UN PROTOTIPO PARA RECONSTRUCCIÓN 3D DE LA CÁMARA CAFADIS

Nombre del Primer Autor(1), Nombre del Segundo Autor(1)
{}@ull.es
(1)Dpto. de Física Fundamental y Experimental, Electrónica y Sistemas.
Universidad de La Laguna
Av. Francisco Sánchez s/n. 38200. La Laguna. S/C de Tenerife

***Abstract*— Tree dimensional scene reconstruction has been a fundamental problem in computer vision in the last decades. Many alternatives exist in order to solve this problem. Pasive techniques use camera arrays for reconstructing 3D information of the scene.**

The CAFADIS camera is an integration of optical, processing, and computing techniques for achieving real time performance in reconstructing depth information of a scene. The optical configuration used was a plenoptic sensor. The processing of incomming data was done using super-resolution techniques that were implemented on multiple graphic processing units (GPUs).

This has allowed to obtain a first prototype of the CAFADIS camera that is able to feed an autostereoscopic 3DTV display with a resolution of 535×435 at 27 frames per second.

I. INTRODUCCIÓN

La reconstrucción 3D ha sido uno de los problemas fundamentales en visión por computador durante las últimas decadas. Existen varias alternativas para dar solución a este problema. Por un lado, mediante técnicas de visión activa que implican interacción del sistema con la escena, y por otro mediante técnicas pasivas que tratan de capturar la información 3D sin interactuar con la escena. Generalmente las técnicas pasivas emplean varios sensores para obtener información sobre la escena desde distintos puntos de vista. Este último enfoque es el que se emplea en este trabajo.

La cámara CAFADIS utiliza un sensor plenóptico [1], [2] para obtener información multivista de la escena, con el fin de reconstruir la información de distancia. La principal ventaja de este montaje óptico es que es compacto, con un aspecto similar al de las cámaras tradicionales, y permite capturar un número elevado de puntos de vista. La utilización de este tipo de sensores, junto a los algoritmos de procesamiento y el harware adecuados permite recuperar la información de distancias de una escena en tiempo real.

En este trabajo se presenta un primer prototipo de la cámara CAFADIS que funciona en tiempo real. Esto se ha conseguido gracias a la utilización de algoritmos de procesamiento adecuados, tanto a los datos que se reciben del sensor plenóptico como al hardware paralelo, que se ha utilizado para el procesamiento de estos datos. Los algoritmos de procesamiento consisten en una estimación simultánea de la imagen toda enfocada y el mapa de distancias utilizando técnicas de superresolución [3]. Como plataforma hardware se han empleado unidades de procesamiento gráfico (GPUs), que en los últimos años han evolucionado hasta convertirse en co-procesadores masivamente paralelos de propósito general [4]. El resultado del procesamiento es un flujo de datos en formato 2D+distancia que puede suministrarse directamente y en tiempo real a una televisón 3D autoestereoscópica.

El resto de este trabajo está organizado de la siguiente manera: En la sección II se describe el mecanismo de captura de imágenes. La sección III trata los algoritmos de procesamiento empleados, mientras que la implementación de los mismos se describe en la sección IV. Los resultados obtenidos se describen en la seccion V. Finalemente, las conclusiones y trabajos futuros se encuentran en la sección VI.

II. MECANISMO DE CAPTURA

La cámara CAFADIS emplea una configuración óptica, que se conoce como cámara plenóptica [1], [2], para capturar múltiples puntos de vista de la escena mediante una única exposición. Aunque existen algunas variantes [5], esencialmente se trata de una modificación de una cámara convencional en la que se sitúa un array de microlentes entre el sensor y la lente principal. Las microlentes se colocan a una distancia F_m del sensor, y éstas a su vez tienen una distancia focal F_m. Con esta configuración óptica se captura la información de radiancia de la escena, ya que cada microlente muestrea para un único punto espacial un conjunto de rayos provenientes de distintas direcciones. Para optimizar la resolución angular los números f de las microlentes y el de la lente principal deben coincidir. La imagen que finalmente se captura en el sensor contiene tanto las muestras espaciales como muestras angulares, ya que se ha sacrificado parte de la resolución espacial del sensor para capturar la información angular. Esto obliga a fijar un compromiso entre la resolución angular y la resolución angular [6]

El *light field* es una función que describe la cantidad de luz que viaja en distintas direcciones a través de todos los puntos del espacio. Para parametrizar el *light field* en el interior de la cámara se emplea una parametrización de dos planos, como se muestra en la figura 1. El primero, corresponde a la lente principal y se parametriza mediante las coordenadas $\mathbf{u} = (u_1, u_2)$ y el segundo corresponde al plano de las microlentes

y se parametriza mediante las coordenadas $\mathbf{x} = (x_1, x_2)$. El valor de un punto del *light field*, $L(\mathbf{x}, \mathbf{u})$, corresponde a la intensidad de luminosa del rayo que viaja entre los puntos $\mathbf{x}$ y $\mathbf{u}$. Utilizando esta parametrización y la versión muestreada del *light field* capturada en el sensor, es posible sintetizar fotogramas enfocados a diferentes distancias [2], [7].

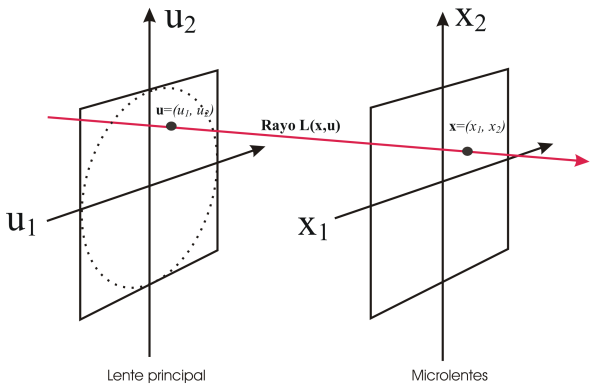

Fig. 1. Parametrización del *light field* mediante dos planos.

III. Procesamiento de datos.

En esta sección se describe brevemente el algoritmo empleado para obtener la información 3D a partir de los datos capturados mediante la configuración de cámara plenoptica descrita en la sección anterior. El algoritmo está constituido por dos pasos fundamentales: El cálculo de la transformada del *focal stack* y la estimación de distancias. El primero computa un conjunto de fotogramas enfocados a distintas distancias, junto con un estimador del desenfoque. El segundo estima cuál es la distancia más probable a partir del estimador del desenfoque. Una vez finalizados estos dos pasos es sencillo obtener una imagen toda enfocada.

El primer paso consiste en el cómputo de una serie de fotogramas enfocados a distintas distancias a partir de los datos del *light field*. Para obtener un fotograma enfocado a una cierta distancia αF es necesario computar la siguiente integral [2]:

$$P_\alpha[L](\mathbf{x}) = \frac{1}{\alpha^2 F^2} \int L\left(\mathbf{u}\left(1 - \frac{1}{\alpha}\right) + \frac{\mathbf{x}}{\alpha}, \mathbf{u}\right) d\mathbf{u} \quad (1)$$

La transformada del *focal stack* se define como $\delta[L](\mathbf{x}, \alpha) = P_\alpha[x]$, es decir un conjunto de fotogramas enfocadas a todas las distancias posibles.

En la práctica, como se utiliza una versión muestreada de L, hay que discretizar la integral en 1 e interpolar las muestras. Suponiendo un *light field* de $n \times n$ microlentes y tras las cuales hay $m \times m$ píxeles, generalmente se espera obtener un conjunto de imágenes cuya resolución espacial es $n \times n$ [2], [8], [9]. Esta resolución es insuficiente para la mayoría de las aplicaciones. Una alternativa para obtener imágenes de mayor resolución de salida es aplicar técnicas de superresolución, consistentes en la obtención de imágenes de alta resolución a partir de un conjunto de imágenes de baja resolución. Este tipo de técnicas han sido desarrolladas para cámaras plenópticas [10], [3], [11].

La versión discreta de la integral en 1 viene dada por:

$$P_\alpha[L](\mathbf{x}) = \frac{1}{|U|} \sum_{u \in U} L\left(\mathbf{u}\left(1 - \frac{1}{\alpha}\right) + \frac{\mathbf{x}}{\alpha}, \mathbf{u}\right) d\mathbf{u} \quad (2)$$

donde U es:

$$U = \left\{\mathbf{u} \in \mathbb{Z}^2 | \mathbf{u}\left(1 - \frac{1}{\alpha}\right) + \frac{\mathbf{x}}{\alpha} \in \mathbb{Z}^2\right\} \quad (3)$$

Los métodos de baja resolución computan 2 para valores de $\mathbf{x} \in \mathbb{Z}^2$ con lo que obtienen una resolución final de $n \times n$. El conjunto de valores de α para los cuales se cumple 3 es un conjunto discreto. Sin embargo, es posible obtener fotogramas cuya resolución espacial es mucho mayor si se utilzan valores de $\mathbf{x} \in \mathbb{R}^2$ que cumplen la condición en 3. Esto permite obtener valores para coordenadas espaciales no enteras a partir de valores del *light field* etiquetados con coordenadas enteras, que están disponibles en la imagen plenóptica. Sin embargo, la máxima resolución alcanzable no es uniforme para todos los valores de α, lo que ha dado lugar a varios algoritmos que computan los fotogramas correspondientes a un subconjunto del conjunto total de valores posibles de α [3], [10].

Paralelamente al cómputo del *focal stack* de color se computa también el *focal stack* de varianzas que sirve como estimador del desenfoque. Esto requiere que $|U| \geq 2$, ya que consiste en computar la varianza a los píxeles que intervienen en 2.

Una vez computado el estimador de desenfoque éste se utiliza para determinar el nivel de profundidad más enfocado. Para ello se utiliza el algoritmo *belief propagation* [12]. Este algoritmo trata de estimar la distancia más probable minimizando una función de energía E, que también puede verse como la máxima verosimilitud de la distribución a posteriori de un campo aleatorio de Markov [13].

Esta función de energía E relaciona una energía de datos E_d y una energía de suavizado E_s mediante la expresión $E = E_d + \lambda E_s$, donde λ determina la importancia relativa de cada término. La energía de los datos E_d viene determinada por una función de costo, que en este caso es el *focal stack* de varianzas.

Para determinar la energía de suavizado E_s se asume que los píxeles de la imagen forman una malla bidimensional (grafo) de modo que p está situado en coordenadas p=(x_1, x_2) sobre esa malla. Si N es el conjunto de los cuatro vecinos del píxel p, la energía de suavizado se define como:

$$E_s = \sum_{\{p,q\} \in N} V_{pq}(d_p, d_q) \quad (4)$$

siendo d_p y d_p los niveles de profundidad del pixel p y q respectivamente.

En este caso se ha optado por definir $V_{pq}(d_p, d_q)$ como:

$$V_{pq}(d_p, d_q) = \begin{cases} 1 & si \quad d_p \neq d_q \\ 0 & si \quad d_p = d_q \end{cases} \quad (5)$$

El algoritmo trata de encontrar un mínimo de la energía E. Para ello se emplea un método iterativo de paso de mensajes entre los píxeles de la malla bidimensional. Se define $M^t_{p \to q}$

como el mensaje que el píxel p envía a su vecino q en la iteración t, siendo éste un vector de tamaño m, que es el número de niveles de profundidad que se tienen en cuenta. La regla de actualización de los mensajes es la siguiente:

$$M^t_{p\to q}(d_q) = \min_{d_p} \begin{cases} \mu(c(d_p) + \sum M^{t-1}_{s\to p}(d_p)) \\ \\ -M^{t-1}_{q\to p}(d_p) + V_{pq}(d_p, d_q) \end{cases} \quad (6)$$

Una vez finalizado el proceso iterativo se calcula el mínimo de la función de creencia para cada píxel y se obtiene un mapa de distancias que permite reconstruir una imagen toda enfocada.

IV. Implementación

La implementación de este prototipo de la cámara CAFADIS se realizó sobre una modificación de una cámara científica Imperx 2M30-C con un *frame rate* de 33 fps y cuyo sensor tiene una resolucón espacial de 1600x1200 píxeles. Frente al mismo se situó un array de microlentes de 112×76 con lo que, tras llevar a cabo el alineamiento adecuado, fue posible obtener un *light field* de $112 \times 76 \times 11 \times 11$.

Para el procesamiento de los datos se conectó la cámara a un ordenador mediante una interfaz *camera link* . Dicho ordendador estaba provisto con 2 procesadores Intel Xeon X5560, 16 Gb de memoria RAM, 2 tarjetas NVidia Tesla C1060, 1 tarjeta gráfica NVidia GeForce GTX 285 y un *frame grabber* DALSA Coreco x64-Full. El sistema operativo empleado fue Debian GNU/Linux 5.0 y la programación de las unidades de procesamiento gráfico (GPUs) se realizó por medio de la API NVidia CUDA 2.2 [14].

Como se indicó en la sección III el uso de algoritmos convencionales permite obtener una resolución espacial de la imagen final baja, en este caso de 112×76, que es insuficiente para rellenar una pantalla con una resolución de vídeo estándar. Por ello se aplicó el algoritmo de superresolución descrito en [3], obteniéndose una resolución final de 580×435 píxeles.

Por otra parte, hubo que paralelizar los algoritmos antertiormente descritos para que se adaptaran a la arquitectura masivamente paralela de las GPUs. Para la ejecución de las funciones o *kernels* los distintos hilos o *threads* se agrupan en una rejilla de bloques. Cada *thread* está identificado mediante un identificador de bloque y un identificador de *thread* dentro del bloque. Estos identificadores pueden ser multidimensionales, hasta un máximo de tres dimensiones. El identificador de cada *thread* se emplea dentro de cada función para determinar que porción de los datos de entrada va ser procesada. Además, al tratarse de un ordenador multi-GPU, fue necesario el uso de técnicas para dividir los datos de entrada, como las descritas en [15], de modo que cada una de las tres GPUs pudiera realizar los cálculos asignados de forma independiente. Luego, los resultados parciales se recombinan para dar lugar al resultado final. Este proceso se muestra de forma más detallada en la figura 2.

Una vez obtenido un par compuesto por una imagen de color toda enfocada y el mapa de distancias correspondiente cada

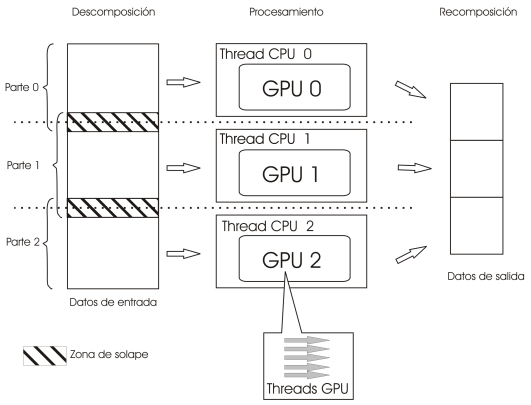

Fig. 2. Planificación de las distintas partes de los datos de entrada sobre las tres GPUs.

frame se visualizó en un display autoestereoscópico Philips WOVvx 20“ cuya resolución operando en 3D es de 800×600 píxeles. Para ello hubo que realizar la implementación de una librería basada en el sistema X-Window que permitiera enviar los datos al monitor según las especificaciones del fabricante [16].

V. Resultados.

Para mostrar la eficacia de los algoritmos empleados se utliza una imagen plenóptica capturada con la cámara CAFADIS que da lugar a un *light field* de $112 \times 76 \times 11 \times 11$ píxeles que se muestra en la figura 3. El resultado obtenido se muestra en la figura 4 donde se puede observar la imagen toda enfocada y el mapa de distancias correspondiente.

El prototipo actual permite el procesamiento de video en tiempo real a una velocidad de frame de 27 fps. Esto se puede comprobar visualizando el video de [17]. El prototipo solo funciona con escenas pequeñas debido a la configuración óptica empleada en el mismo. Esta limitación se puede salvar utilizando otros elementos ópticos distintos a los empleados.

VI. Conclusiones y trabajos futuros.

Se ha presentado un primer prototipo de la cámara CAFADIS que funciona en tiempo real. La adecuada integración de técnicas ópticas con los algoritmos y el hardware de procesamiento han permitido obtener un sistema que genera un flujo de video 3D en tiempo real en formato 2D+distancia, que puede alimentar una televisión 3D autoestereoscópica.

En el futuro, se plantea la posibilidad de adaptar la salida a otros formatos de visualización. Asimismo, el sistema será probado con otras configuraciones ópticas, para construir el sensor plenóptico que permita capturar escenas mayores que las que admite el diseño actual. También se considerará portar el sistema a otras plataformas hardware.

Agradecimientos

Este proyecto ha sido financiado por el Plan Nacional de I+D+i (Proyecto AYA-1305) del Ministerio de Ciencia e Innovación, por la Agencia Canaria de Investigación,

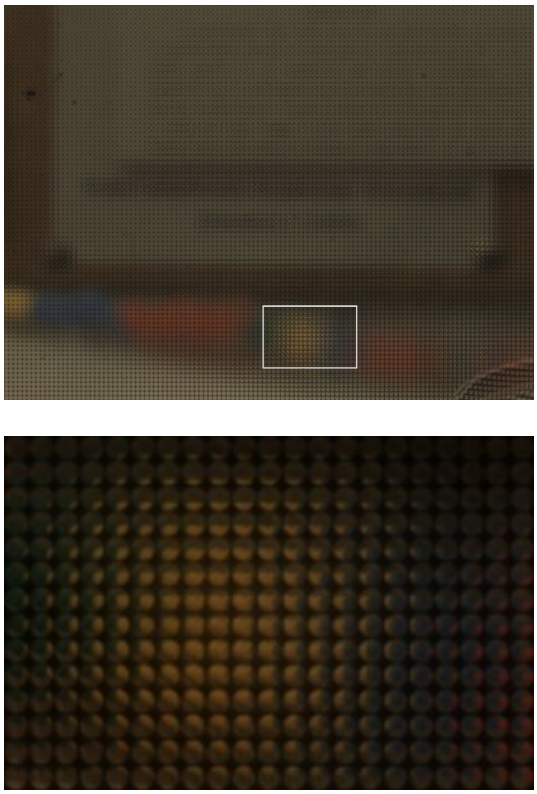

Fig. 3. *Light field* capturado con la cámara CAFADIS. Arriba: Imagen completa. Abajo: Detalle de la imagen encerrada en el cuadro blanco.

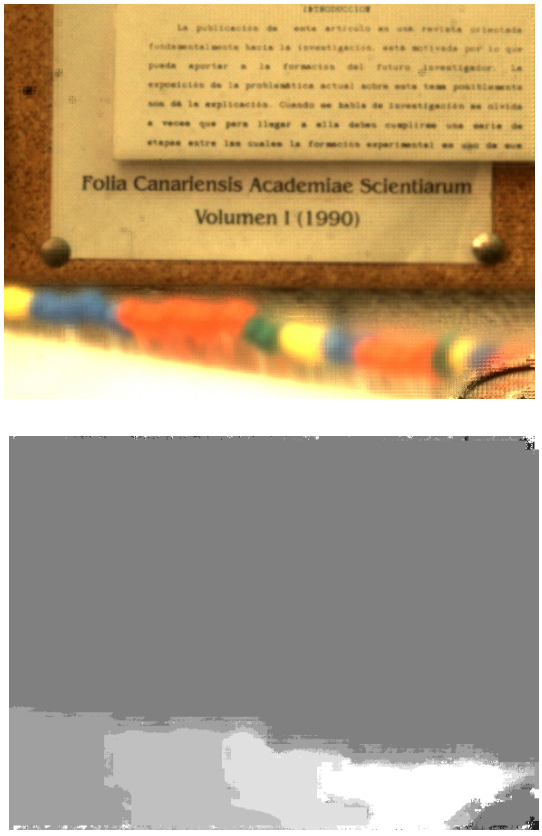

Fig. 4. Resultado de la aplicación de los algoritmos de procesamiento a una imagen estática. Arriba: Imagen toda enfocada. Abajo: Mapa de distancias.

Innovación y Sociedad de la Información (ACIISI) y por el Fondo Europeo de Desarrollo Regional (FEDER).

REFERENCES

[1] E. H. Adelson and J. Y. A. Wang, "Single lens stereo with a plenoptic camera," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 99–106, 1992.

[2] R. Ng, M. Levoy, M. Brédif, G. Duval, M. Horowitz, and P. Hanrahan, "Light field photography with a hand-held plenoptic camera," Tech. Rep., April 2005. [Online]. Available: http://graphics.stanford.edu/papers/lfcamera/

[3] F. Perez Nava and J. Luke, "Simultaneous estimation of super-resolved depth and all-in-focus images from a plenoptic camera," in *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video, 2009*, may 2009, pp. 1 –4.

[4] J. Owens, M. Houston, D. Luebke, S. Green, J. Stone, and J. Phillips, "Gpu computing," *Proceedings of the IEEE*, vol. 96, no. 5, pp. 879 –899, may 2008.

[5] A. Lumsdaine and T. Georgiev, "The focused plenoptic camera," in *International conference on computational photography*, 2009.

[6] T. Georgiev, K. C. Zheng, B. Curless, D. Salesin, S. Nayar, and C. Intwala, "Spatio-angular resolution tradeoff in integral photography," in *Proceedings of Eurographics Symposium on Rendering*, 2006.

[7] A. Isaksen, L. McMillan, and S. J. Gortler, "Dynamically reparameterized light fields," in *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 2000, pp. 297–306.

[8] F. Perez Nava, J. Marichal Hernandez, and J. Rodriguez Ramos, "Discrete focal stack transform," in *Proceedings of 16th European Signal Processing Conference (EUSIPCO 2008)*, 2008.

[9] J. Marichal Hernandez, J. Luke, F. Rosa, F. Perez Nava, and J. Rodriguez Ramos, "Fast approximate focal stack transform," in *3DTV09*, 2009, pp. 1–4.

[10] A. Lumsdaine and T. Georgiev, "Full resolution lightfield rendering," Tech. Rep., 2008.

[11] T. Bishop, S. Zanetti, and P. Favaro, "Light field superresolution," in *Proceedings of International conference on computational photography*, 2009.

[12] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient belief propagation for early vision," *Int. J. Comput. Vision*, vol. 70, no. 1, pp. 41–54, 2006.

[13] V. Kolmogorov and M. Wainwright, "On the optimality of tree-reweighted max-product message-passing," in *Proc. e on Uncertainty in Artificial Intelligence*, Edinburgh, Scotland, 2005.

[14] NVidia Corporation, "nvidia cuda programming guide."

[15] J. Lüke, F. Pérez Nava, J. Marichal-Hernández, J. Rodríguez-Ramos, and F. Rosa, "Near real-time estimation of super-resolved depth and all-in-focus images from a plenoptic camera using graphics processing units," *International Journal of Digital Multimedia Broadcasting*, vol. 2010, p. 12, 2010.

[16] Philips 3D Solutions, "3d interface specifications, white paper."

[17] [Online]. Available: http://pejeverde.lct.ull.es/cafadis/ursi2010