

- **Media:** Su complejidad es $O(n)$, su resultado incrementa cuantas más imágenes contiene el buffer. La ventaja es el tiempo de cálculo, es muy corto al tratar las imágenes como matrices. El inconveniente es que la imagen fondo muestra una estela tras los objetos móviles. Esto es debido a que el color de un píxel viene determinado por la media de todos los píxeles que han ocupado esa posición durante la toma.
- **Moda:** Su complejidad es $O(n^2)$, sus resultados son buenos con un número pequeño de imágenes. La ventaja es que los objetos móviles no dejan rastro en la imagen de fondo. El inconveniente el tiempo de cálculo es elevado.

6.2 Resta de Frames (Frame Difference)

El cálculo para la obtención de la imagen fondo es sencillo. Se obtiene restando píxel a píxel la imagen actual con la imagen previa (si el intervalo de actualización es igual a uno). El resultado obtenido es la diferencia entre un frame y otro, es decir, el movimiento que se ha producido en el intervalo de tiempo transcurrido entre un frame y el siguiente. La ventaja que presenta esta técnica es que el tiempo de cálculo es muy reducido, ya que son resta de matrices. Además si el objeto se ha desplazado en un corto espacio respecto al tiempo, recoge dicho movimiento. Es una ventaja pero también implica un inconveniente, ya que esta técnica obtiene el más mínimo movimiento producido (por el objeto, vibración de la cámara, etc) y esto se traduce en ruido dentro de la imagen resultante [14].

6.3 Reducción de Ruido y Redundancia.

Al aplicar las técnicas citadas, la imagen resultado puede contener “ruido” o falso movimiento como se muestra en la Fig. 14, es decir, los píxeles obtienen valores superiores a 0 (equivale al color negro en escala de grises) indicando que en esa zona o conjunto de píxeles se ha detectado movimiento cuando en realidad no lo hubo. Este problema es debido a cambios de iluminación en las escenas, posibles cambios en la posición o zoom de la cámara, vibración de la cámara, etc. La solución empleada es el Umbralizado, todos los píxeles que no superen cierto umbral (indicado como parámetro) tomarán el valor de 0 o negro, dando lugar a que todos los píxeles cuyo valor supere este umbral formen parte del objeto.

Se ha implementado un *umbralizado manual*, indicando el umbral deseado y un *umbralizado dinámico*, la dificultad inherente en este método es elegir un umbral que se adapte de manera óptima a la imagen.



En el umbralizado dinámico se crea un histograma de las intensidades del píxel de la imagen y utiliza el punto de valle como el umbral. El acercamiento del histograma asume que hay cierto valor medio para los píxeles de fondo y del objeto, pero que los valores reales del píxel tienen cierta variación alrededor de estos valores medios. Sin embargo, esto puede ser de cómputo costoso, y los histogramas de la imagen pueden no tener puntos bien definidos del valle.



Fig. 14. Imagen resultado con ruido Fig. 15. Imagen resultado sin ruido

Una de las características destacables e importantes de la cámara Retina es que no envía información de forma *redundante*, es decir, no envía más de 4 eventos de media en el mismo intervalo de tiempo, del píxel que ha sufrido un cambio de luminosidad. Esto hace que el AER sea tan rápido y efectivo. A la hora de imitar este comportamiento en nuestro sistema, se ha diseñado dos mecanismos.

- **Escalado de la imagen:** La imagen movimiento o resultado tiene unos valores de intensidad de 0 a 255, ya que está en escala de grises. Este mecanismo modifica dichas intensidades a un valor entre 0 y la cota o máximo, en este caso a 4 (ya que la Retina envía máx. 4 evts.) de manera proporcional, obteniendo así una imagen cuyo nivel de intensidad del píxel más alto será 4 y el más bajo 0.
- **Umbralizado de la imagen:** Consiste en obtener los n primeros niveles de intensidad de la imagen, donde n será 4 como ya se citó, aunque este parámetro es ajustable desde la aplicación

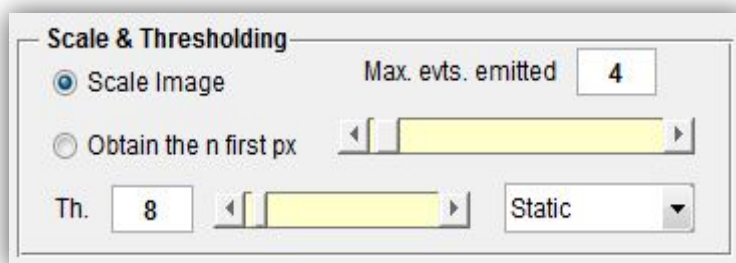


Fig. 16. Módulo de escalado y umbralizado de la aplicación

Capítulo 7

7 Métodos de codificación en eventos AER

7.1 Introducción

Uno de los propósitos de este trabajo se centra en la creación de una herramienta que permita la transformación de video digital (o cualquier otro tipo de información digital) en formato AER a partir de imágenes estáticas y que permita la transmisión de los eventos AER asociados por el bus AER, así como la recepción de eventos AER y su interpretación vía software.

Un sistema AER de visión, como ya se explicó, transforma la luminosidad de los píxeles en secuencia de eventos. Esta secuencia de eventos se transmite en forma de secuencia de direcciones por el bus AER y en la recepción se vuelven a transformar en secuencias de eventos que son integrados por cada píxel.

Surge la necesidad de conectar los sistemas AER a un Computador. Para resolver esta conexión se necesita, por una parte pasar imágenes digitales a formato AER, y por otra obtener imágenes digitales de un bus AER. Ambas transformaciones no son equivalentes en complejidad.

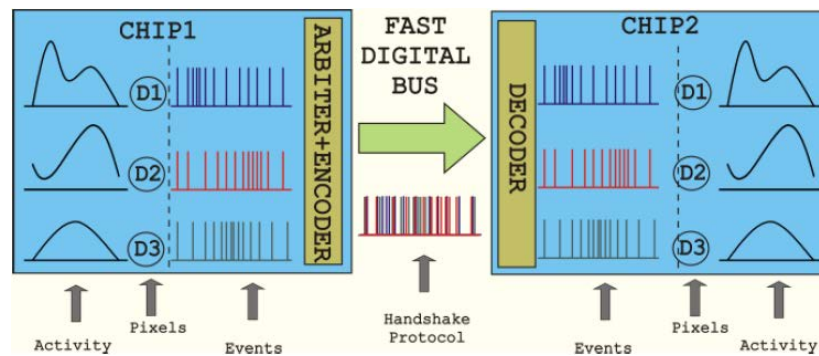


Fig. 17. Cadena completa de sensado, generación AER, transmisión por bus AER y recepción AER.

El paso de bus AER a imágenes digitales consiste simplemente en contar para cada píxel el número de veces que aparece su dirección en el bus durante un período mayor o igual que el T_{max} . Si se integra durante un período mayor obtendríamos lo equivalente en fotografía a un mayor “tiempo de exposición”, donde los objetos con movimientos rápidos aparecerían movidos. Si se integrase durante un período menor, disminuiríamos el tiempo de exposición y bajaría la luminosidad de los píxeles. De hecho los píxeles con el mínimo número de luminosidad no se verían reflejados en algunos frames. El ajuste en el tiempo de integración no es crítico ya que la imagen no será posteriormente tratada, sólo se pretende visualizarla.



El objetivo de este capítulo consiste en proponer diferentes métodos de transformación de imágenes estáticas al formato AER, siguiendo ciertas reglas para minimizar los errores, así como que el tráfico total de eventos resultante en el bus AER esté repartido.

El algoritmo de conversión se basa en asignar una dirección a cada píxel y repetir en el bus AER eventos de la dirección de cada píxel tantas veces como el valor del píxel. Existen múltiples formas de hacer esta conversión siempre que se respete la frecuencia de aparición de la dirección de cada píxel, es decir, que la localización de cada evento de un píxel no tiene importancia, sino que lo importante es la separación entre ellos. Por este motivo, ocurre que el tiempo empleado para convertir imágenes en formato AER depende de la información de la imagen, ya que una imagen con todos los píxeles al valor máximo tardará el máximo tiempo en convertir los eventos de cada píxel, pero una imagen poco iluminada (valores bajos de píxel) tardará muy poco tiempo en convertirse.

Supongamos ‘*vector de eventos*’ un vector de direcciones de píxeles que representa la distribución en el tiempo de los eventos tal y como se producirán en el bus AER una vez generado. En este vector se va a almacenar una imagen en formato AER, por lo que su tamaño $N_{max} = Ancho_Imagen \cdot Alto_Imagen \cdot K$ (máximo nivel de intensidad del píxel).

La forma de distribuir los eventos dentro del vector nos determina varios algoritmos de generación de Address-Event de una imagen. El número de eventos asociado a una imagen será como máximo N_{max} , por tanto, una imagen no siempre llenará el vector. A las posiciones sin direcciones AER se les denomina *huecos*.

El vector debe ser transmitido por el bus AER en un tiempo T_{frame} . Cada evento AER deberá ser transmitido en T_{pulso} . Un *hueco* supone una pausa en el bus AER de T_{pulso} . Y T_{pulso} se obtiene según la siguiente ecuación: $T_{pulso} = \frac{N_{max}}{T_{frame}}$



7.2 Timestamp

El *timestamp* es una marca o estampa de tiempo (expresado en microsegundos μs) que se le añade a cada evento que se ha emitido. Esta marca no cobra importancia a la hora de procesar eventos, pero sí a la hora de monitorizarlos. Surge la necesidad de tener una referencia para poder representar y visualizar los eventos vía software, y de esta manera, marcando a cada evento una estampa de tiempo, el software en este caso el jAER, puede representar los eventos que recibe por el bus en el instante aproximado³ en el que fue emitido, pudiendo monitorizar los eventos a tiempo real [16].

El timestamp es añadido vía hardware por la placa UsbAERmini2 [17], con un tamaño de 32 bits, es decir, puede representar $2^{32} = 4294967296 \mu s = 1,20h$.

El problema surge cuando no se dispone de la cámara Retina ni el hardware necesario, disponiendo sólo de un video digital formado por una secuencia de fotogramas o imágenes estáticas, en contraposición a las imágenes continuas que se representan en los sistemas AER. Una solución que se ha aplicado a esta problemática, es simular el estampado de tiempo que lleva a cabo el hardware mediante el *frame rate* del video, es decir, los fps (*frames per seconds*). Dado los fps del video podemos calcular gracias a la ecuación: $T_{frame} = \frac{1}{fps}$, que tiempo tarda en procesar un fotograma o frame.

Por ejemplo, el periodo de muestreo para un video digital convencional será el correspondiente a 25 imágenes por segundo (*fps*), es decir, $T_{frame} = 40$ ms.

Llegado a esto, podemos macar los eventos generados con el tiempo correspondiente de dividir el T_{frame} entre el número de eventos totales que se han generado, es decir, dividir T_{frame} entre N_{max} . Por lo tanto $Timestamp = \frac{T_{frame} \cdot 10^6}{N_{max}}$ (timestamp (μs) / evento).

Suponemos que tenemos un vector de eventos con $N_{max} = 800$, y el video digital tiene un frame rate de 25 fps. Por lo que a cada evento le corresponde un timestamp de 50 μs . Esta operación es iterativa a cada frame o fotograma que compone el video digital.

³ Con un pequeño margen de error debido a las latencias de comunicación de la placa UsbAERmini2 y el Pc mediante Usb. Los eventos son almacenados en una memoria FIFO de la placa, y se va etiquetando según se envía por el Usb 2.0 cuando el protocolo lo permita. Es necesaria la memoria ya que genera muchos más eventos la Retina de lo que permite transmitir el Usb 2.0.



7.3 Métodos de generación de eventos

7.3.1 Método Scan

Este primer método es el más sencillo. Consiste en recorrer la imagen k veces, siendo k el valor máximo de intensidad asociado a un píxel de la imagen de entrada. Por cada iteración colocamos en el *vector de eventos* la dirección de cada píxel y se decrementa el valor del píxel en 1 . Cuando el valor del píxel sea 0 no se añade al vector su dirección. Con este método el vector recoge las direcciones de forma creciente al principio, pero conforme los píxeles de menor valor van terminando, sólo aparecen los de mayor valor, quedando en el final, más juntos que al principio, y los de menor valor quedan muy próximos al principio del vector, no teniendo aparición al final.

Por tanto, este método, que es muy sencillo de implementar y rápido de ejecutarse, no respeta la distribución ideal de eventos de los píxeles, que implica un error en los integradores de los receptores, y sólo puede ser útil en aquellas aplicaciones dónde este error no sea importante o significativo. Se evalúa como rápido por la ausencia de procesamiento añadido para la elección de la posición del evento dentro del vector, cuestión que perjudicará a los demás métodos [18].

7.3.2 Método Uniform

Con este método se pretende ser muy estricto con la posición que debe ocupar cada evento de cada píxel más que con la separación entre ellos. Es decir, se parte de la idea de que existe un lugar para cada evento en el vector, lo que significa que nunca existirán colisiones en el tiempo.

De esta forma, recorreremos la matriz una vez, y por cada píxel insertamos en el vector el número de eventos que le corresponda partiendo dicho vector en tantas partes iguales como el valor del píxel. El tamaño del vector de eventos será el mismo para todas las imágenes:

$$\text{Alto} \cdot \text{Ancho} \cdot \text{Nivel Max. Intensidad} = 128 \cdot 128 \cdot 256 = \text{longitud del vector de eventos.}$$

Obviamente ocurriran colisiones, las cuales seran tratadas individualmente asignando la posicion libre mas cercana. Se proponen tres tipos de soluciones diferentes:



- ⇒ **Uniform Back-Forward (*Uniform-BF*)**: Busca en sentido izquierda el slot libre más cercano. En caso de que haya llegado al principio del vector y no haya habido éxito, buscará desde el final del vector hasta la posición actual de la colisión, es decir, actúa de forma circular con prioridad a la izquierda.

- ⇒ **Uniform Forward (*Uniform-F*)**: Busca en sentido derecha el slot libre más cercano. En caso de que haya llegado al final del vector y no haya habido éxito, buscará desde el principio del vector hasta la posición actual de la colisión, es decir, actúa de forma circular con prioridad a la derecha.

- ⇒ **Winner-Take-All (*Uniform-WTA*)**: Se busca el slot libre más cercano en los dos sentidos, izquierda y derecha, de forma simultánea. El evento es colocado en el slot que primero se ha encontrado.

Si para todos los píxeles de la imagen tomamos el mismo vector para buscar los eventos de forma equidistante, vamos a tener muchas posiciones conflictivas, las cuales serán demandadas por todos aquellos píxeles de igual valor. Para evitar esto se implementa el método añadiendo un pequeño desplazamiento dentro del vector para cada píxel, de forma que el píxel $(1,1)$ toma la posición temporal 1 como punto de partida, el píxel $(1,2)$ la 2 y así sucesivamente. Esta mejora del método corrige el crecimiento cuadrático de gasto temporal de ejecución en la búsqueda de huecos al final del procesamiento de una imagen muy cargada en información de intensidad. Sin embargo, conforme se va llenando el vector se va haciendo más costosa la distribución de los eventos en los huecos libres, por lo que el tiempo de computación de este algoritmo crece cuadráticamente⁴ conforme el vector se va completando y, además, hace que los últimos píxeles no respeten la distancia entre sus eventos. Sin embargo, esta situación sólo se dará en aquellas imágenes muy cargadas de eventos AER.

7.3.3 Método Random

7.3.3.1 Random LFSR

Para corregir el defecto del método anterior en cuanto a búsqueda de huecos para cualquier carga de la imagen de entrada se plantea este tercer método donde se introducen las propiedades de los sistemas generadores de números *pseudoaleatorios* basados en registros de desplazamiento *LFSR* (Linear Feedback Shift Register) para la búsqueda de posiciones en el espacio temporal del bus AER.

⁴ El método uniforme calcula las posiciones a ocupar en el vector temporal únicamente en función del valor de intensidad del píxel, y posteriormente lo asigna en el vector, recorriendo una fracción del vector completo si no hay problemas (n). Si en esta asignación encuentra que esa posición del vector está ocupada busca la posición libre más cercana, pudiendo llegar a tener que recorrer el vector completo para buscar el hueco, lo cual aumenta el grado de procesado (n^2).



Dado que la idea es repartir los eventos asociados a la imagen en el vector de la forma más equidistante posible para cada píxel, una distribución aleatoria pero controlada puede conseguir este reparto. Una de las características más atractivas de la generación de números aleatorios usando LFSR es que dada una semilla inicial distinta de cero, no se repetirá ningún valor hasta que todos los posibles valores que pueden generarse con los bits del LFSR se hayan presentado, exceptuando el cero, el cual nunca se genera.

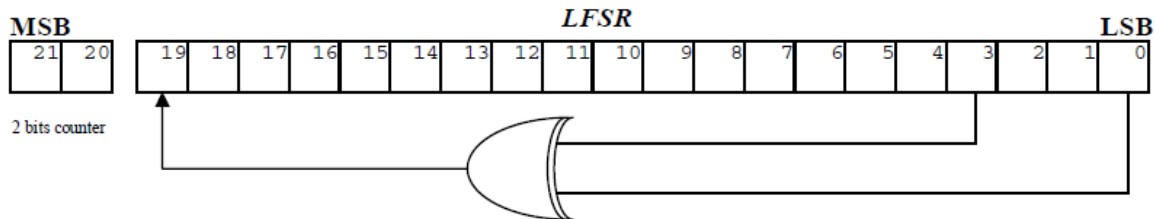


Fig. 18. Estructura LFSR + Contador 2 bits para generación pseudo-random de posiciones

La ventaja de este método es que su tiempo de ejecución no depende del número de eventos a transmitir sino únicamente del tamaño de la imagen, y puede conseguir un reparto aceptable de eventos para minimizar el error de integración de cada píxel en la recepción

7.3.3.2 Random Square

Del método anterior puede observarse que, debido a que los números pseudoaleatorios generados pueden estar o muy juntos o bien muy dispersos, el hecho de separarlos en 4 parcelas no nos va a asegurar que para todos los píxeles se respete la equidistancia, ya que puede ocurrir y ocurre, que dos números aleatorios consecutivos del LFSR no estén distantes, lo cual significa que si un píxel tiene una intensidad alta, puede ocurrir que eventos de este píxel estén muy próximos, pues utilizan más de 1 número aleatorio del LFSR para cubrir a todos sus eventos.

Puede construirse un generador de números aleatorios especial, constituido por dos LFSR, el primero generará la parcela y el segundo la posición dentro de la parcela para cada evento. El funcionamiento que asegure el reparto de huecos, consistirá en generar una posición de parcela para cada píxel usando el primer LFSR, y que el segundo LFSR actúe sobre la parte más significativa de la dirección, generando aleatoriamente las parcelas que deben ocupar los eventos de este píxel.

Algorítmicamente se recorrerá la imagen una vez, y por cada píxel generaremos un número aleatorio con un LFSR de 14 bits (LFSR-14), que usamos para posicionarnos dentro de una parcela del vector. Ahora invocamos al generador LFSR de 8 bits (LFSR-8) tantas veces como el valor de intensidad que tenga el píxel, y usamos estos 8 bits como parte alta de la dirección dentro del vector.

La distribución será como mínimo de tal forma que la separación entre eventos resultará ser el tamaño de una parcela, pero la distancia no se asegura que se mantenga equidistante entre 2 eventos del mismo píxel, pudiendo ser de n veces una parcela, con n entre 1 y 256.

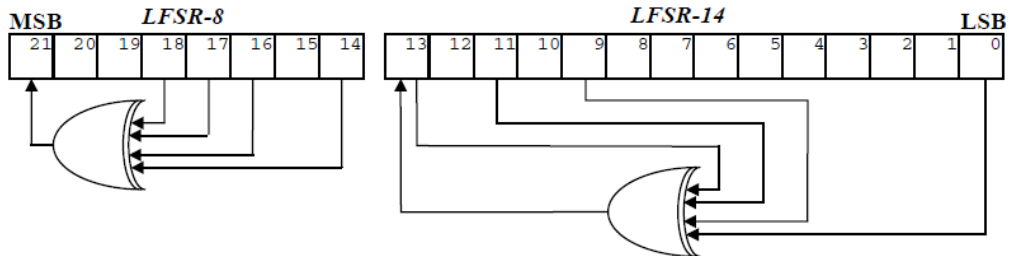


Fig. 19. Estructura LFSR-8 + LFSR-14 para generación pseudo-random de posiciones

7.3.4 Método Exhaustive

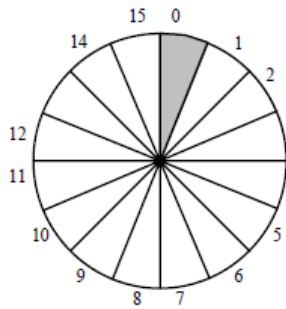
El problema que presenta el método anterior respecto a la distribución de los eventos está en elegir de forma aleatoria las parcelas para los eventos de un píxel, es probable que las parcelas usadas no resulten equidistantes, y por tanto, se cometa un error de distribución. Ahora se tratará de hacer un reparto equidistante de las parcelas.

Con este nuevo método se conserva la idea de dividir el espacio del vector en parcelas, en tantas partes como el color o intensidad máxima asociada a un píxel. Sin embargo, dado que la distribución de eventos en las parcelas no siempre es equidistante, debido a que el número de eventos no es divisible por el número de parcelas, hay que cometer errores en esta distribución, haciendo que la separación entre eventos no sea siempre un número fijo de parcelas.

Lo que pretende este método es asumir este error y minimizarlo, de forma que el error de distribución sea como máximo de una parcela. Para evitar tener que almacenar el vector completamente antes de su transmisión, se ha buscado un algoritmo que va seleccionando en tiempo real qué evento de qué píxel debe ser puesto en el vector de tiempo en cada instante, o ninguno, si en ese instante no debe ir evento alguno.

La formulación consiste en procesar la imagen K veces, siendo K el valor máximo de intensidad asociado a un píxel. Por tanto, sea s el barrido de 1 a K . Por cada s analizamos cada píxel, de forma que si el resto de dividir la multiplicación de s por el valor del píxel entre la intensidad máxima (K) supera a K , entonces un evento de este píxel debe ser puesto en el bus AER en este instante, y el contador de tiempo incrementarse a un siguiente evento del bus AER (una nueva posición del vector de eventos). Es decir, se debe de cumplir la siguiente condición: $(s \cdot P_{i,j}) \bmod K \geq K$.

Supongamos una circunferencia dividida en K porciones. Sea $p = \text{píxel}[i][j]$ la intensidad de la posición (i, j) de la imagen. Para buscar las parcelas donde deben aparecer eventos para este píxel, debemos avanzar de p en p posiciones, y cada vez que avanzamos significa que pasamos a la siguiente parcela. Sólo colocamos el evento en la parcela si al avanzar las p porciones de la circunferencia pasamos por la porción K de la tarta. En la Fig. 20 se muestra un ejemplo.



Sea $K=16$, $p=3$. Entonces hemos de avanzar de 3 en 3. La secuencia será: 2,5,8,11,14,1 (Son 6 incrementos, por tanto el primer evento se coloca en la parcela 6), 4,7,10,13,0 (Son 5 incrementos, luego se coloca el segundo evento en la parcela 11), 3,6,9,12,15 (tercer evento en parcela 16). Este método asegura que no se va a avanzar por la tarta más de K veces. En la última parcela siempre hay evento.

Fig. 20. Ejemplo de distribución de eventos del método Exhaustive

De esta forma se consigue una distribución más uniforme de los eventos para cada píxel.

El problema de esta distribución radica, en que siempre se toma como parcela a evitar la primera, y siempre se toma como parcela a usar la última, de forma, que desde el punto de vista del tráfico en el bus AER, vamos a tener una parcela de 16383 ($128 \cdot 127 + 127$) eventos todos sin usar y a continuación 16383 eventos todos con información, lo cual puede provoca llevar al máximo las consecuencias de posibles saturaciones de receptores por picos de tráfico.

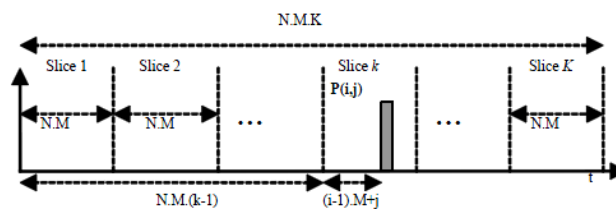


Fig. 21. Distribución en slices del método Exhaustive

Para evitar esto la idea consiste, simplemente en ir rotando la posición de la parcela 0, de forma que al analizar la imagen para cada intensidad sumamos dicha intensidad a la multiplicación de k por el valor del píxel antes de hacer la operación módulo, moviendo así el origen del ejemplo anterior, en cada intensidad de color analizada.

A esta mejora la denominamos **Exhaustive Enhanced**, por tanto la condición que se ha de cumplir para emitir el evento en el vector, es la siguiente: $(s \cdot P_{i,j}) \bmod K + P_{i,j} \geq K$.

Capítulo 8

8 Arquitectura del sistema: Sináptica y Neuronal

Han sido desarrollados dos arquitecturas de envío de eventos por el bus AER [14].

Estas dos arquitecturas han sido diseñadas como una aplicación nueva e innovadora respecto al sistema AER existente, aplicado en un entorno real a un contexto de automoción. Por ello se estudiará estas dos arquitecturas diseñadas para mostrar que mejoras y diferencias entre ellas existe.

En el caso de la *arquitectura Sináptica* envía toda la información de la imagen por el bus AER, en cambio en la *arquitectura Neuronal* procesa la información antes de ser enviada por el bus.

8.1 Arquitectura Sináptica

La entrada es tomada por una cámara estándar con un sensor CMOS, en nuestro caso será un video digital. Éste es descompuesto en frames en la parte de *AER Imager*, él cual se encarga de transformar dichos frames a un tamaño y niveles de intensidad compatibles con el.

Además extrae el movimiento con las distintas técnicas citadas en el punto 6: [Técnicas de extracción de movimiento](#). Por último transforma los frames resultantes en eventos con los métodos ya también citados en el punto 7.3: [Métodos de generación de eventos](#).

Estos eventos contiene toda la información de la imagen. Todos ellos son enviados por el bus AER marcados con un timestamp.

En la parte de recepción de los eventos, estos son recibidos y posteriormente procesados, observando cuales de ellos pertenecen a la zona de interés definida previamente.

Por lo que en el proceso de esta arquitectura solo se encarga de la transmisión de eventos, ya que estos son procesados una vez que son recibidos por el bus AER.

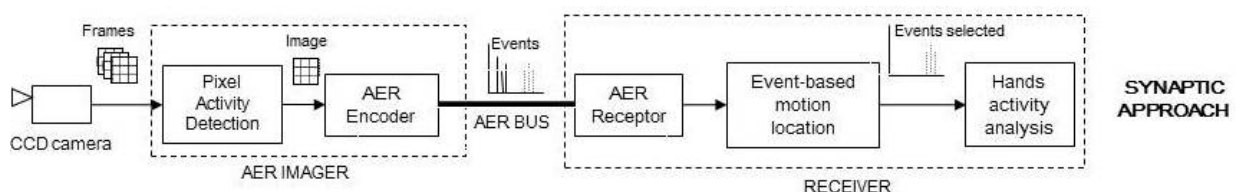


Fig. 22. Diseño de la arquitectura Sináptica



8.2 Arquitectura Neuronal

Esta arquitectura es muy similar a la anterior. El video digital como entrada del sistema *AER Imager* es descompuesto en los frames que le forman, redimensionando el tamaño del frame a un máximo de 128×128 , a escala de grises, todo ello si fuera necesario.

Se extrae el movimiento con las distintas técnicas citadas y se transforma a eventos AER con los distintos métodos vistos.

En este punto difiere de la anterior arquitectura, los eventos son procesados antes de transmitirlos por el bus AER. Son emitidos aquellos eventos que pertenecen a la zona de interés que se ha definido, por lo que la “carga” en el bus AER es mucho menor que en el enfoque sináptico.

La principal diferencia entre las dos arquitecturas es la etapa donde los eventos contenidos dentro de la región de interés (ROI) o región de movimiento esta hecho. En el caso de la arquitectura neuronal, el procesado se hace antes de la emisión de los eventos, por lo que solamente se envían eventos que pertenecen a la región de movimiento, mientras que en la sináptica el procesado se hace una vez recibido todos los eventos de la imagen, es decir, solamente se encarga del envío o transmisión de eventos.

Como consecuencia varia el tamaño del vector de eventos final, el tráfico en el bus AER en el caso de la arquitectura neuronal será menor que en la sináptica, ya que sólo se envía información relevante, mientras que en la sináptica el tráfico aumenta en gran medida debido al envío de toda la información posible.

La diferencia de rendimiento y tamaño entre ambos sistemas se verá posteriormente en el capítulo de pruebas y resultados, donde se explicará con más detalle.

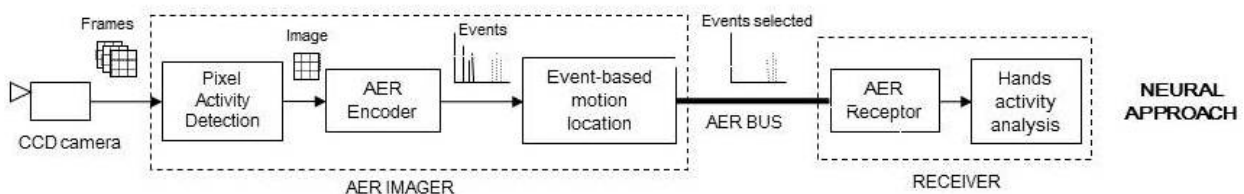


Fig. 23. Diseño de la arquitectura Neuronal



Capítulo 9

9 Resultados

En este apartado se comentarán en detalle las pruebas realizadas con los distintos algoritmos de codificación a eventos AER, y las distintas arquitecturas diseñadas, aplicadas a automoción con el fin de detectar el porcentaje de movimiento del conductor.

Para las pruebas se ha desarrollado un módulo capaz de calcular de formas automática dichas estadísticas, en formato Excel.

9.1 Consideraciones previas

Los datos de entrada han sido tomados de un subconjunto de imágenes de 15.000 fotogramas que componen el video original, con una conducción por parte de un profesional en un simulador de conducción de camiones, en una zona interurbana.

Los fotogramas han sido extraídos del proyecto **CABINTEC** (*ref: PSE-370100-2007-2*) cuyo proyecto tiene como objetivo, el diseño del habitáculo de un vehículo dotado con tecnologías inteligentes capaces de detectar el comportamiento del conductor (hábitos saludables frente a conductas de peligro en el contexto de una conducción segura), así como el estudio de los parámetros que caracterizan al vehículo y al conductor en los instantes previos a un accidente.

Los fotogramas tienen una resolución original de 640x480 píxeles y una velocidad del video a 14 fps.

Como se ha explicado en las primeras secciones del proyecto, ha sido adaptado con el fin de poder ser visualizado en el software jAER para la representación visual de eventos AER. Este software tiene una serie de restricciones y por ello, este proyecto ha sido adaptado a ellas. La resolución máxima que puede tomar una imagen es de 128x128 píxeles (debido al tamaño máximo que dispone hasta ahora el sensor de la cámara Retina) por eso, para la realización de estas pruebas, se han reducido los fotogramas a una resolución de 96x128 píxeles (para respetar las proporciones de la imagen y no distorsionarla). Además, se trabajará con fotogramas a escala de grises para reducir la complejidad del problema.

El número de eventos AER que se emitirá como máximo por píxel, es de 4, debido a que la cámara Retina envía dicha cantidad con el fin de reducir redundancia de información por píxel.



9.2 Generador de estadísticas

Se ha desarrollado un módulo que genera de forma automática las estadísticas en formato Excel. Mide la cantidad de actividad de movimiento de las manos detectada en las regiones de interés definidas.

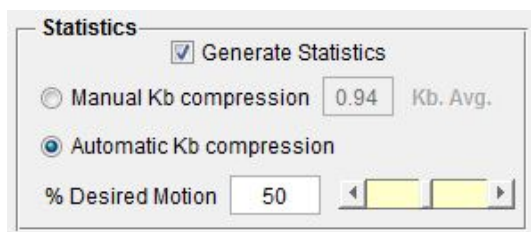


Fig. 24. Módulo encargado de las estadísticas

La ventaja de este módulo es que en una sola ejecución, calcula las estadísticas para un rango de porcentaje de emisión de eventos, es decir, calcula el movimiento detectado cuando se envía un porcentaje específico. El rango probado es el de [100% 90% 80% 70% 60% 50% 40% 30% 20% 10% 5% 4% 3% 2% 1%]. Además calcula las estadísticas tanto para la **arquitectura Sináptica** como para la **arquitectura Neuronal** en la misma ejecución, diferenciados por un título y por cada región interés definido, una hoja de cálculo distinta dentro del mismo fichero Excel.

Las estadísticas calculadas por cada porcentaje de emisión de eventos AER, y por cada arquitectura (Neuronal y Sináptica) son: *total de pixeles en el frame; su desviación típica; total de eventos en el frame; tamaño en kb que ocupa el vector de eventos; pixeles en la ROI detectados; su desviación típica; eventos en la ROI detectados; porcentaje de movimiento en la ROI detectado; su desviación típica; porcentaje de movimiento detectado en la imagen completa y su desviación típica*. Todos estos datos son en media por ello, también se calcula su desviación típica.

Además se muestra dos tipos de datos que cobra gran importancia. El primero, es la cantidad (en porcentaje) de eventos AER totales que se ha de emitir en el bus AER para que su tamaño expresado en Kb, sea igual al tamaño que ocupa las imágenes resultantes de extraer el movimiento en disco, con una compresión JPG. El segundo, es la cantidad (en porcentaje) de eventos AER totales que se ha de emitir en el bus AER para que el porcentaje de movimiento detectado en las regiones de interés, sea igual al parámetro seleccionado, por ejemplo, si se desea obtener solamente un 50% de movimiento en la ROI, indicará el porcentaje de eventos AER que han de ser necesarios emitir para detectar dicho porcentaje definido y el tamaño expresado en Kb que ocupa.

9.3 Evaluación de algoritmos

Se pretende ilustrar la ventaja de la codificación AER frente al uso de fotogramas mediante un simple ejemplo. Como modelo, se ha tomado una imagen de tamaño 360x720 píxeles.



Fig. 25. Imagen ejemplo para codificar

A continuación, se muestra una tabla donde las columnas son los distintos algoritmos de codificación AER utilizados y las filas los distintos porcentajes de eventos emitidos.



























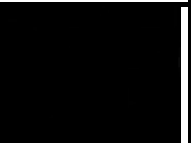

%	SCAN	RANDOM	UNIFORM	EXHAUSTIVE
100				
50				
25				
15				
10				
5				
1				

Tabla 3. Reconstrucción de eventos AER



Se puede apreciar la diferencia visual que hay entre los distintos algoritmos, dependiendo del porcentaje emitido, siendo el 100% la imagen original. Las imágenes de la tabla son la reconstrucción de los eventos AER, extraídos de la imagen original.

Porcentaje	Eventos	SCAN		RANDOM		UNIFORM		EXHAUSTIVE	
		Píxeles	% Acierto	Píxeles	% Acierto	Píxeles	% Acierto	Píxeles	% Acierto
100	10444795	95729	100	95729	100	95729	100	95729	100
50	5222398	95729	100	94185	98.38	94160	98.36	94040	98.23
25	2611199	95729	100	91162	95.22	91501	95.58	91118	95.18
15	1566720	95729	100	88053	91.98	87879	91.79	86007	89.84
10	1044480	95729	100	84873	88.65	82291	85.96	82082	85.74
5	522240	95729	100	78796	82.31	75193	78.54	75384	78.74
1	104448	95729	100	51688	54	46192	48.25	57703	60.27

Tabla 4. Porcentajes de acierto

Como se observa en la tabla resultado, la imagen original se compone de **95729** píxeles (con valores distintos de 0). En este ejemplo, se ha emitido tantos eventos AER como nivel de intensidad tenía el píxel, es decir, si por ejemplo el píxel (1,1) tiene un nivel de intensidad de 234, se emitirá 234 eventos AER en el vector de eventos. En este caso, no se ha tenido en cuenta la reducción de redundancia (*emitir como máx. 4 eventos por píxel*) con el objetivo de poder ilustrar el ejemplo con el mayor detalle posible.

La diferencia entre los algoritmos, es la manera en que distribuyen los eventos AER en el vector de eventos, por lo que al enviar un porcentaje reducido, las distintas técnicas de distribución influyen.

Para este ejemplo se puede apreciar que el mejor algoritmo que se adapta es el *Scan*, aunque de forma visual no lo refleja, pero la distribución de los eventos con información relevante se encuentra al principio del vector y por ello se obtiene dicho porcentaje de acierto.

El Random al ser aleatorio, en la ejecución de este ejercicio se obtiene estos datos, mientras que el Uniform o Exhaustive realizan una distribución de eventos más uniforme, pero para este ejemplo, emitiendo un porcentaje de eventos bajo se obtiene peores resultado que el Scan, ya que los eventos con información relevante se encuentran más dispersos dentro del vector.

9.4 Tablas de resultados y gráficas

Los resultados han sido obtenidos de un conjunto de 2.000 fotogramas, extraídos de un video grabado en un simulador de conducción totalmente inmersivo, con una velocidad de 14fps y una resolución de imagen de 96x128 pixeles en escala de gris.

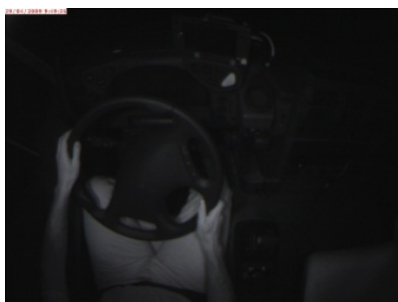


Fig. 26. Fotograma utilizado en los resultados

A continuación se presentan los resultados resumidos en tablas, que han sido obtenidos al aplicar las dos arquitecturas diseñadas para un entorno de conducción, incluyendo las distintas técnicas de extracción de movimiento (FD, Mean y Mode) y dos implementaciones a la hora de emitir eventos AER (Escalando la Imagen y Emitiendo los 4 primeros eventos del pixel).

Los resultados siguen la siguiente estructura para mayor facilidad a la hora de interpretar los datos. En la columna izquierda de la página, se presenta las tablas con los datos obtenidos aplicando una arquitectura **Sináptica** o **Neuronal**, con las distintas técnicas de extracción de movimiento (*FD*, *Mean* y *Mode*) una debajo de la otra, quedando en la columna derecha la gráfica que representa los datos de la tablas. De esta manera, los datos pueden ser comparados de manera más eficiente y clara.

Se presentan las tres técnicas de extracción de movimiento, **Frame Difference** con un intervalo de valor 1, **Mean** y **Mode** con valores de 30 para el tamaño del buffer y 5 como intervalo. Estos valores han sido seleccionados tras los resultados obtenidos de una batería de pruebas realizada, presentando unos resultados más óptimos, frente a los demás valores probados.

Además se presenta los resultados obtenido de aplicar las dos técnicas de reducción de redundancia comentadas en el punto 6.3 [Reducción de Ruido y Redundancia.](#), emitiendo los primeros **cuatro eventos** del pixel y escalando la imagen con **valores máximos de cuatro**, como ya se ha indicado en dicho capítulo.

Se han definido tres regiones de interés ilustradas en la Fig. 27. La cuales pertenecen son la zona del **volante**, la zona de la **caja de cambios** y la zona del **freno de mano** que disponen los camiones.

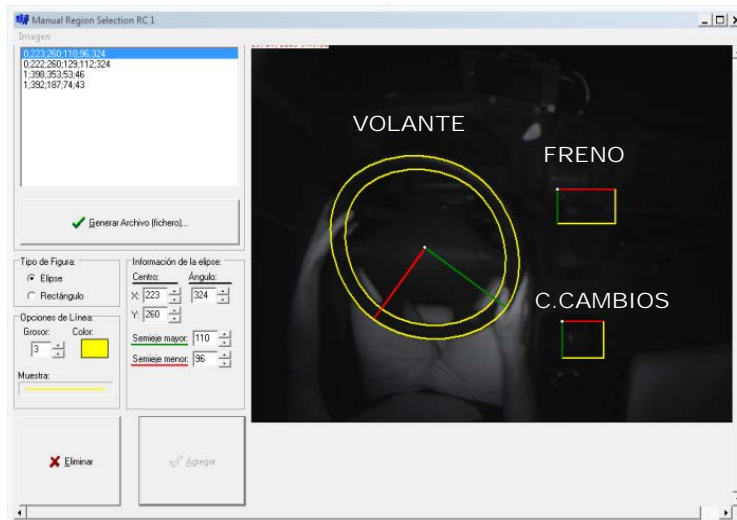


Fig. 27. ROIs definidas para el resultado

En este documento se presentarán los resultados obtenidos en la región de interés del volante, debido a que en esta zona, presenta más actividad de las manos respecto a las demás regiones, durante la secuencia de video tomada como entrada. Las tablas y gráficas de las regiones de interés del freno y caja de cambios, están disponibles en formato digital, en el cd adjuntado.

Se presentan tres tablas de medidas de evaluación diferentes:

- En primer lugar se muestra la tabla que refleja la cantidad de movimiento detectado, dependiendo del porcentaje de eventos AER emitidos. Como se ha comentado, la tabla situada a la izquierda, muestra los datos obtenidos tanto de la arquitectura *Sináptica* como la arquitectura *Neuronal*, quedando en la parte derecha la gráfica que representa dichos datos.
- En segundo lugar, se ha medido la capacidad del sistema en detectar y enviar el movimiento. Para ello, se presentan dos tablas por cada arquitectura mencionada anteriormente. Una tabla muestra el tamaño expresado en Kb ocupado en disco de los eventos AER de media por cada frame, al detectar un **50%** y un **100%** de movimiento en la región de interés.
- Por otro lado, se ha comparado el sistema basado en AER con un sistema convencional basado en fotogramas con una compresión JPG. Para ello se presenta otra tabla, mostrando el porcentaje de movimiento detectado en dicha región, cuando su tamaño expresado en Kb es igual al tamaño, también expresado en Kb, que ocupa la imagen movimiento en disco, al utilizar una compresión común como es el **JPG**.



9.4.1 Resultados obtenidos de escalar a 4 eventos AER por píxel

SINÁPTICA – FD – ESCALADO 4 EVENTOS				
% Movimiento Detectado en el VOLANTE				
% Emitido	SCAN	RANDOM	UNIFORM WTA	EXHAUSTIVE
1	3,86	1,56	0,52	2,59
2	8,00	3,71	1,26	5,56
3	11,72	5,95	1,78	8,56
4	14,95	7,97	2,44	11,49
5	18,29	10,00	3,03	14,37
10	32,61	19,69	6,66	26,58
20	58,76	37,16	24,77	49,00
30	77,42	50,77	49,37	64,23
40	90,15	63,12	71,11	74,53
50	97,47	72,90	84,38	82,82
60	99,75	80,32	94,71	88,53
100	100	100	100	100

Tabla 5. Eval. de la emisión y detección de la Act. Manos. Sináp. y FD

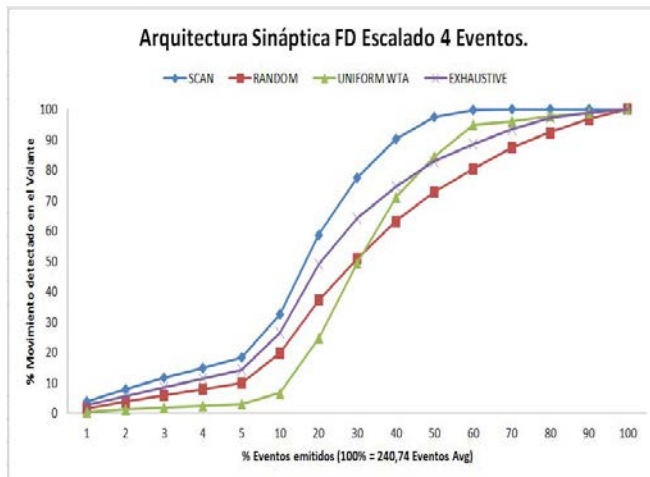


Fig. 28. Eval. de la emisión y detección de la Act. Manos. Sináp. y FD

SINÁPTICA – MEAN – ESCALADO 4 EVENTOS				
% Movimiento Detectado en el VOLANTE				
% Emitido	SCAN	RANDOM	UNIFORM WTA	EXHAUSTIVE
1	2,98	1,72	0,00	1,62
2	6,13	3,42	0,00	3,37
3	9,01	5,12	0,00	6,17
4	11,93	6,77	0,02	9,28
5	15,23	8,42	0,12	12,33
10	29,39	16,33	2,68	26,47
20	51,90	30,76	15,38	43,18
30	67,61	43,72	34,56	51,63
40	83,35	54,91	57,92	59,59
50	94,45	65,16	74,94	72,81
60	99,42	73,93	91,87	82,74
100	100	100	100	100

Tabla 6. Eval. de emisión y detección de la Act. Manos. Sináp. y MEAN

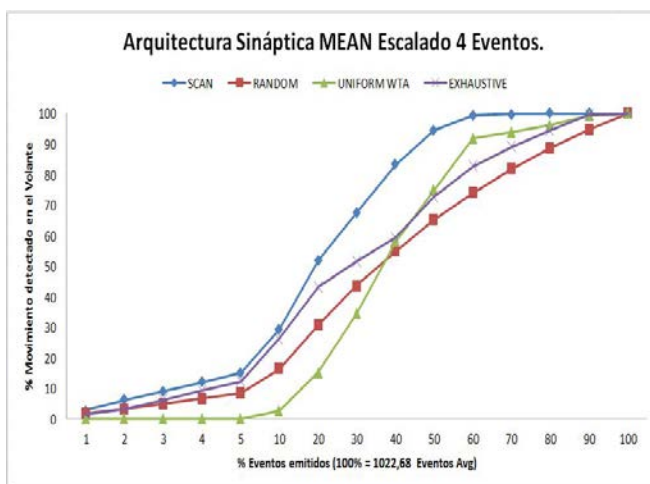


Fig. 29. Eval. de emisión y detección de la Act. Manos. Sináp. y MEAN

SINÁPTICA – MODE – ESCALADO 4 EVENTOS				
% Movimiento Detectado en el VOLANTE				
% Emitido	SCAN	RANDOM	UNIFORM WTA	EXHAUSTIVE
1	2,53	1,73	0,00	1,76
2	6,14	3,48	0,00	4,12
3	9,29	5,18	0,01	6,27
4	12,61	6,85	0,03	8,93
5	15,96	8,53	0,07	12,00
10	28,90	16,64	1,47	27,04
20	48,24	31,43	14,04	47,33
30	66,70	44,62	35,98	52,07
40	83,96	55,87	59,71	58,91
50	96,21	65,84	75,18	71,43
60	99,69	74,52	91,25	81,42
100	100	100	100	100

Tabla 7. Eval. de emisión y detección de la Act. Manos. Sináp. y MODE

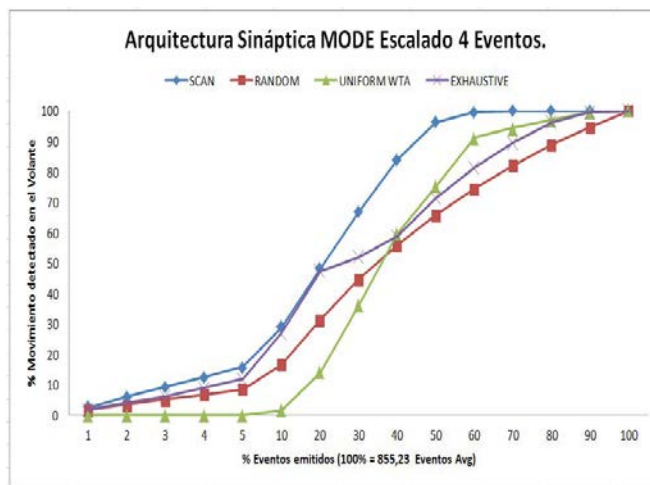


Fig. 30. Eval. de emisión y detección de la Act. Manos. Sináp. y MODE



NEURONAL – FD – ESCALADO 4 EVENTOS				
% Movimiento Detectado en el VOLANTE				
% Emitido	SCAN	RANDOM	UNIFORM WTA	EXHAUSTIVE
1	0,90	0,90	0,88	0,90
2	2,66	2,66	2,57	2,66
3	4,63	4,61	4,39	4,63
4	6,68	6,63	6,22	6,68
5	9,07	8,94	8,43	9,07
10	19,86	19,01	18,13	19,84
20	42,91	39,04	39,50	42,24
30	64,08	54,14	59,58	60,32
40	79,84	65,17	74,06	71,24
50	91,20	75,37	84,79	77,90
60	96,11	81,88	92,07	82,63
100	100	100	100	100

Tabla 8. Eval. de la emisión y detección de la Act. Manos. Neuro. y FD

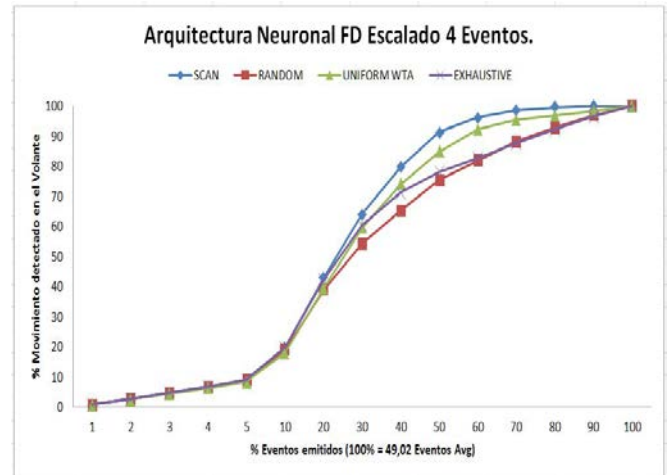


Fig. 31. Eval. de la emisión y detección de la Act. Manos. Neuro. y FD

NEURONAL – MEAN – ESCALADO 4 EVENTOS				
% Movimiento Detectado en el VOLANTE				
% Emitido	SCAN	RANDOM	UNIFORM WTA	EXHAUSTIVE
1	1,51	1,51	1,41	1,51
2	3,53	3,50	3,29	3,53
3	5,13	5,07	4,70	5,13
4	6,80	6,68	6,10	6,80
5	8,61	8,41	7,54	8,60
10	17,13	16,32	14,82	17,03
20	34,19	30,90	30,26	33,48
30	51,38	43,84	46,58	47,34
40	68,14	55,16	61,27	54,82
50	83,99	65,37	75,05	62,03
60	94,72	74,08	88,19	70,85
100	100	100	100	100

Tabla 9. Eval. de emisión y detección de la Act. Manos. Neuro. y MEAN

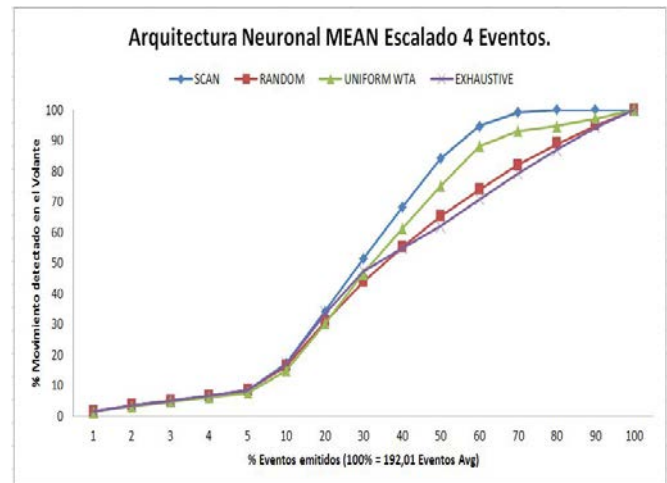


Fig. 32. Eval. de emisión y detección de la Act. Manos. Neuro. y MEAN

NEURONAL – MODE – ESCALADO 4 EVENTOS				
% Movimiento Detectado en el VOLANTE				
% Emitido	SCAN	RANDOM	UNIFORM WTA	EXHAUSTIVE
1	1,64	1,63	1,48	1,64
2	3,51	3,49	2,93	3,51
3	5,38	5,32	4,47	5,38
4	7,01	6,88	5,76	7,01
5	8,88	8,65	7,26	8,88
10	17,89	16,91	14,84	17,82
20	35,62	31,63	30,24	34,73
30	53,46	44,68	46,55	49,04
40	71,09	55,95	61,77	55,76
50	86,90	66,21	75,22	61,90
60	95,38	74,59	88,42	70,23
100	100	100	100	100

Tabla 10. Eval. de emisión y detección de la Act. Manos. Neuro. y MODE

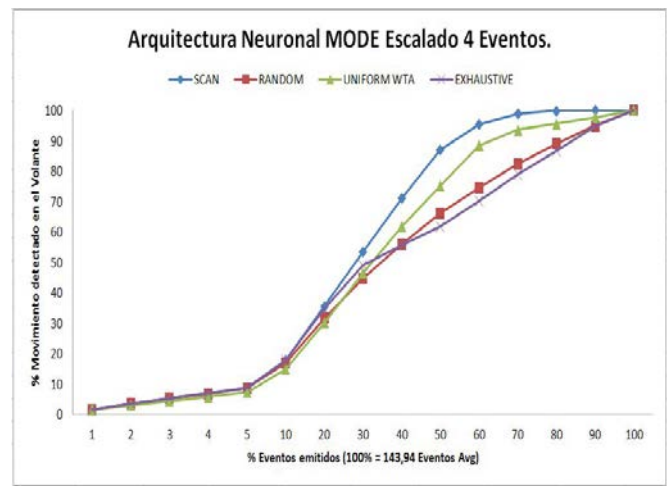


Fig. 33. Eval. de emisión y detección de la Act. Manos. Neuro. y MODE



	SINÁPTICA - ESCALADO 4 EVENTOS					
	FD		MEAN		MODE	
	Tamaño Eventos Kb 50% ROI Volante	Tamaño Eventos Kb 100% ROI Volante	Tamaño Eventos Kb 50% ROI Volante	Tamaño Eventos Kb 100% ROI Volante	Tamaño Eventos Kb 50% ROI Volante	Tamaño Eventos Kb 100% ROI Volante
SCAN	0,0828	0,3025	0,3571	1,4233	0,3235	1,0440
RANDOM	0,1362	0,4408	0,6667	1,8726	0,5456	1,5660
UNIFORM WTA	0,1366	0,4408	0,6673	1,8135	0,5552	1,5546
EXHAUSTIVE	0,1085	0,4408	0,5352	1,7847	0,4120	1,5530

Tabla 11. Evaluación del tamaño de los datos emitidos para detectar el 50% o el 100% de la actividad en la ROI. Sináptico

	SINÁPTICA - ESCALADO 4 EVENTOS		
	FD (JPG = 0,9431Kb)	MEAN (JPG = 0,9442Kb)	MODE (JPG = 0,9528Kb)
	% Movimiento Detectado al igualar JPG. ROI Volante	% Movimiento Detectado al igualar JPG. ROI Volante	% Movimiento Detectado al igualar JPG. ROI Volante
SCAN	100	94,73	99,76
RANDOM	100	65,61	75,10
UNIFORM WTA	100	76,05	92,11
EXHAUSTIVE	100	73,26	82,77

Tabla 12. Comparativa del sistema propuesto frente al JPG. Sináptico

	NEURONAL - ESCALADO 4 EVENTOS					
	FD		MEAN		MODE	
	Tamaño Eventos Kb 50% ROI Volante	Tamaño Eventos Kb 100% ROI Volante	Tamaño Eventos Kb 50% ROI Volante	Tamaño Eventos Kb 100% ROI Volante	Tamaño Eventos Kb 50% ROI Volante	Tamaño Eventos Kb 100% ROI Volante
SCAN	0,0831	0,4017	0,3409	1,6026	0,3540	1,3576
RANDOM	0,1169	0,3903	0,6687	1,7686	0,5482	1,4737
UNIFORM WTA	0,1186	0,3484	0,6728	1,5568	0,5112	1,2675
EXHAUSTIVE	0,0874	0,3365	0,5846	1,5539	0,4533	1,2488

Tabla 13. Evaluación del tamaño de los datos emitidos para detectar el 50% o el 100% de la actividad en la ROI. Neuronal

	NEURONAL - ESCALADO 4 EVENTOS		
	FD (JPG = 0,9431Kb)	MEAN (JPG = 0,9442Kb)	MODE (JPG = 0,9528Kb)
	% Movimiento Detectado al igualar JPG. ROI Volante	% Movimiento Detectado al igualar JPG. ROI Volante	% Movimiento Detectado al igualar JPG. ROI Volante
SCAN	100	90,08	95,69
RANDOM	100	65,64	75,30
UNIFORM WTA	100	75,32	88,42
EXHAUSTIVE	100	75,42	78,80

Tabla 14. Comparativa del sistema propuesto frente al JPG. Neuronal

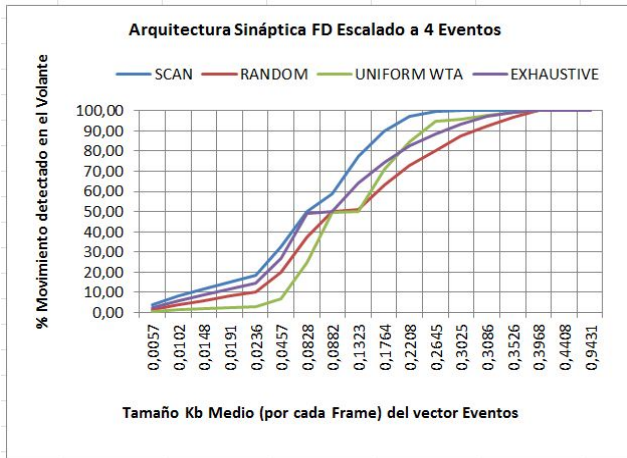


Fig. 34. Evaluación y comparativa de tamaño frente al JPG. Sináptica y FD

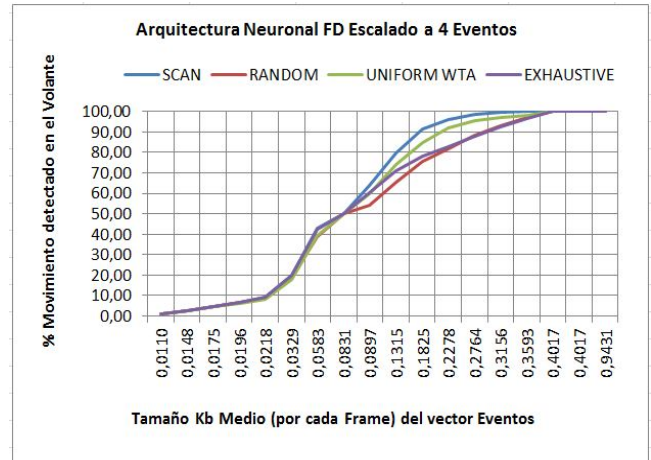


Fig. 35. Evaluación y comparativa de tamaño frente al JPG. Neuronal y FD

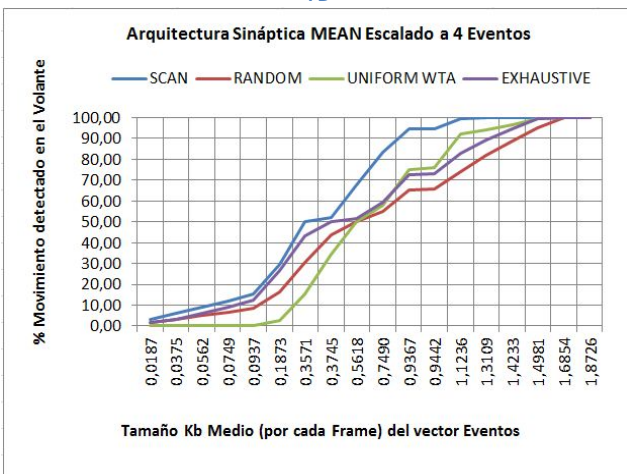


Fig. 36. Evaluación y comparativa de tamaño frente al JPG. Sináptica y MEAN

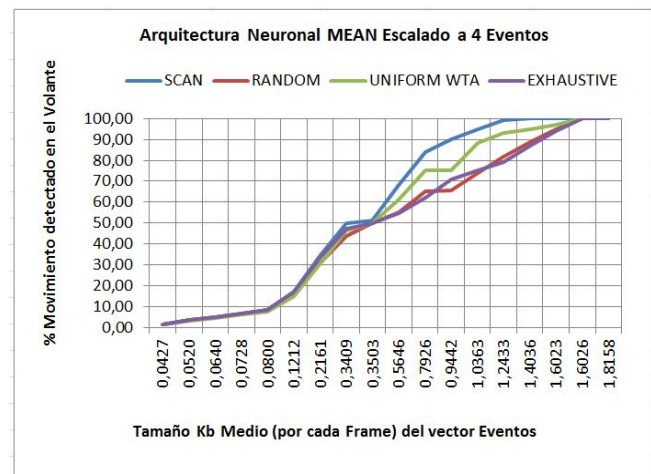


Fig. 37. Evaluación y comparativa de tamaño frente al JPG. Neuronal y MEAN

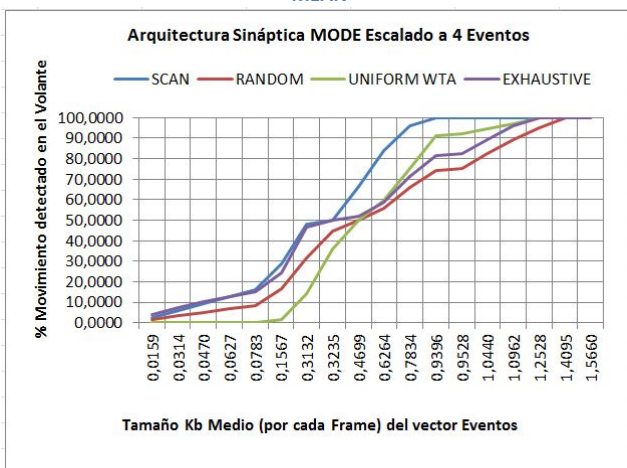


Fig. 38. Evaluación y comparativa de tamaño frente al JPG. Sináptica y MODE

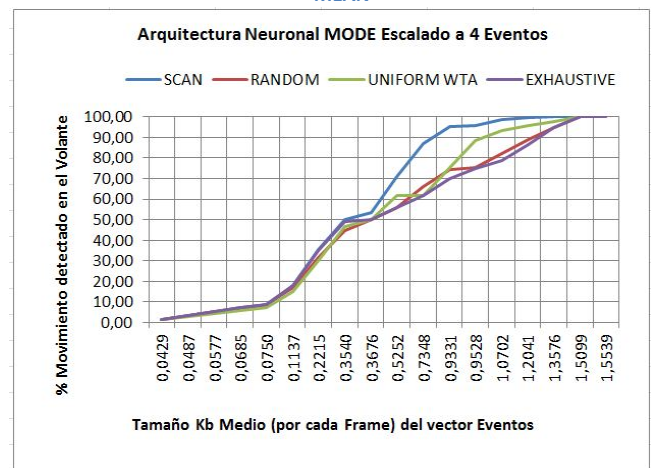


Fig. 39. Evaluación y comparativa de tamaño frente al JPG. Neuronal y MODE



Estos resultados son obtenidos al escalar la imagen a un máximo de 4 eventos por píxel, es decir, los valores de intensidad por cada píxel tomarán un rango comprendido entre el valor mínimo de 0 y el máximo de 4, por lo que como cota se podrá emitir 4 eventos AER por cada píxel.

Como se puede apreciar en los datos de las tablas (Tabla 5 a la Tabla 10) y reflejados en las gráficas (Fig. 28 a la Fig. 33), el método **Scan** alcanza un porcentaje de detección de movimiento más alto y más rápido que los demás métodos, de forma que con menos porcentaje de emisión de eventos, se adquiere un alto porcentaje de detección de movimiento.

Esto es una ventaja ya que por ejemplo, para solo un 1% de emisión, se obtiene un 3,86% de información del movimiento en la zona de interés del volante.

Los métodos Uniform WTA y Exhaustive, obtienen unos resultados parecidos y buenos, pero no tan óptimos como el anterior método.

El método Random, como es de esperar obtiene el peor porcentaje de detección, debido a su naturaleza aleatoria, ya que no es constante en la distribución de eventos AER.

Además, la mejora en el rendimiento del método Scan es mayor para la arquitectura Sináptica que en la Neuronal, debido a que la propia naturaleza del escalado *altera* el orden de la información emitida.

Comenzando con la arquitectura Sináptica, en la Tabla 11 puede observarse que en la mayoría de los casos, con un tamaño (en Kb) muy pequeño del vector de eventos, se envía una gran cantidad de información.

El mejor rendimiento se obtiene con la técnica de extracción de movimiento **Frame Difference** (FD) y el método **Scan**, donde con sólo **0,30 Kb** se envía toda la información de movimiento de la ROI, frente al tamaño del JPG de **0,94 Kb**, obteniendo así un **67,93%** de reducción de tamaño respecto a la compresión común JPG.

Comparando con el tamaño del JPG, como puede apreciarse en la Tabla 12, en el caso de Frame Difference siempre se supera el rendimiento del JPG para cualquier método de codificación AER, alcanzando el **100%** de movimiento detectado para todos los casos.

En el caso de la arquitectura Neuronal, los resultados son muy semejantes, como puede observarse en las Tabla 13 y Tabla 14.



9.4.2 Resultados obtenidos de los 4 primeros eventos AER por píxel

SINÁPTICA – FD – PRIMEROS 4 EVENTOS				
% Movimiento Detectado en el VOLANTE				
% Emitido	SCAN	RANDOM	UNIFORM WTA	EXHAUSTIVE
1	3,87	3,90	0,63	3,87
2	8,01	8,09	1,33	8,01
3	11,73	11,72	2,06	11,73
4	14,96	15,45	2,57	14,96
5	18,30	18,96	3,35	18,30
10	32,61	35,29	7,12	32,61
20	58,76	60,41	56,15	58,76
30	77,42	76,99	99,40	77,42
40	90,15	87,54	99,40	90,15
50	97,47	94,22	99,40	97,47
60	99,76	97,91	99,62	99,76
100	100	99,40	99,62	100

Tabla 15. Eval. de la emisión y detección de la Act. Manos. Sináp. y FD

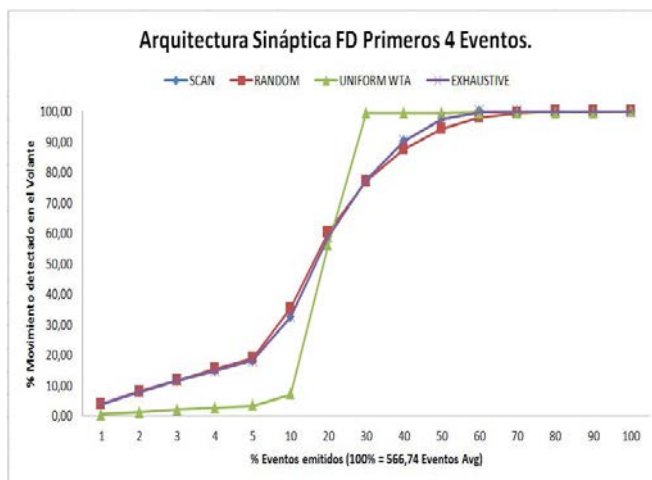


Fig. 40. Eval. de la emisión y detección de la Act. Manos. Sináp. y FD

SINÁPTICA – MEAN – PRIMEROS 4 EVENTOS				
% Movimiento Detectado en el VOLANTE				
% Emitido	SCAN	RANDOM	UNIFORM WTA	EXHAUSTIVE
1	7,53	3,87	0,00	7,53
2	14,66	7,70	0,00	14,66
3	22,75	11,39	0,00	22,75
4	29,62	15,04	0,00	29,62
5	36,00	18,49	0,06	36,00
10	58,27	34,27	3,66	58,27
20	96,53	59,06	52,12	96,53
30	100	76,12	100	100
40	100	87,21	100	100
50	100	93,77	100	100
60	100	97,45	100	100
100	100	99,20	100	100

Tabla 16. Eval. de emisión y detección de la Act. Manos. Sináp. Y MEAN

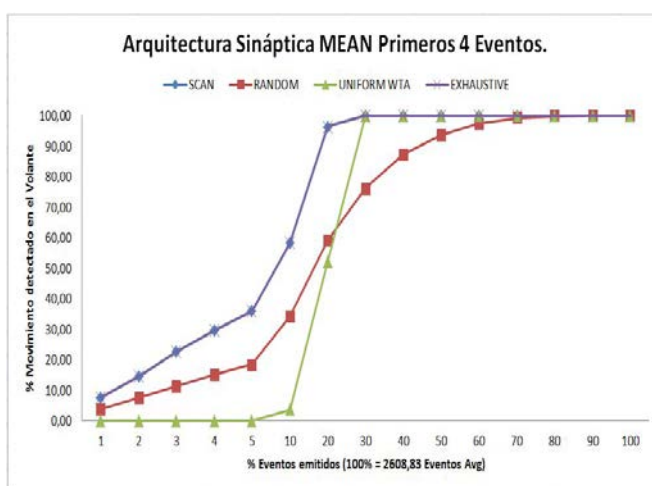


Fig. 41. Eval. de emisión y detección de la Act. Manos. Sináp. y MEAN

SINÁPTICA – MODE – ESCALADO 4 EVENTOS				
% Movimiento Detectado en el VOLANTE				
% Emitido	SCAN	RANDOM	UNIFORM WTA	EXHAUSTIVE
1	7,69	4,00	0,00	7,69
2	15,54	7,89	0,00	15,54
3	22,70	11,56	0,01	22,70
4	28,68	15,15	0,03	28,68
5	33,98	18,69	0,10	33,98
10	57,69	34,41	1,89	57,69
20	98,20	59,07	51,09	98,20
30	100	76,01	100	100
40	100	87,07	100	100
50	100	93,66	100	100
60	100	97,42	100	100
100	100	99,21	100	100

Tabla 17. Eval. de emisión y detección de la Act. Manos. Sináp. y MODE

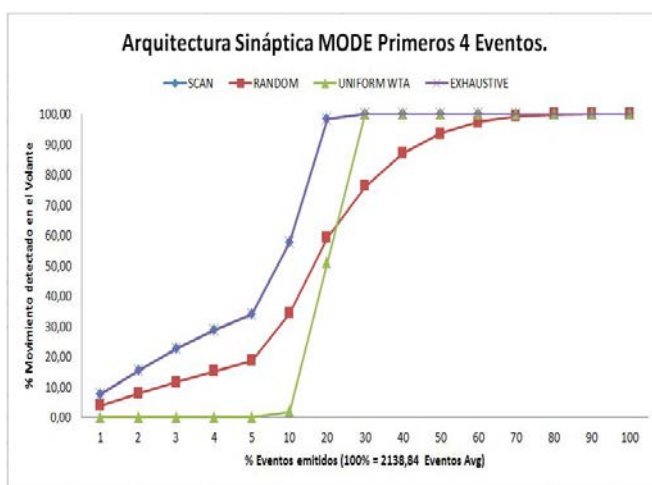


Fig. 42. Eval. de emisión y detección de la Act. Manos. Sináp. y MODE



NEURONAL – FD – PRIMEROS 4 EVENTOS				
% Movimiento Detectado en el VOLANTE				
% Emitido	SCAN	RANDOM	UNIFORM WTA	EXHAUSTIVE
1	2,66	2,64	2,44	2,66
2	6,35	6,27	5,71	6,35
3	9,95	9,70	8,90	9,95
4	13,58	13,11	12,16	13,58
5	18,11	17,25	16,34	18,11
10	38,05	34,06	34,83	38,05
20	81,89	63,98	77,37	81,89
30	100	79,26	98,57	100
40	100	89,30	99,52	100
50	100	95,23	99,57	100
60	100	98,21	99,66	100
100	100	99,48	99,66	100

Tabla 18. Eval. de la emisión y detección de la Act. Manos. Neuro. y FD

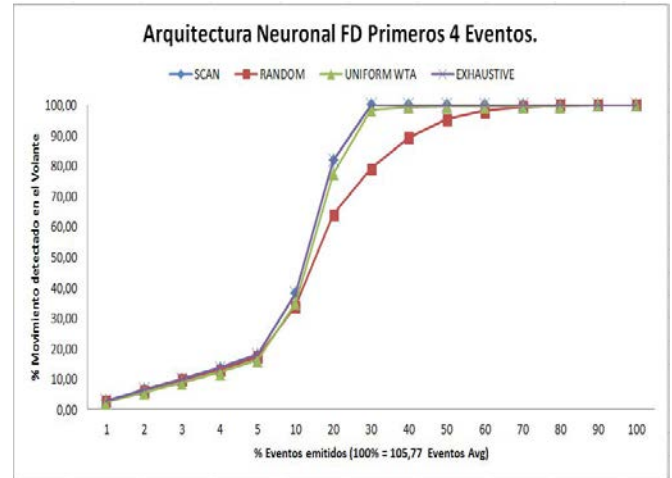


Fig. 43. Eval. de la emisión y detección de la Act. Manos. Neuro. y FD

NEURONAL – MEAN – PRIMEROS 4 EVENTOS				
% Movimiento Detectado en el VOLANTE				
% Emitido	SCAN	RANDOM	UNIFORM WTA	EXHAUSTIVE
1	3,99	3,95	3,27	3,99
2	8,03	7,84	6,56	8,03
3	11,99	11,53	9,96	11,99
4	16,02	15,16	13,54	16,02
5	19,96	18,61	17,15	19,96
10	40,00	34,51	36,45	40,00
20	80,04	59,28	76,46	80,04
30	100	76,18	99,99	100
40	100	87,27	100	100
50	100	93,90	100	100
60	100	97,52	100	100
100	100	99,25	100	100

Tabla 19. Eval. de emisión y detección de la Act. Manos. Neuro. y MODE

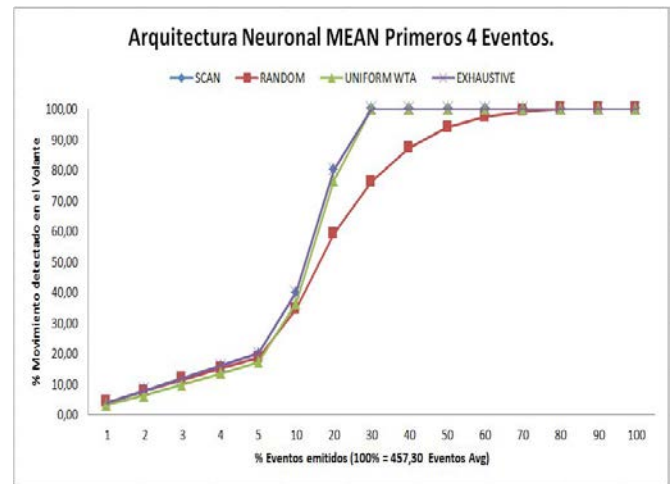


Fig. 44. Eval. de emisión y detección de la Act. Manos. Neuro. y MODE

NEURONAL – MODE – PRIMEROS 4 EVENTOS				
% Movimiento Detectado en el VOLANTE				
% Emitido	SCAN	RANDOM	UNIFORM WTA	EXHAUSTIVE
1	4,00	3,96	3,62	4,00
2	7,97	7,77	7,21	7,97
3	12,03	11,59	10,96	12,03
4	15,92	15,10	14,64	15,92
5	20,02	18,71	18,59	20,02
10	39,98	34,61	38,18	39,98
20	79,98	59,40	78,14	79,98
30	100	76,36	99,97	100
40	100	87,31	100	100
50	100	93,84	100	100
60	100	97,52	100	100
100	100	99,27	100	100

Tabla 20. Eval. de emisión y detección de la Act. Manos. Neuro. y MODE

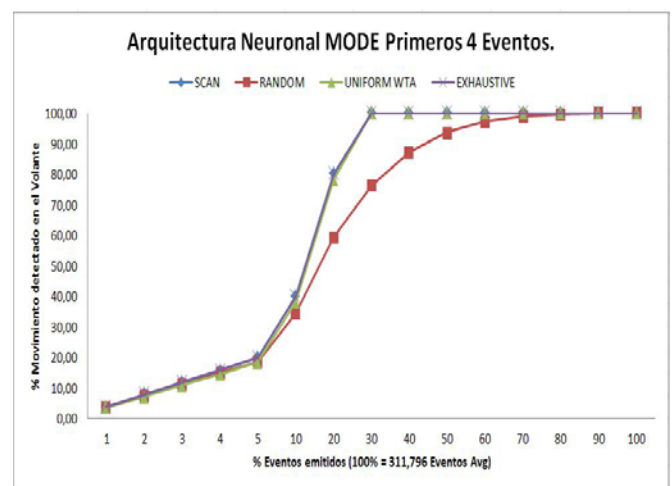


Fig. 45. Eval. de emisión y detección de la Act. Manos. Neuro. y MODE



	SINÁPTICA - PRIMEROS 4 EVENTOS					
	FD		MEAN		MODE	
	Tamaño Eventos Kb 50% ROI Volante	Tamaño Eventos Kb 100% ROI Volante	Tamaño Eventos Kb 50% ROI Volante	Tamaño Eventos Kb 100% ROI Volante	Tamaño Eventos Kb 50% ROI Volante	Tamaño Eventos Kb 100% ROI Volante
SCAN	0,1005	0,3113	0,3679	1,1024	0,3212	0,9533
RANDOM	0,1657	0,9339	0,7611	4,6487	0,6222	3,7090
UNIFORM WTA	0,1939	1,0366	0,9299	1,4331	0,7695	1,1749
EXHAUSTIVE	0,1005	0,3113	0,3679	1,1024	0,3212	0,9533

Tabla 21. Evaluación del tamaño de los datos emitidos para detectar el 50% o el 100% de la actividad en la ROI. Sináptico

	SINÁPTICA - PRIMEROS 4 EVENTOS		
	FD (JPG = 0,9477Kb)	MEAN (JPG = 0,9441Kb)	MODE (JPG = 0,9533Kb)
	% Movimiento Detectado al igualar JPG. ROI Volante	% Movimiento Detectado al igualar JPG. ROI Volante	% Movimiento Detectado al igualar JPG. ROI Volante
SCAN	100	95,82	100
RANDOM	100	58,54	67,13
UNIFORM WTA	100	50,36	76,88
EXHAUSTIVE	100	95,82	100

Tabla 22. Comparativa del sistema propuesto frente al JPG. Neuronal

	NEURONAL - PRIMEROS 4 EVENTOS					
	FD		MEAN		MODE	
	Tamaño Eventos Kb 50% ROI Volante	Tamaño Eventos Kb 100% ROI Volante	Tamaño Eventos Kb 50% ROI Volante	Tamaño Eventos Kb 100% ROI Volante	Tamaño Eventos Kb 50% ROI Volante	Tamaño Eventos Kb 100% ROI Volante
SCAN	0,0880	0,1760	0,4684	0,9442	0,3611	0,7222
RANDOM	0,1571	0,9076	0,8041	4,2895	0,6570	3,5181
UNIFORM WTA	0,1697	0,8396	0,8223	1,6253	0,6989	1,3439
EXHAUSTIVE	0,0880	0,1760	0,4684	0,9442	0,3611	0,7222

Tabla 23. Evaluación del tamaño de los datos emitidos para detectar el 50% o el 100% de la actividad en la ROI. Neuronal

	NEURONAL - PRIMEROS 4 EVENTOS		
	FD (JPG = 0,9477Kb)	MEAN (JPG = 0,9442Kb)	MODE (JPG = 0,9528Kb)
	% Movimiento Detectado al igualar JPG. ROI Volante	% Movimiento Detectado al igualar JPG. ROI Volante	% Movimiento Detectado al igualar JPG. ROI Volante
SCAN	100	100	100
RANDOM	100	58,73	67,58
UNIFORM WTA	100	52,01	77,51
EXHAUSTIVE	100	100	100

Tabla 24. Comparativa del sistema propuesto frente al JPG. Neuronal

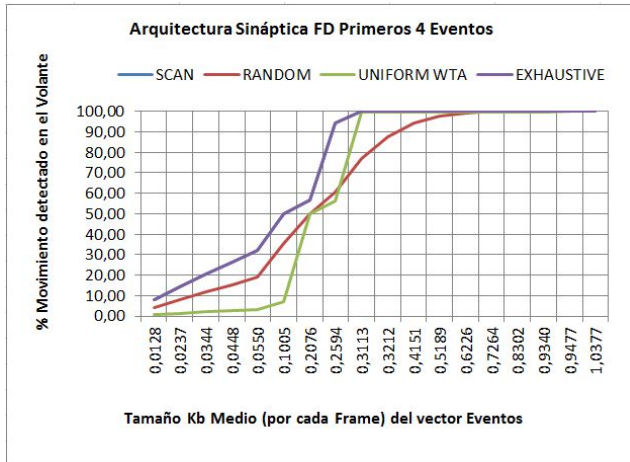


Fig. 46. Evaluación y comparativa de tamaño frente al JPG. Sináptica y FD

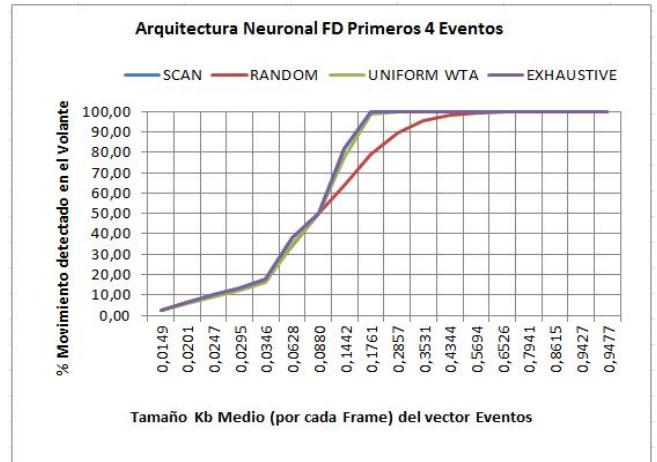


Fig. 47. Evaluación y comparativa de tamaño frente al JPG. Neuronal y FD

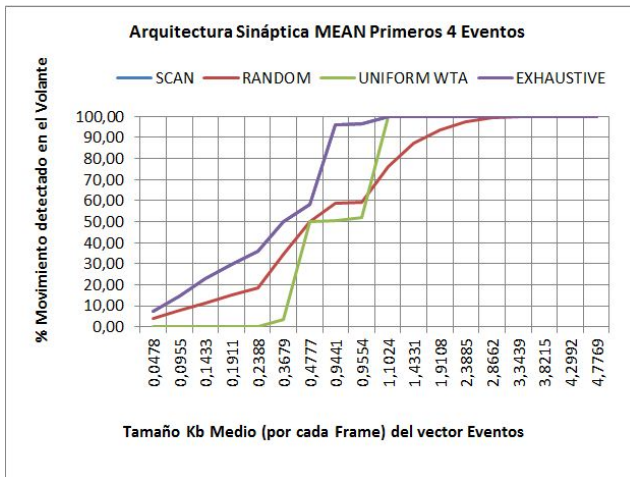


Fig. 48. Evaluación y comparativa de tamaño frente al JPG. Sináptica y MEAN

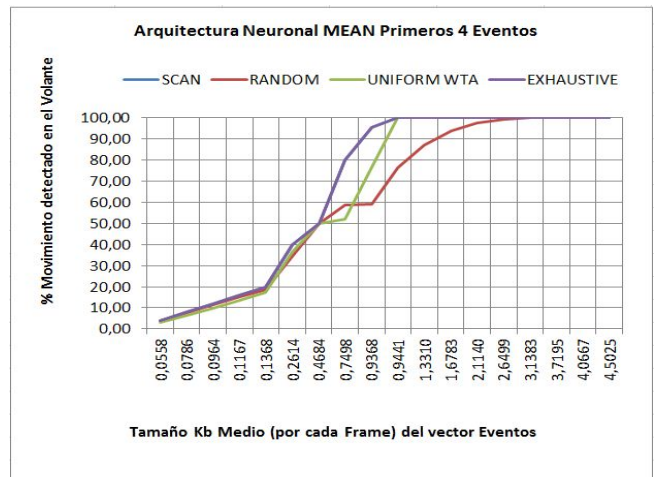


Fig. 49. Evaluación y comparativa de tamaño frente al JPG. Neuronal y MEAN

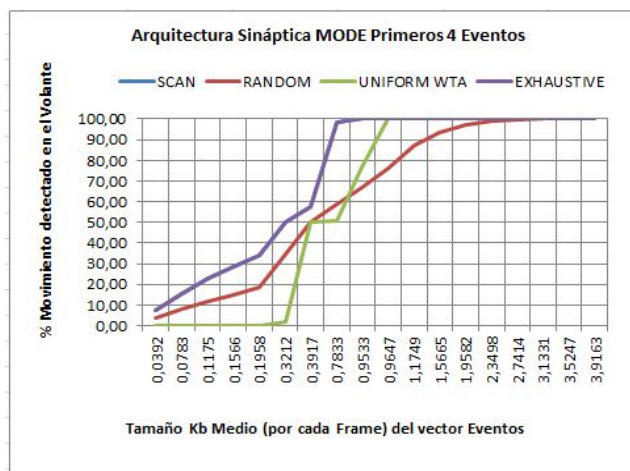


Fig. 50. Evaluación y comparativa de tamaño frente al JPG. Sináptica y MODE

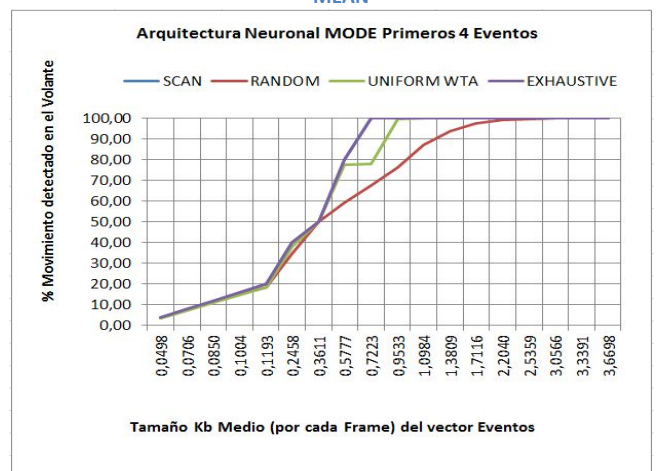


Fig. 51. Evaluación y comparativa de tamaño frente al JPG. Neuronal y MODE



Estos resultados son obtenidos al obtener como máximo, los 4 primeros eventos AER por cada píxel de la imagen, es decir, si el valor de intensidad del píxel supera el valor de 4, sólo se emitirá los 4 primeros eventos AER, en caso contrario, se emitirá tantos eventos como nivel de intensidad tenga el píxel, ya que no supera el umbral establecido. De esta manera reducimos la redundancia de información.

En este escalado seleccionando directamente los 4 primeros eventos, la arquitectura del sistema (Sináptica o Neuronal), tiene una mayor relevancia, mejorando la neuronal frente a la Sináptica.

Como era de esperar, el envío de información en el caso de la arquitectura Sináptica es mucho mayor que en la Neuronal, por la propia definición del sistema.

Como se observa en las gráficas, el rendimiento de los métodos de codificación AER para la arquitectura Neuronal es mucho mayor respecto a la arquitectura Sináptica.

En general, con este tipo de reducción de redundancia, el envío de información es un poco mayor al escalado, pero mejora de manera incremental el rendimiento del sistema.

Como puede observarse en las tablas Tabla 23 y Tabla 24, la configuración óptima del sistema se obtiene con la técnica de extracción de movimiento **Frame Difference** y el método de codificación **Scan**, ya que para alcanzar el 100% de actividad se requiere una menor cantidad de datos.

En el caso de la arquitectura Neuronal sólo con un **0,17 Kb** de tamaño, se detecta el **100%** de la actividad, reduciendo el tamaño respecto a la compresión JPG hasta un **81,42%**, manteniendo la misma cantidad de información relevante.



Capítulo 10

10 Conclusiones

Se ha trabajado en el entorno de los sistemas biológicamente inspirados basados en eventos AER, comparándolos con los sistemas convencionales basados en fotogramas con una comprensión común JPG. Se ha diseñado e implementado un sistema “**AER Imager**” capaz de codificar cualquier video convencional, tomado con cámaras comerciales, a eventos AER. Este sistema se acopla al software jAER específico para la monitorización y procesamiento de eventos, eliminando así la dependencia existente que hasta ahora había del hardware específico que requería dicha aplicación, como es la cámara **Retina Artificial**. Cualquier video común, grabado con cualquier cámara comercial, es posible su reproducción o procesamiento a través del software jAER, gracias al sistema desarrollado en este proyecto.

Como conclusión, se ha obtenido que las dos arquitecturas implementadas Neuronal y Sináptica, son determinantes sólo en el caso de usar un tipo de reducción de redundancia: la emisión de los 4 primeros eventos AER por cada píxel.

En el otro caso, escalando la imagen a 4 eventos AER como máximo, no influye ya que el propio escalado “altera” el orden de la información emitida, haciendo menos determinante la selección, antes del envío realizado por la arquitectura Neuronal.

Como se ha podido observar en los resultados obtenidos, la técnica de extracción de movimiento que presenta mayor eficacia y rendimiento es el **Frame Difference**. Esta técnica es más efectiva a la hora de detectar la actividad de las manos, en concreto, en el volante.

El método de codificación AER más eficiente para una aplicación como la presentada, es el **Scan**.

En general, seleccionando la configuración **Frame Difference** como técnica de extracción de movimiento, **Scan** como método de codificación AER y la arquitectura **Neuronal**, presenta un rendimiento mayor y una mejora respecto a un sistema clásico basado en fotogramas y comprensión **JPG**.

Llegando a alcanzar una reducción de tamaño expresado en Kb, de hasta un **81,42%** (*para la configuración de la arquitectura Neuronal, emitiendo los primeros 4 eventos AER y codificación Scan con la técnica Frame Difference*) respecto al tamaño que ocupa una imagen JPG, manteniendo la misma información relevante en ambas.

Se ha demostrado que los sistemas bio-inspirados basados en eventos AER, son más eficientes en cuanto a tamaño y porcentaje de información emitida, respecto a los sistemas clásicos basados en fotogramas JPG, obteniendo la misma información relevante pero con un tamaño de ocupación menor, para la aplicación de la detección de la actividad de un conductor en un entorno de automoción.



Bibliografía

- [1] Luis Alejandro Camuñas Mesa, “**Microchips convolucionadores AER para procesamiento asíncrono neocortical de información sensorial visual codificada en eventos**”, *Tesis doctoral dirigida por Dr. Bernabé Linares Barranco, Universidad de Sevilla, Marzo 2010.*
- [2] José-Antonio, Pérez-Carrasco, Carmen Serrano-Gotarredona, Begoña Acha-Piñero, Teresa Serrano-Gotarredona and Bernabé Linares-Barranco, “**Advanced Vision Processing Systems: Spike-Based Simulation and Processing**”, *Instituto de Microelectrónica de Sevilla (IMSE-CNM-CSIC), 2009.*
- [3] Rafael Paz-Vicente, Ángel Jiménez-Fernández, Alejandro Linares-Barranco, Gabriel Jiménez Moreno, Francisco Gómez-Rodríguez, Lourdes Miró-Amarante and Anton Civit-Ballcells , “**Image convolution using a probabilistic mapper on USB-AER board**”, *Circuits and Systems, 2008. ISCAS 2008. IEEE International Symposium on.*
- [4] A. Jiménez-Fernández, R. Paz-Vicente, M. Rivas, A. Linares-Barranco, G. Jiménez, A. Civit, “**AER-based robotic closed-loop control system**”, *IEEE 2008*
- [5] Patrick Lichtsteiner, Christoph Posch and Tobi Delbruck, “**A 128x128 120 dB 15 μ s Latency Asynchronous Temporal Contrast Vision Sensor**”, *IEEE Journal of Solid-State Circuits, Vol. 43, N°2, February 2008.*
- [6] Jorg Conradt, Raphael Berner, Matthew Cook, Tobi Delbruck, “**An Embedded AER Dynamic Vision Sensor for Low-Latency Pole Balancing**”, *IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops 2009.*
- [7] L. Miró-Amarante, A. Jiménez-Fernández, A. Linares-Barranco, F. Gómez-Rodríguez, R. Paz, G. Jiménez, A.Civit., R. Serrano-Gotarredona, “**LVDS Serial AER Link performance**”, *Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on.*
- [8] R. Berner, T. Delbruck, A. Civit-Balcells and A. Linares-Barranco, “**A 5 Meps \$100 USB2.0 Address-Event Monitor-Sequencer Interface**”, *Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on.*
- [9] Tobi Delbruck, “**Frame-free dynamic digital vision**”, *Proceedings of Intl. Symp. on Secure-Life Electronics, Advanced Electronics for Quality Life and Society, Univ. of Tokyo, 2008.*
- [10] Tobi Delbruck, “**Freeing vision from frames**”, *The Neuromorphic Engineer, May 2006.*



- [11] A. Jimenez-Fernandez, J.L. Fuentes-del-Bosh, R. Paz-Vicente, A. Linares-Barranco, G. Jiménez, “**Live Demonstration: Neuro-inspired system for realtime vision tilt correction**”, *Circuits and System (ISCAS), Proceedings of 2010 IEEE International Symposium on*.
- [12] Alfredo Restrepo Palacios y Javier Villegas Plazas, “**Evaluación del uso de arquitectura foveal en visión Estéreo**”, *Laboratorio de Señales, Dpt. Ing. Eléctica y Electrónica, Universidad de los Andes, Bogotá, Colombia*.
- [13] [Gabriel Peyré](#), *Ceremade, Université Paris-Dauphine*, Toolbox for Matlab.
- [14] Cristina Conde, Eduardo Orbe, Isaac Martin de Diego and Enrique Cabello, “**Event based visual codification in automotive environments**”, *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*.
- [15] Enrique Alegre Gutiérrez, “**Procesamiento digital de imagen: fundamentos y prácticas con MATLAB**”, *Universidad de León. Secretariado de Publicaciones y Medios Audiovisuales, 2004, ISBN: 8497730526*.
- [16] Bernabé Linares-Barranco, “**Spike-Based Vision Processing. Seeing without Frames**”, *Circuits and Systems, 2006. ISCAS 2007. IEEE International Symposium on*.
- [17] Raphael Berner, “**Highspeed USB2.0 AER Interfaces**”, *proyecto fin de master dirigido por Dr. Tobias Delbruck, Prof. Anton Civit Balcells y Dr. Alejandro Linares Barranco, Universidad de Sevilla e Instituto de Neuroinformática (UNI - ETH Zurich), Abril 2006*.
- [18] Alejandro Linares Barranco, “**Estudio y Evaluación de Interfaces para conexión de Sistemas Neuromórficos mediante Address-Event-Representation**”, *Tesis doctoral dirigida por Dr. Bernabé Linares Barranco y Gabriel Jiménez Moreno, Universidad de Sevilla, Abril 2003*.
- [19] Jayram Moorkanikara Nageswaran, Nikil Dutt, Yingxue Wang and Tobi Delbrueck, “**Computing Spike-based Convolutions on GPUs**”, *Circuits and Systems, 2009. ISCAS 2009. IEEE International Symposium on*.
- [20] Scott Smith, “**MATLAB Advanced GI Development**”, *Dog Ear Publishing, 2006, ISBN: 1598581813*.



Anexo A

Manual de Usuario de la Aplicación: **Video2AER**



Introducción:

La aplicación **video2AER** ha sido diseñada con el objetivo principal de poder transformar videos tomados con una cámara comercial a eventos AER.

La aplicación principal para el estudio y monitorización de eventos AER es el jAER, el cuál carecía de este módulo, importante si no se dispone de una cámara específica llamada **Retina** y todo el hardware que ello conlleva, o bien, si se dispone de videos previamente grabados con cámaras comerciales y se quiere realizar un estudio.

La Retina es una cámara que por el momento se encuentra en laboratorios universitarios y no ha sido comercializada. Tiene un coste elevado debido a su sensor especial (128x128), capaz de capturar únicamente cambios de luminosidad producidos por varios factores como la luz ambiente o el movimiento del objeto al que se le está grabando.

El jAER es una herramienta muy potente implementada en java, capaz de monitorizar y procesar eventos en tiempo real, es decir, conectado la cámara Retina al Pc transmite los eventos por la interfaz Usb mientras que la aplicación jAER es capaz de representar dichos eventos en pantalla, además de poder realizar operaciones de *convolución*, *tracking*, etc. si se desea, todo ello en tiempo real. Otra forma que permite, es realizar las mismas operaciones pero con un video generado con la Retina que ha sido grabado previamente, cargando a la aplicación el fichero binario generado.

Por lo que se ha desarrollado una aplicación capaz de transformar un conjunto de fotogramas o un video tomado con una cámara estándar o comercial, en un conjunto de eventos AER, empaquetados en un fichero con el formato específico del jAER (extensión *.aerdat* o *.dat*), para su posterior estudio dentro de la aplicación principal jAER.

La gran potencia de esta aplicación es poder transformar cualquier video en distintos formatos, generados por una cámara comercial (*a un bajo coste*) a eventos AER, para realizar posteriormente un estudio ya basado en eventos AER frente a los fotogramas, con la velocidad y ventajas que ello conlleva.

Esta aplicación permite generar el conjunto de eventos AER de distintas formas según las opciones escogidas, ya que una principal característica es que es muy parametrizable.

Además genera unas estadísticas en formato Excel para un estudio más detallado.



¿Qué es el jAER?

jAER es un proyecto libre implementado en java, se compone aproximadamente de 500 clases, y permite la monitorización y procesamiento de eventos a tiempo real en el Pc conectando una interfaz hardware como UsbAERmini2, UsbAERmapper y la retina DVS128 o bien, de forma *offline* con un fichero de eventos previamente grabado en formato binario **.aerdat** o **.dat**, entre otras muchas funciones. Al ser proyecto libre, se encuentra en la forja de código [SourceForge](#).

Ha sido creado por *Institute for Neuroinformatics* (INI) de la Universidad de Zurich (Suiza). Las principales entidades colaboradoras de este proyecto son Universidad de Sevilla y ETH de Zurich entre otras.

La estructura principal del jAER se divide en cuatro grandes bloques:

- **Filtros:** son procesamiento de eventos a tiempo real, la retina envía un paquete de eventos con la dirección del píxel que emitió el impulso. Al aplicar un filtro, captura el paquete que ha llegado, procesa los eventos realizando unas operaciones matemáticas y devuelve el paquete ya procesado. Transcurre todo ello a tiempo real. Una característica importante de los filtros que son muy parametrizables.
Hay distintos filtros creados por distintas entidades, los más destacables son *Background Activity Filter* (Elimina el ruido de fondo que captura la retina), *Rotate Filter* (Invierte las posiciones de representación por pantalla), *Rectangular Cluster Tracker* (Seguimiento de objetos), etc.
- **AE Chips:** Hay una gran lista de chips creados por distintas entidades y para distintas finalidades. Lo que diferencia una clase chip de otra, es la forma de tratar y leer los eventos. La clase chip estándar es DVS128 donde el direccionamiento AER es de 16 bits. Los distintos chips leen de distinta forma el direccionamiento, bien por que el problema lo requiere (se necesita ocupar el bit N.C, por ejemplo para retinas en estéreo), el diseño de las placas ha sido modificado, se quiere tratar direcciones en vez de 16 bits a 32 bits, etc.

N.C	POS. Y	POS. X	POLARITY
1 bit	7 bits	7 bits	1 bit

← 16 bits →

Fig 1. Distribución de los bits del AER



- Grabación de eventos: Además de monitorizar los eventos generados por los distintos sistemas basados en AER, puede almacenarlos en un fichero binario .aerdat o .dat. Es interesante esta idea, si se quiere guardar lo que está capturando la retina para posteriormente procesar o reproducir dichos eventos de forma “*offline*”, permitiendo trabajar con el fichero sin disponer de la cámara Retina y todo el sistema hardware.

- Reproductor del fichero .aerdat: Dispone de botones para el control de reproducción (play, fw, rw, pause, ...) y cajas de texto donde se muestra el tiempo en microsegundos y el número de eventos que se está leyendo. Se puede acotar la reproducción por número de eventos, por franja de tiempo (expresado en microsegundos) o bien según se captura, reproduce en vivo, con la opción *real time playback*. Dispone de la opción de grabar la monitorización de los eventos en una secuencia de frames.

Esta última característica es la más importante para nuestra aplicación, ya que nuestro principal objetivo es transformar un video convencional a un fichero con formato *aerdat* para reproducirlo en el jAER.

El fichero *aerdat* es un fichero binario que está compuesto de una cabecera (máx. dos líneas), indicando al jAER si los eventos están expresados con un direccionamiento de 16 bits (en el caso común, tamaño máximo de 128x128) o de 32 bits (si en un futuro se mejora el sensor de la cámara Retina).

A continuación le sigue los eventos AER expresados como número entero de 16 bits como resultado de transformar el evento AER de binario a decimal, intercalándose con su *Timestamp* expresado en microsegundos y con un ancho de 32 bits.



Requisitos e Instalación:

Requisitos de la aplicación:

La aplicación está implementada en Matlab y puede ser ejecutada bajo sistemas operativos Windows y Linux, aunque se recomienda Windows, ya que una parte de ella, el selector de regiones de interés o *ROIs* está desarrollada en C++ con librerías incluidas para Windows.

Los métodos y datos almacenados durante la ejecución de la aplicación, y por tanto son datos que residen en memoria, han sido totalmente optimizados para aprovechar el menor porcentaje de uso de memoria RAM, ajustando la reserva de espacio al menor tipo siempre que fuera posible (Ej. *uint8*).

Se recomienda si se desea transformar videos o una carpeta que contiene una secuencia de imágenes, a un tamaño superior a lo permitido por el software jAER (128x128 px.) o bien, si el número de imágenes o fotogramas que componen el vídeo es elevado (superior a 5000 fotogramas), que se utilice un sistema operativo de 64 bits, ya que la gestión de memoria varía entre las dos versiones, pudiendo obtener más espacio en memoria si la versión es de 64 bits.

El rendimiento en lo que tiempo se refiere no varía de una versión a otra, únicamente en almacenamiento temporal de datos en memoria.

Por tanto es recomendable el uso de mayor capacidad de memoria RAM o un S.O de 64 bits si el número de fotogramas o duración del video es elevado o bien, la resolución de la imagen es grande.

Instalación:

Hay dos principales maneras para la instalación de la aplicación dependiendo el usuario final que lo maneje.

Usuarios desarrolladores:

Toda la aplicación ha sido desarrollada en Matlab exceptuando el selector de Regiones de interés que ha sido implementada en C++. Por tanto, es necesario tener instalado el entorno de desarrollo de Matlab (*Se recomienda al última versión*).



Una vez abierto el entorno de programación Matlab, se abre el directorio del proyecto que contiene todo el código dividido en sub carpetas según la funcionalidad que realiza, ya que está modularizado. Se ejecuta el fichero principal o *main* del programa, ***generateEvents.m***.

Esta manera está enfocada para usuarios desarrolladores que quieran realizar cambios o mejoras a la aplicación.

Usuarios o Viewers:

Está diseñado para usuarios que solo quieren ejecutar la aplicación y obtener los resultados de ello. Se proporciona un paquete instalador que instala de forma automática todo lo necesario para que el usuario final pueda utilizar la aplicación.

Se ejecuta el paquete ***Video2AER-pkg.exe*** que instala el MCR (Matlab Component Runtime) si no está instalado previamente o nunca se ha instalado Matlab en el Pc es necesario su instalación.

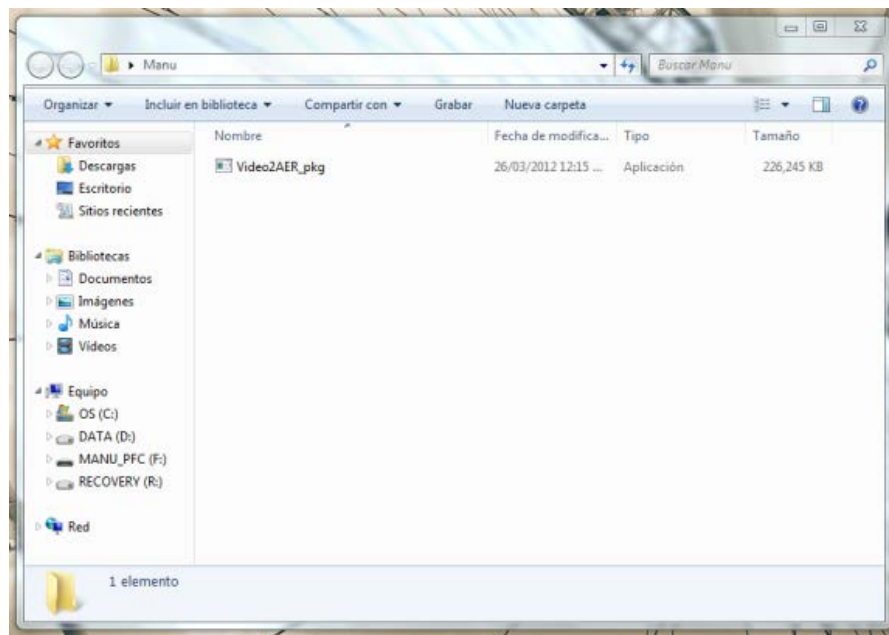


Fig 2. Paquete de instalación

Se inicia un proceso *batch* en el que mostrará el configurador de instalación del MCR. A continuación se mostrará el proceso mediante ilustraciones orientativas.

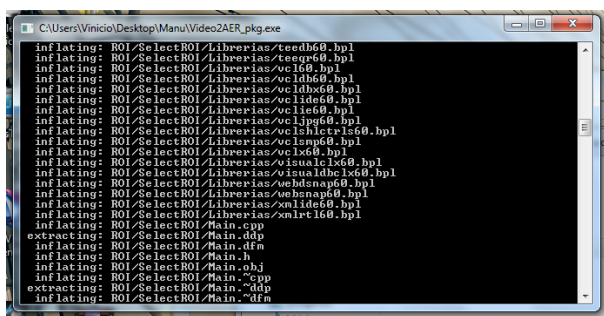


Fig 3. Proceso Batch de instalación

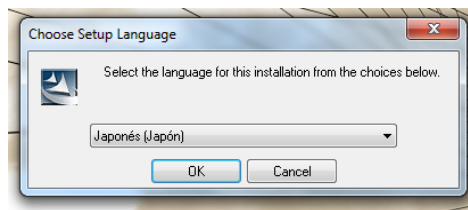


Fig 4. Selección de idioma

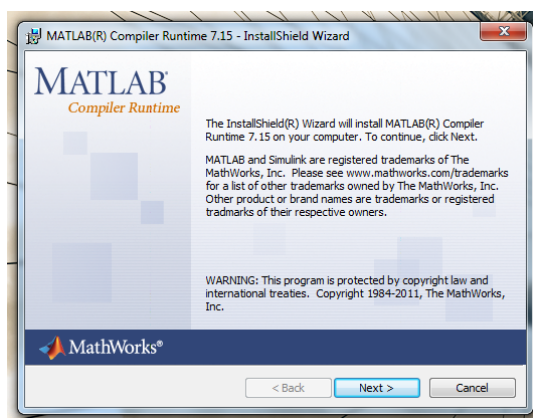


Fig 5. Instalación MCR

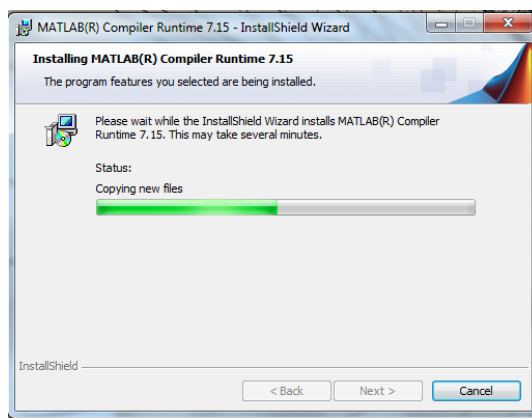


Fig 6. Proceso de instalación MCR

Una vez terminado la instalación del MCR se crearán una serie de archivos necesarios para el correcto funcionamiento de la aplicación, dentro del directorio donde se ha ejecutado el paquete de instalación. Contiene una serie de carpetas llamadas *Librerías*, *Result*, *ROI* y dos archivos uno llamado *ROIDemo.txt* y *Video2AER.exe*, este último es el ejecutable de la aplicación.

- **Librerías:** Esta carpeta contiene todas las librerías necesarias (.dll) para que el selector de ROI pueda ejecutarse con normalidad. Si no se disponen previamente de estas librerías se han de mover al directorio por defecto de Windows *C:\Windows\System32*, donde se encuentran las demás librerías de otras aplicaciones.
- **Result:** Esta carpeta se crea vacía para almacenar todos los resultados y salidas generados por la aplicación.
- **ROI:** Esta carpeta contiene el programa para elegir las Regiones de interés y generar un archivo de texto plano con las coordenadas de dichas regiones. Este archivo es idéntico al que se incluye como ejemplo, llamado *ROIDemo.txt*.

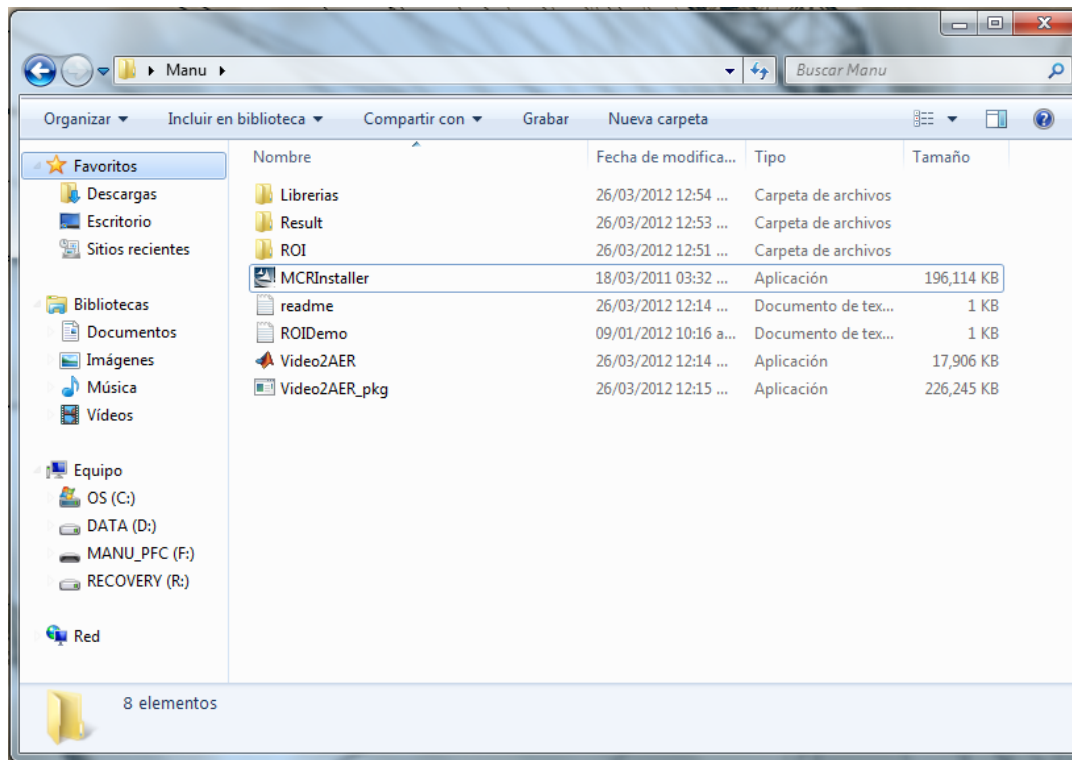


Fig 3. Resultado de instalación

Algún posible error de ejecución se solicita que el usuario revise los permisos de los directorios o bien, se ejecute la aplicación en modo administrador

click derecho del ratón → propiedades → Compatibilidad → check Ejecutar este programa en modo compatibilidad para: → Windows 7 → Aceptar.

Funcionalidad:

A continuación se va explicar punto a punto los pasos que sigue la aplicación como así también los diferentes parámetros y su funcionalidad con el fin de obtener un resultado final ajustado al propósito del usuario que lo utilice.

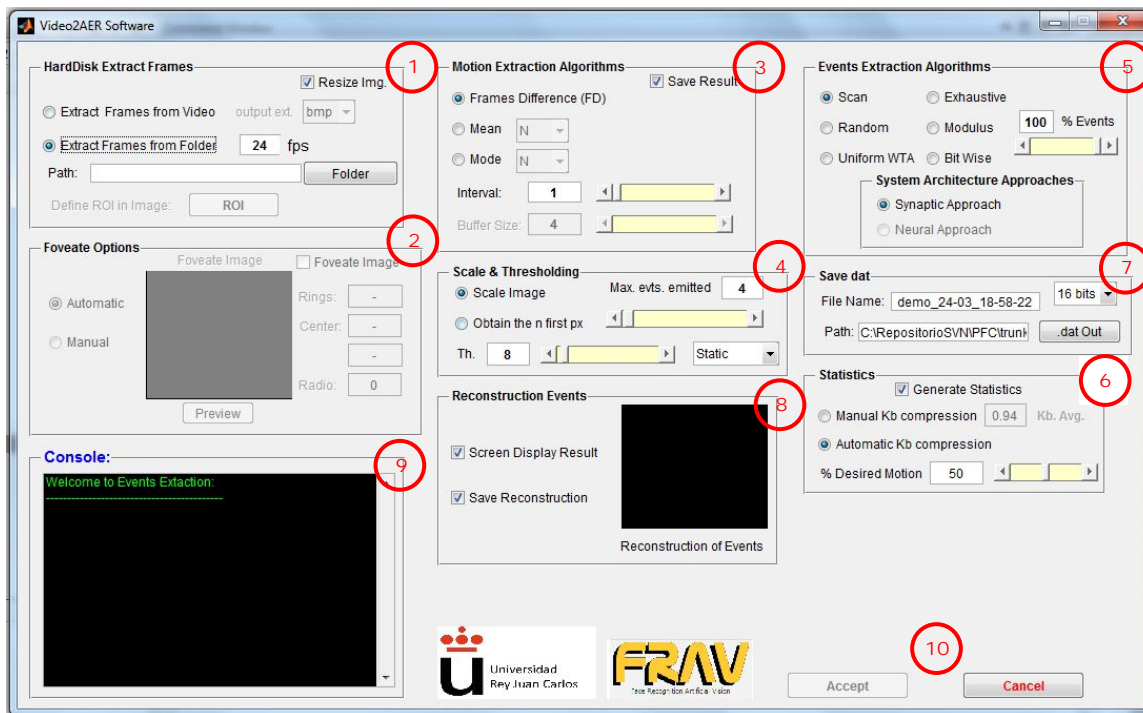


Fig 4. Aplicación principal Video2AER

- | | | | |
|---|--|----|--------------------------------------|
| 1 | Extracción de fotogramas y redimensionamiento. | 6 | Generador de estadísticas. |
| 2 | Ajuste Foveado de imágenes. | 7 | Salida fichero <i>aerdat</i> . |
| 3 | Extracción de movimiento. | 8 | Visualizador de eventos resultantes. |
| 4 | Reducción de ruido y redundancia. | 9 | Consola de salida de la aplicación. |
| 5 | Extracción de fotogramas y redimensionamiento. | 10 | Botón de comienzo de la ejecución. |

Extracción de los fotogramas de un video digital:

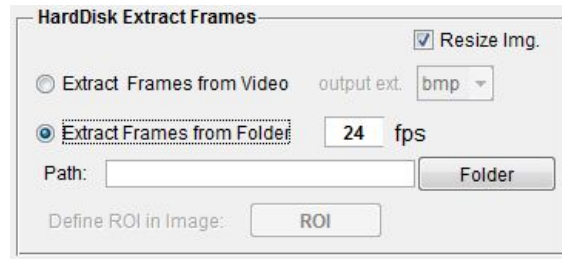


Fig 5. Extracción de Frames

La funcionalidad principal es la de extraer los fotogramas de un video y redimensionarlos al tamaño máximo permitido por la aplicación jAER y transformarlos a escala de grises, todo ello si fuera necesario.

El checkbox **Resize Img.**⁵ si está activado, redimensionará la imagen a un tamaño permitido por el jAER(128x128 como máx.) conservando las dimensiones. Se reducirá en el caso de que el tamaño original supere al máximo permitido por el jAER. Ejemplo: disponemos de una imagen de tamaño 480x640 px. ésta se reducirá a un tamaño de 96x128 respectivamente, respetando las limitaciones impuestas por el jAER y conservando las proporciones de resolución. Posteriormente si las imágenes están en RGB serán transformadas automáticamente a escala de grises, siendo el mínimo valor de intensidad 0 y el máximo 255.

Hay dos formas de cargar el origen de entrada dependiendo del combobox seleccionado.

- **Video:** Se elige un video en formato *.avi .mj2 .wmv .asf .asx .mp4 .m4v .mov .mpg* y se especifica que formato de compresión de imagen (*.bmp, .jpg, .png, .tiff*) se desea para la salida final de los frames . Automáticamente detecta el número de frames que lo componen, la resolución y la velocidad del video (fps).
- **Carpeta de fotogramas:** Esta opción es necesaria si solamente se dispone de un conjunto de fotogramas que componen el video. Para ello se selecciona el directorio donde se encuentra los fotogramas y se especifica la velocidad a la que iría dicho video en fps.

⁵ Si este *checkbox* no es activado automáticamente la salida del fichero final **aerdat** tomará **32 bits** como ancho de direccionamiento.

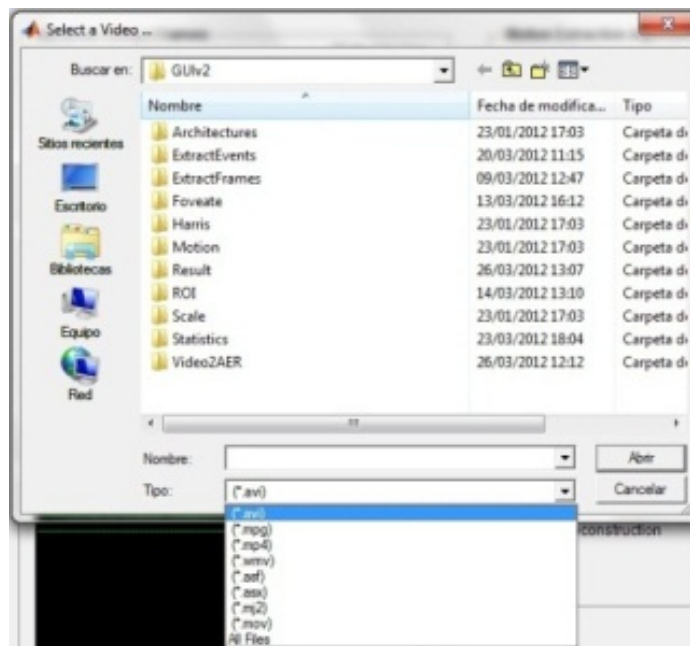


Fig 6. Selección de Video

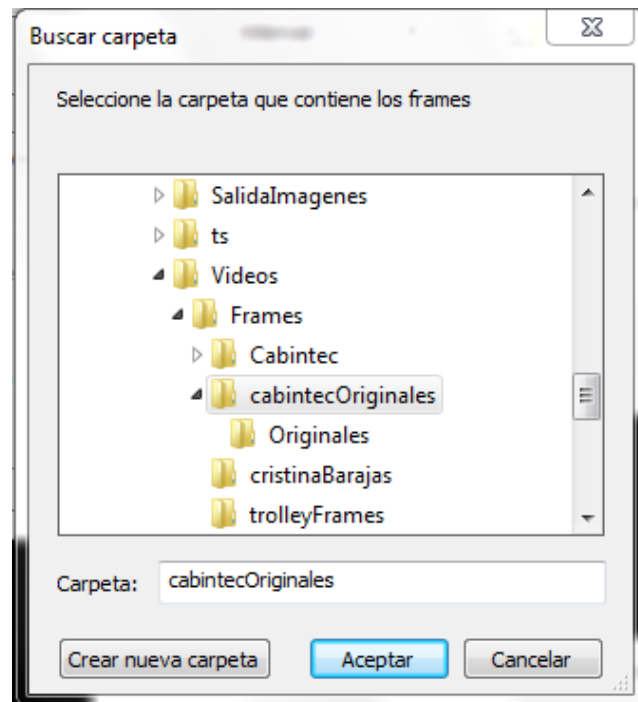


Fig 7. Selección de directorio de Fotogramas

Selector de ROIs:



Fig 8. Botón ROI

Este botón es activado cuando la operación de extracción de fotogramas de un video ha finalizado correctamente. Se inicia una interfaz que permite elegir regiones de interés mediante figuras elípticas y rectangulares. Se muestra la primera imagen del video o directorio de fotogramas seleccionado en el paso anterior, para poder seleccionar las regiones de dicha imagen. Si se desea, se puede cargar cualquier otra imagen.

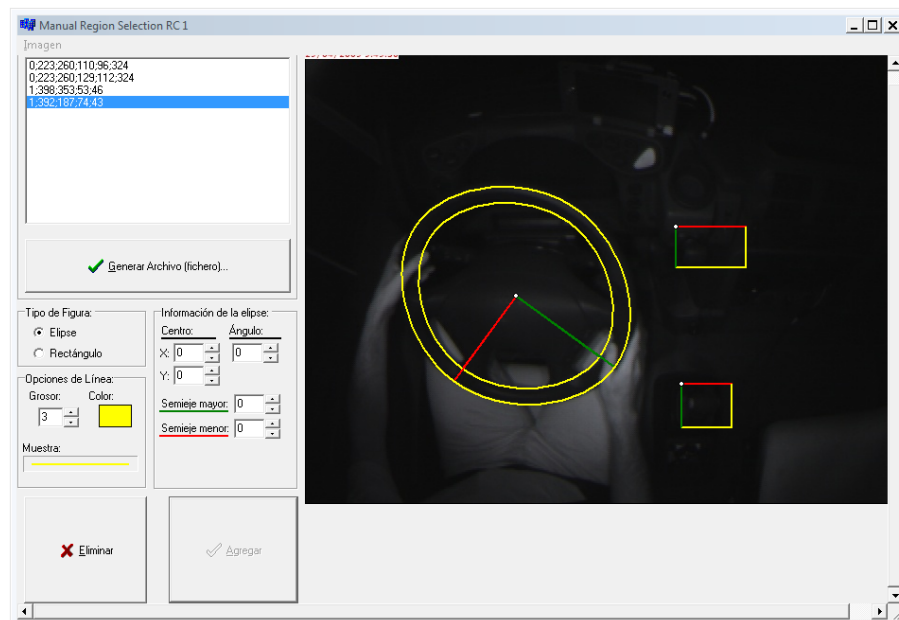


Fig 9. Interfaz de selección de ROI

Se dibuja tantas figuras como sean necesarias con forma elíptica o rectangular dependiendo de que zona queramos estudiar y que figura se adapte mejor.

En primer lugar se dibuja con el ratón la figura escogida en la zona de la imagen deseada. Una vez hecho esto, se puede hacer un “*ajuste fino*” cambiando las coordenadas de posiciones de las figuras con las cajas de texto proporcionadas por la aplicación. Definida ya la ROI se procede a pulsar el botón de **Agregar** para fija dicha figura.

Este proceso se repite tantas veces como regiones se quieran definir. Cuando se definen dos regiones iguales concéntricas, se tomará como una sola figura, es decir, las figuras concéntricas (elípticas o rectangulares) su región de interés es el espacio que hay entre la figura de menor tamaño con la de mayor tamaño, como se muestra en la ilustración.



Una vez finalizado y seleccionado todas las regiones deseadas, se pulsa el botón **Generar Archivo**, la aplicación automáticamente generará un archivo de texto plano con el nombre y ruta especificada por el usuario. Este fichero contiene como primera línea el número de figuras seleccionadas y a continuación un *0* indicando que la figura que sigue es una elipse o un *1* si es un rectángulo seguido de las coordenadas en píxeles de la figura.

```
1 4
2 0;223;260;110;96;324
3 0;223;260;129;112;324
4 1;398;353;46;53;0
5 1;392;187;43;74;0
```

Fig 10. Fichero generado por la Interfaz ROI

Posteriormente se solicitará que se cargue dicho fichero generado para que la aplicación principal pueda hacer operaciones internas.

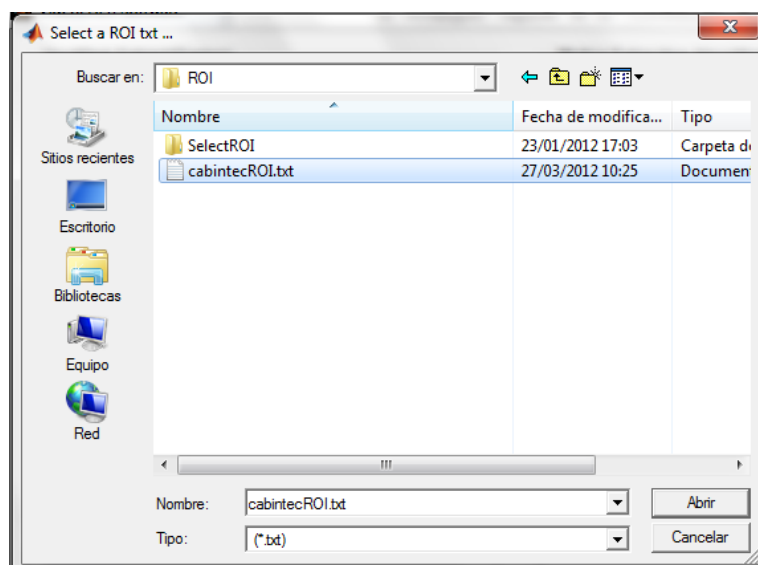


Fig 11. Selección del fichero generado por la Interfaz ROI

Fovear una imagen:



Fig 12. Foveado de una imagen

Se presenta una interfaz para manejar los distintos parámetros necesarios para realizar un “foveado” de la imagen. Si se desea realizar el proceso de foveado para todos los fotogramas se hace check en Foveate Image.

A continuación se describe los parámetros importantes para realizar el foveado y son los **Anillos** o **Rings**, siendo el número de difuminado o *blurring* que se realizará a la imagen. Se recomienda un valor superior a 15 si se desea realizar un foveado “suave” y un valor entorno al 4 para un foveado “acentuado” en el cual se diferenciarán claramente cada anillo.

El **centro** de la **Fóvea** y su **Radio**, se indica que posición en X e Y se situará el centro de la fóvea con un radio, todo ello en píxeles. La Fóvea es la parte que no se aplicará difuminado y quedará exactamente igual que en la imagen original.

La imagen modelo que se tomará al igual que en el selector de ROIs será la primera imagen de la toma de fotogramas. Todos los parámetros escogidos para esta imagen serán aplicados a todos los fotogramas por igual.

Hay dos formas de poder ajustar estos parámetros y son:

- **Automática:** El centro y radio de la fóvea se obtiene de forma automática de la región de interés elíptica más pequeña de todas las seleccionadas, en el caso de que se halla seleccionado ROI/s, en caso contrario, se tomará como fóvea el centro de la imagen y un radio de 15 píxeles.
- **Manual:** Se activará un botón al hacer *click* se activará una interfaz que permite al usuario poder elegir el centro y radio de la fóvea de forma visual con el *mouse*.

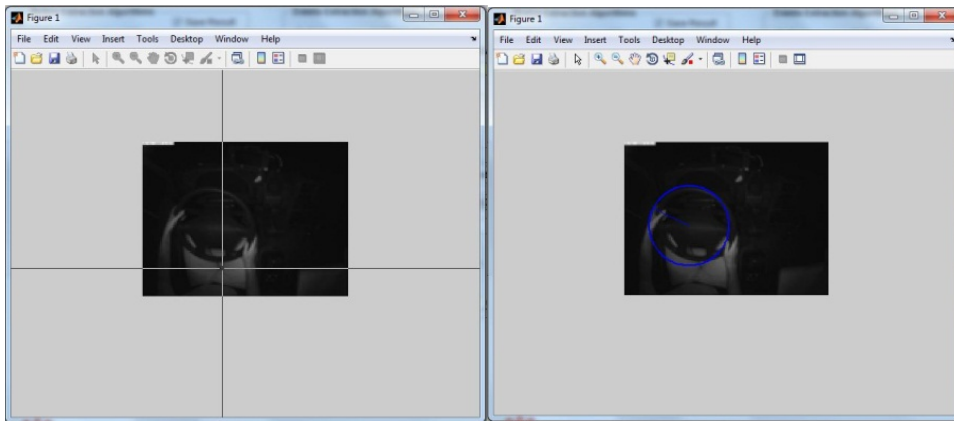
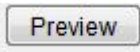



Fig 13. Selección de la Fóvea

Una vez seleccionado, se cierra la ventana y los parámetros quedarán reflejados en las cajas de texto correspondientes.

Se podrá hacer una pre-visualización para ambos casos (automático y manual) pulsando el botón . Mostrará en el cuadro de visualización el resultado con los parámetros introducidos. Si se desea ver en otra ventana o ampliado solo hace falta pulsar el botón de ampliar  y mostrará una ventana con las dos imágenes para poder ver el resultado de forma más detallada.

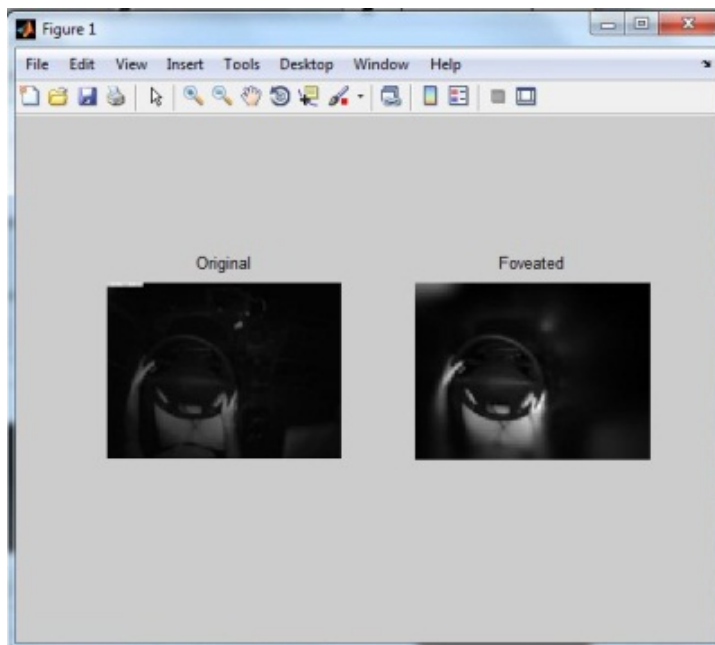


Fig 14. Resultado del Foveado

Extracción de movimiento:

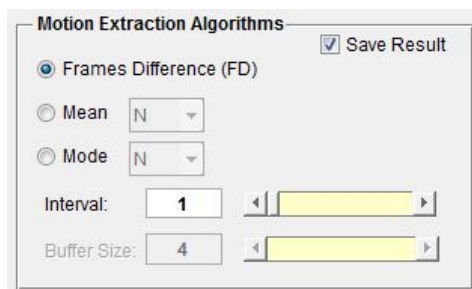
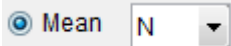


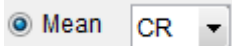
Fig 15. Algoritmos de extracción de movimiento

Se presentan los tres algoritmos utilizados para la extracción de movimiento en los frames.

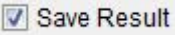
El algoritmo **FD** solo tiene un parámetro a definir, el *intervalo* desde 1 a *máximo de fotogramas extraído*. Se recomienda que este valor sea 1.

Tanto el algoritmo **Mean** como el **Mode** tienen la opción de *intervalo* y *buffer* con un rango desde 1 a *máximo de fotogramas extraído*. El *buffer* indica el tamaño de imágenes que se va a tomar para generar la imagen fondo o modelo.

Se ha diseñado dos formas distintas de refrescar el buffer para generar la imagen fondo, la *Normal*  en el que el buffer sigue una política de replazo FIFO. Cuando el intervalo indica el momento de refresco del buffer, la imagen más antigua de este es remplazada por la nueva y genera la nueva imagen fondo continuando con el proceso.

En cambio la forma *Refresco Continuo (CR)*  la única diferencia es que en vez de remplazar únicamente la imagen más antigua, remplaza todas las contenidas en el buffer, obteniendo así una nueva imagen fondo más actual.

Se recomienda un intervalo no demasiado elevado y un buffer amplio dependiendo todo ello del número de frames que componen el video y de la experiencia de los resultados previos obtenidos.

El checkbox  ⁶ indica si se quiere almacenar en disco las imágenes resultantes de realizar la extracción de movimiento. Se almacenará en el directorio *Result* → *Nombe_Video* → *F_interval* o *Mean_interval_buffer* o *Mode_interval_buffer*.

⁶ Esta opción ha de estar **activada** si se desea realizar estadísticas automáticas en Excel de los resultados. Necesario para obtener los datos necesarios para obtener una compresión igual a la que realiza un **jpg**.



Reducción de ruido y redundancia:

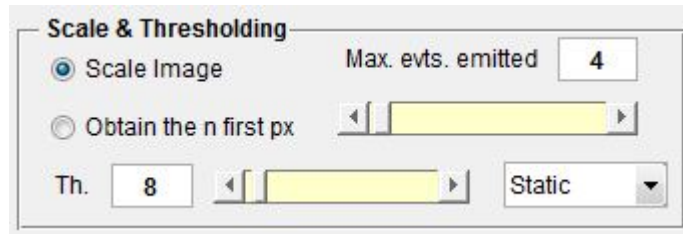


Fig 16. Reducción de Ruido y Redundancia

Este módulo se encarga de reducir el ruido producido por el proceso anterior y la redundancia de información a procesar, es decir, limitamos el número de eventos AER máximo que se va emitir.

Para reducir la redundancia se ha implementado dos mecanismos:

- **Escalando la imagen:** Los niveles de intensidad de la imagen resultante de la extracción de movimiento es escalado a valores comprendidos entre 0 y N eventos emitidos que se hallan escrito en la correspondiente caja de texto. Max. evts. emitted 4
Por ejemplo, valores comprendidos entre 0 y 255, tomarán ahora valores entre 0 y 4 para este caso.
- **Obteniendo los N primeros:** Aquí son emitidos los primeros N niveles de intensidad del píxel cuando este supera a N . Es decir, si por ejemplo en el primer píxel tiene un nivel de intensidad de 87, será emitido como máximo 4 eventos de esa misma dirección, por el contrario si su nivel es de 3 serán enviado esos 3 eventos ya que no supera al umbral máximo impuesto por el usuario.

Por defecto el valor de **Máximos eventos emitidos** es 4 ya que se quiere emular a la cámara Retina de la forma más realista y ésta envía solamente 4 eventos como máximo de una misma dirección del píxel que ha sufrido variación de luminosidad.

Con estas dos técnicas se reduce la redundancia ya que para píxeles con niveles de intensidad elevados no inundan el vector final de eventos con su dirección, reduciendo así el tamaño del vector.

La extracción de movimiento genera un pequeño ruido en la imagen, dependiendo de los parámetros seleccionados puede incrementar este ruido. Para ello se ha diseñado dos mecanismos para la reducción de ruido.



- **Estático:** Se establece de manera manual un umbral fijo que se aplicará a todas las imágenes que componen el video. Todos los valores de intensidad que **no** superen este umbral tomarán el valor de 0, ya que será considerado ruido.



- **Dinámico:** Automáticamente para cada fotograma independientemente, se estudia que valores bajos de intensidad son repetidos con mayor frecuencia. Estos tomarán el valor de 0, ya que como en el anterior caso serán considerados ruido.



Algoritmos de conversión a Eventos AER:

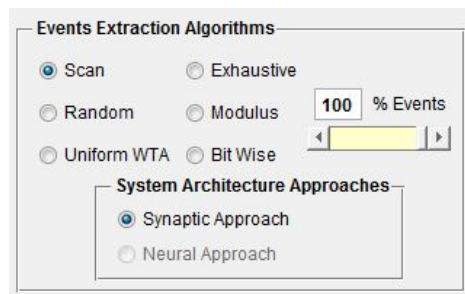


Fig 17. Algoritmos de generación de eventos AER

Se definen seis algoritmos capaces de transformar imágenes a eventos AER. La diferencia entre ellos es la distribución de eventos AER dentro del vector final.

Se define que porcentaje de eventos AER se desea emitir por cada fotograma que compone el video digital, con un rango de 1% a 100% sobre el máximo de eventos AER que se pueden emitir.

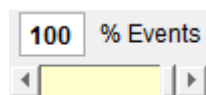


Fig 18. Porcentaje de eventos AER a emitir



Arquitecturas de la forma de envío:

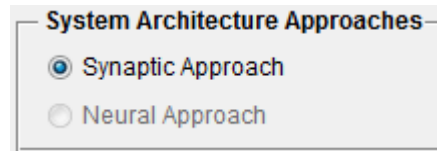


Fig 19. Arquitecturas Sináptica y Neuronal

Se presentan dos enfoques a la hora de enviar los eventos.

- **Sináptica:** Todos los eventos que se han extraído de los fotogramas son empaquetados en un fichero binario con extensión *aerdat* específico del jAER. Este fichero contiene tanto eventos pertenecientes a la/s ROI/s como eventos que no pertenecen, es decir, esta arquitectura recoge todos los eventos extraídos de la imagen.
- **Neuronal:** A diferencia de la anterior arquitectura, esta únicamente emite eventos que se encuentran dentro de la/s ROI/s definidas. Por lo que si no se ha definido ninguna región de interés, esta opción **no** estará disponible. El fichero binario final tendrá menor tamaño que el de la anterior arquitectura ya que se reduce el número de eventos, concentrando solo así eventos que pertenezcan a la ROI/s.

Generación del fichero binario aerdat:

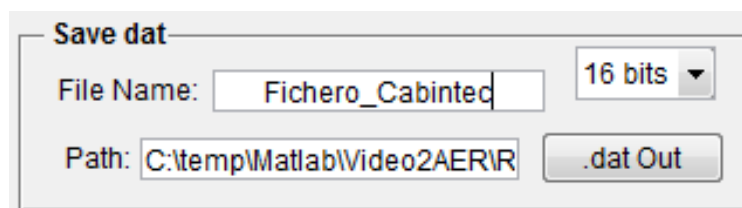


Fig 20. Opciones fichero aerdat

Se define un nombre que tomará el fichero binario y el directorio destino. Por defecto el resultado será almacenado en el directorio actual, en la carpeta Result.

El jAER es capaz de leer ficheros con extensión *.aerdat* como *.dat*, entre ellos no hay ninguna diferencia.



El combobox indica que ancho de banda tendrá las direcciones de los eventos AER. Este valor ha de ser de *16 bits* si el tamaño de imagen es menor o igual a 128x128 px. si excede se ha de cambiar al valor de *32 bits*, pero no se podrá visualizar aún en el jAER ya que no lo permite por el momento. Como ya se comentó en la primera parte, este valor será automáticamente actualizado si **no** se activa el checkbox **Resize Img.**, tomando el valor de *32 bits*.

Visualización y Reconstrucción de eventos AER:

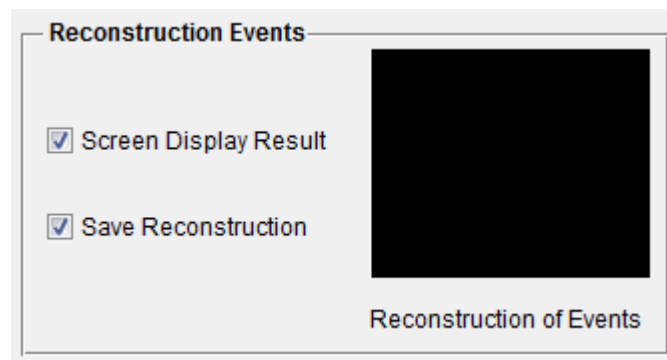


Fig 21. Visualizador de eventos AER

Este módulo se encarga de reconstruir los eventos AER generados a partir de los fotogramas.

Una vez realizado todo el proceso, el resultado puede ser mostrado a modo de pre visualización si el usuario lo desea, para ello se debe activar el checkbox **Screen Display Result**. Además si se desea conservar el resultado de la reconstrucción se ha de activar el checkbox **Save Reconstruction** para que sea almacenado en disco.

La reconstrucción de los eventos en el pre-visualizador no se reproduce a la misma velocidad (fps) que en el video original, ya que se ha añadido un retardo entre fotograma y fotograma para así poder apreciar mejor el resultado.



Generación de estadísticas:

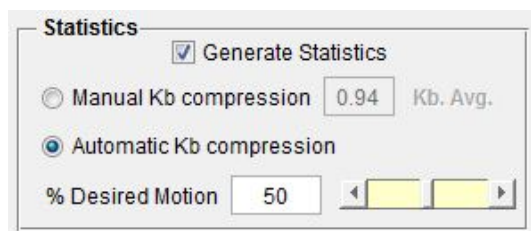


Fig 22. Generador de estadísticas

Si se desea realizar las estadísticas se ha de activar el check **Generate Statistics**. Estas estadísticas generan un fichero Excel en el cual se imprime toda la información extraída a la hora de extraer los eventos AER de los fotogramas.

Estos datos son *cantidad de eventos totales extraídos del fotograma, cantidad de eventos que pertenecen alguna región de interés seleccionada por el usuario, número de píxeles en la imagen y en la ROI, tamaño en Kb que ocupa los eventos*, etc. todo ello con su desviación típica ya que los datos mostrados son en media.

Se realiza automáticamente para un intervalo de porcentaje desde *1%* a *100%* con un incremento del *10%* en cada iteración. Además se realiza estadísticas tanto para la arquitectura sináptica como la neuronal en una sola ejecución, independientemente de las opciones elegidas por el usuario.

Se ha optado por este diseño, para disminuir tiempo y número de ejecuciones para obtener resultados.

Estas estadísticas proporcionan al usuario una estimación media de **cuánto porcentaje** total de eventos se necesitaría emitir por el bus AER para obtener un porcentaje de movimiento en la o las regiones de interés dibujadas y para obtener un tamaño del vector de eventos AER final cuyo tamaño expresado en Kb sea igual a la media que ocupa cada imagen de movimiento en disco con una compresión **JPG**.

En primer lugar para obtener un tamaño en Kb del vector de eventos AER igual a la media obtenida de lo que ocuparía en Kb las imágenes movimiento con una compresión JPG, hay dos formas de realizarlo:

- Automática: Es la recomendable, ya que calcula automáticamente cuánto ocupa en disco, expresado en Kb, cada fotograma del video. Posteriormente realiza una media y este dato será el máximo tamaño en Kb que debe tomar el vector final de eventos AER para que su tamaño sea igual a la media que ocupa la compresión JPG y así poder estudiar cuanto movimiento se ha detectado en la o las regiones de interés..

Automatic Kb compression



- Manual: Se especifica de forma manual el máximo tamaño en Kb que debe tomar el vector de eventos AER. Obteniendo como resultado la cantidad de movimiento detectado en la o las regiones de interés con dicho tamaño como máximo.

Manual Kb compression Kb. Avg.

En segundo lugar se define de forma manual el porcentaje total de movimiento que se desea detectar en la o las regiones de interés , obteniendo como resultado la cantidad o porcentaje de eventos AER totales mínimos que se han de emitir para obtener dicho movimiento en la ROI.

Para obtener estos datos es imprescindible haber seleccionado al menos una región de interés y haber activado el check Save Result en el apartado de extracción de movimiento.

El fichero Excel generado contiene toda esta información en columnas, con títulos para identificar cada dato. La información que aparece en cada hoja Excel es de las regiones de interés, es decir, por cada región de interés definida se obtiene su correspondiente hoja Excel.

Mensajes de la aplicación y otros elementos:

La aplicación mostrará mensajes a través de ventanas modales para que el usuario tome decisiones en el mismo instante, como por ejemplo salir de la aplicación, necesitará una confirmación por parte del usuario. Para ello se mostrarán ventanas modales.

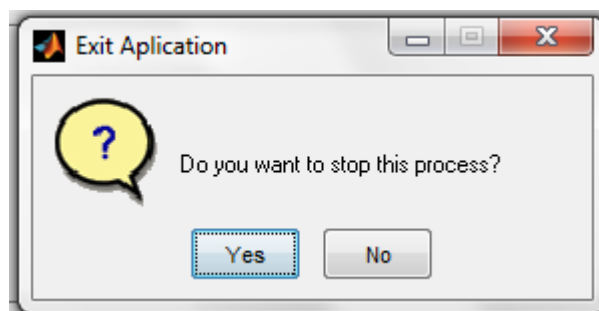


Fig 23. Ventana modal de notificación



Además todos los mensajes que la aplicación genera (un proceso ha comenzado o terminado, información o características del video, información del resultado obtenido, etc.) serán mostrados en la consola integrada en la aplicación.

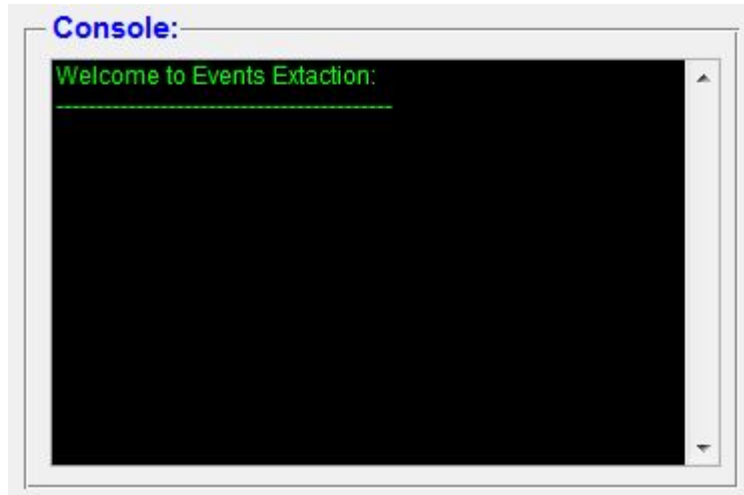


Fig 24. Consola de salida del sistema

Toda la información mostrada por dicha consola será recogida en un fichero **LOG** que creará la aplicación en el directorio de *Result*. Se creará un fichero LOG por cada ejecución de la aplicación, si se ejecuta varias veces al día, será recogido en un solo fichero LOG cuyo nombre será el día y mes en el que se ha ejecutado la aplicación. (Ej. *25-Mar-2012_LOG.txt*).

La barra de progreso es un elemento que cobra mucha importancia en esta aplicación, ya que nos da una estimación de cuánto tiempo de cálculo está procesando las operaciones.

Se ha rediseñado y mejorado la barra de progreso original que proporciona Matlab (*Waitbar*).

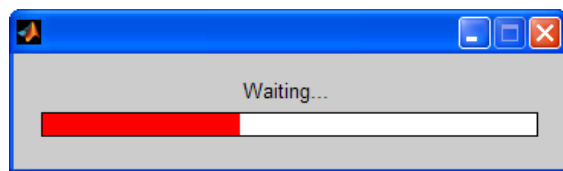


Fig 25. Barra de progreso Original



Se ha añadido el título del proceso que se está llevando a cabo dentro de la propia barra, además de añadir un porcentaje y un tiempo estimado que le queda para completar dicho proceso.

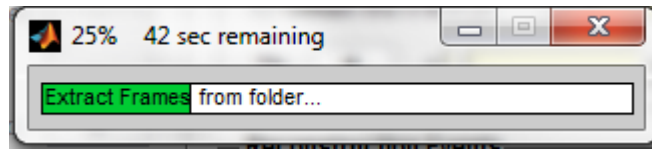


Fig 26. Barra de progreso Modificada

Para más ayuda, todos los controles proporcionan una ayuda extra a la comprensión de la tarea que realiza dejando el puntero del ratón sobre el control. Son los llamados tooltip.

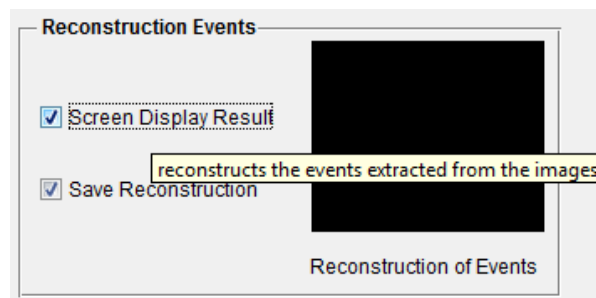


Fig 27. Mensajes informativos o *tooltips*