Universidad Zaragoza
1542

Trabajo Fin de Máster

Framework para la Evaluación de Técnicas de Reconstrucción de geometría no visible

An Evaluation Framework for Non-Line-of-Sight Reconstruction Techniques

Autor

Miguel Jorge Galindo Ramos

Director

Adrián Jarabo Torrijos

Ponente

Diego Gutiérrez Pérez

ESCUELA DE INGENIERÍA Y ARQUITECTURA
2019

AGRADECIMIENTOS

A mi familia, varios Marcos y amigos por soportar mis divagaciones con luces en movimiento y vóxeles ininteligibles, a mis directores Adrián y Diego por aguantarme y darme esta gran oportunidad, a Julio, Matt y Gordon por su colaboración en el trabajo, a Ibón por su inestimable ayuda con la teoría y el software y a todos los compañeros del Graphics and Imaging Lab

¡Muchas gracias!

Título del resumen

RESUMEN

En la última década, la captura del transporte transitorio de la luz a trillones de fotogramas por segundo está teniendo un gran impacto en los campos de gráficos y visión por computador. La riqueza de la información en el perfil temporal, combinada con técnicas de imagen computacional apropiadas, hace posible recuperar vídeos de la luz en movimiento, capturar objetos a través de medios turbios, inferir propiedades materiales, o incluso ver a través de esquinas. Esta última aplicación, conocida como *Non-Line-of-Sight imaging (NLOS)*, ha resultado ser de especial interés, con multitud de aplicaciones potenciales como apoyo en situaciones de rescate, seguridad en vehículos autónomos o endoscopia médica. No obstante, estas aplicaciones siguen lejos de ser realidad, con la mayoría prototipos de la tecnología funcionando tan solo en entornos controlados de laboratorio ideados para lograr ver superficies sencillas y aisladas a través de una esquina.

En este trabajo modificamos un motor de render transitorio para crear un dataset sintético de 300 escenarios *NLOS* con complejidad variada y que publicamos para fomentar la colaboración con otros equipos de invesigación simplificando su tarea. Tenemos como objetivo proponer retos mucho más complejos a los que se han enfrentado los investigadores hasta ahora, buscando lograr mejoras significativas en los resultados que lleven *NLOS imaging* a aplicaciones prácticas en el mundo real. En consecuencia, buscamos poder evaluar resultados de diferentes métodos de reconstrucción, para lo que proponemos métricas que permitan compararlos de forma justa a datos de referencia. Además, esperamos que el dataset permita el uso de técnicas de aprendizaje automático en el campo. Finalmente, derivamos un nuevo método y mostramos sus resultados junto los dos métodos más representativos dentro del estado del arte de la visión a través de esquinas, verificando los datos de nuestro dataset.

Índice

Capítulo 1

Introducción y objetivos

La tecnología involucrada en la captura de una fotografía avanza con gran velocidad. La primera fotografía realizada por Joseph Nicéphore Niépce en 1826 requirió de 8 horas de exposición a la luz, pasando a minutos tras unos pocos años de avances. En 1872, Eadweard Muybridge logró capturar la secuencia del galope de un caballo y en 1964 Harold Edgerton *pausó* el tiempo para capturar una bala atravesando una manzana. Hoy en día, las cámaras rápidas pueden capturar cientos de miles de fotogramas en un segundo, permitiendo observar eventos mucho más rápidos de lo que nuestro sistema visual es capaz de percibir.

Pero el avance no se detiene ahí: en 2013, Velten et al. (2013) presentaron "femto-fotografía" esta tecnología permite numerosas aplicaciones en los campos de la visión y los gráficos por computador, incluyendo: recuperar profundidad en una escena midiendo el retardo de la luz desde que se emite hasta que alcanza al sensor, recuperar funciones de reflectancia de las superficies (*BRDFs*) (Pandharkar, 2011; Naik et al., 2014), separar los componentes de iluminación (O'Toole et al., 2014), estimar las propiedades ópticas de medios participativos y ver a través de ellos (Heide et al., 2014b), etc. Yendo más allá de lo visible, *Non-Line-of-Sight Imaging (NLOS)* se refiere al estudio de las propiedades de una escena que no observamos de forma directa, es decir, no existe una línea de visión directa con la escena (ver Figura 1.1). La única información de la que se dispone proviene de interacciones indirectas con regiones visibles de la escena. Con la información del tiempo de vuelo de la luz podemos desambiguar parcialmente los caminos que sigue por la parte oculta de la escena para obtener información útil de ella. Hasta la fecha, varios trabajos han demostrado el uso de esta tecnología para reconstruir geometría ocluída (Velten et al., 2012; Buttafava et al., 2015; La Manna et al., 2018; Heide et al., 2018; O'Toole et al., 2018; Liu et al., 2018; Lindell et al., 2019).

Realizamos un análisis del campo de la reconstrucción de información de escenas *NLOS* y descubrimos que actualmente presentan un gran problema: el coste y

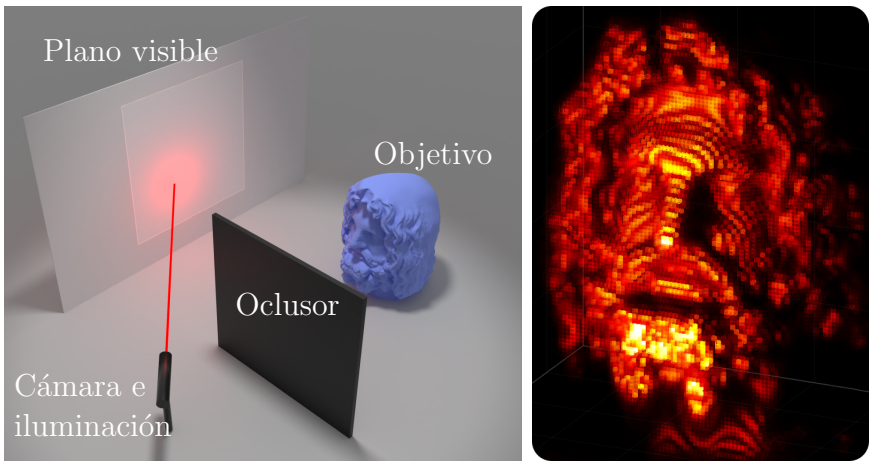


Figura 1.1: Escena NLOS *básica, con un único objeto enfrentado a un plano difuso en la línea de visión de los elementos de iluminación y captura. A la derecha reconstrucción de la escena oculta utilizando el método de (Velten et al., 2012) y ajustado manualmente para su visualización.*

complejidad de los dispositivos de captura hacen que no existan grandes conjuntos de datos públicos, variados y de alta calidad sobre los que trabajar. Otros problemas en los campos de visión por computador y de informática gráfica han empleado datasets y benchmarks para mejorar resultados drásticamente. Este ha sido el caso, por ejemplo, en problemas de *optical flow* (Butler et al., 2012), reconstrucción de profundidad con línea de visión y corrección de interferencias por caminos múltiples (Marco et al., 2017; Su et al., 2018), entre otros. Siguiendo el ejemplo de estos trabajos, buscamos suplir la carencia de datos con un dataset sintético de escenas *NLOS*, haciendo uso de los últimos avances en informática gráfica para la simulación de imagen transitoria o *transient rendering* (Jarabo et al., 2014). Junto a los datos, proponemos métricas cuantitativas basadas en profundidad de reconstrucción para permitir una evaluación objetiva de diferentes métodos de reconstrucción de geometría oculta. Este dataset, reducirá la barrera de entrada a la investigación en el campo, permitirá realizar prototipados rápidos, comparaciones justas de diferentes métodos, detectar situaciones problemáticas para reconstrucción, e incluso incitará a la competición en la mejora de resultados.

En resumen, la principal aportación es un *framework* para evaluar y diseñar nuevas técnicas de reconstrucción NLOS. Este *framework* consiste en un dataset de acceso público, las métricas, y evaluaciones con trabajos recientes, incluyendo implementaciones de referencia de algoritmos de reconstrucción de geometría oculta. Junto a cada elemento del dataset, se proporcionan volúmenes de vóxeles de referncia *ground-truth* de las regiones ocultas de las escenas para permitir evaluar las reconstrucciones de forma objetiva en base a métricas cualitativas y cuantitativas.

Este trabajo se ha presentado cómo un póster en la conferencia de fotografía computacional *ICCP 2019* en Tokio en mayo de 2019, y se presentará de nuevo en *SIGGRAPH 2019* en Los Ángeles en agosto de 2019.

1.1. Contexto

Este trabajo se realiza desde el grupo de investigación *Graphics and Imaging Lab* de la universidad de Zaragoza y en particular dentro del proyecto *REVEAL* financiado por *DARPA*, en el que participa el grupo junto a otras instituciones de prestigio internacional como Stanford, University of Wisconsin, University of Toronto y Carnegie Mellon. En concreto, en este trabajo colaboramos directamente con las instituciones Stanford y Carnegie Mellon.

Como construimos el dataset dentro de este proyecto de investigación, es de especial interés que el acceso a los datos sea público. Se esperan colaboraciones con los otros miembros del proyecto y que se fomente el uso del dataset por medio de futuras publicaciones. Se puede encontrar en la página web `graphics.unizar.es/nlos`.

Capítulo 2

Conocimiento previo

En este capítulo repasamos las técnicas de informática gráfica en relación con la simulación del transporte de la luz, incluyendo su estado transitorio, y las técnicas de imagen computacional que emplean imagen transitoria para recuperar información sobre geométria oculta. Ambos temas resultan clave para guiar el diseño de los escenarios de los que consta el dataset, realizar simulaciones correctas y asegurar su validez.

2.1. Transporte de luz

Cuando encendemos la luz en una habitación, los elementos lumínicos comienzan a emitir radiación electromagnética en el espectro visible. La luz se propaga por el aire hasta que alcanza una superficie donde ocurrirá una interacción en la que parte de su energía se verá absorbida y parte reflejada en una nueva dirección, dependiendo del material de la superficie. Estas interacciones ocurren trillones de veces por segundo, rápidamente alcanzando un equilibrio en la luz entrante y saliente de las superficies. Nuestro sistema visual recoge la luz proveniente de las interacciones con el entorno y transmite información sobre sus cualidades al cerebro, permitiéndonos interpretar formas y colores.

Para una vista técnica detallada del estado del arte en el transporte de luz y su estado transitorio, referimos el lector al trabajo de Jarabo et al. (2017), que hace un estudio en profundidad en el que nos apoyamos a lo largo de este trabajo.

2.1.1. Captura del transporte de luz en estado transitorio

La fotografía nos permite emular el comportamiento de nuestro sistema visual para capturar el estado del transporte de luz en una escena. Con casi siglo y medio de historia, las técnicas de captura de imágen de la fotografía están ya muy asentadas en la sociedad, formando parte de nuestro día a día. Tanto tiempo de desarrollo nos ha

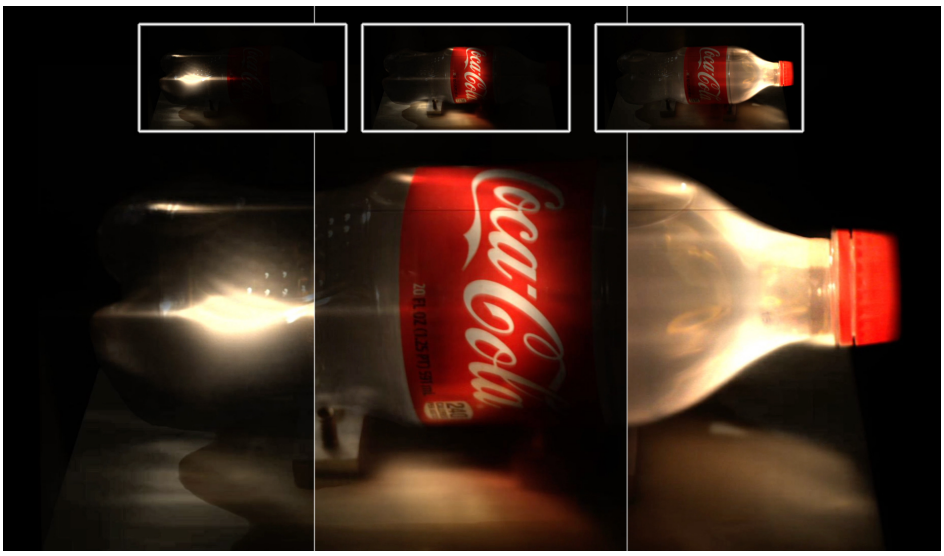


Figura 2.1: Captura del transporte de luz a escala de pico-segundos. Un pulso láser es emitido a través del fondo de la botella, el cual viaja interactuando con el medio contenido en la botella. La imagen está compuesta de los tres fotogramas mostrados arriba (imagen extraída de (Velten et al., 2013)).

permitido superar, y con creces, las capacidades del sistema visual humano, al menos en aspectos como resolución, sensibilidad a la luz y resolución temporal, siendo este último de particular interés en este trabajo (Wetzstein et al., 2011).

Centrándonos en el dominio temporal, las cámaras rápidas son capaces de capturar hasta un millón de fotogramas por segundo. A velocidades tan altas, el equipo de captura está limitado por el ancho de banda que puede transmitir en memoria, alcanzando estas velocidades solo con muy baja resolución, y por la cantidad de luz que pueden recibir en el tiempo de exposición correspondiente a cada fotograma. Evitando el *hardware* específico para captura ultrarrápida, trabajos recientes utilizan imagen computacional para capturar este tipo de secuencias con una sola exposición (Serrano et al., 2017; Antipa et al., 2019). Estos sistemas son útiles para aplicaciones industriales y científicas que buscan desentramar sucesos muy rápidos.

Pero cuando vamos más allá y consideramos intervalos de tiempo mucho más cortos, de pico e incluso femtosegundos (10^{-15}s), hablamos de transporte transitorio de la luz. La captura de este tipo de transporte radiativo es muy compleja, y está muy lejos de las posibilidades de la fotografía clásica, incluso con cámaras rápidas, requiriendo de capturas un millón de veces más rápidas. El trabajo seminal de Velten et al. (2013) demostró que es factible, logrando capturar imágenes como la de la Figura 2.1 utilizando cámaras de imagen unidimensional (*streak cameras*) (Hamamatsu, 2012) para capturar luz a un billón de fotogramas por segundo. Pese a que se han propuesto variaciones del sistema de Velten et al. (2013) para acelerar la captura (Heshmat et al., 2016), su coste sigue siendo muy elevado, y son muy complicados

de operar fuera de entornos controlados en un laboratorio. Para solucionar dichos problemas, se han propuesto otros sistemas para la captura de imagen transitoria: Los *Single-Photon Avalanche Diodes (SPAD)* (Kirmani et al., 2014; Gariepy et al., 2015; O'Toole et al., 2017), son lo suficientemente sensibles como para detectar fotones individuales y tienen un coste menor que las *streak cameras*, pero su resolución temporal y espacial es menor. Los *Photonic Mixer Devices (PMD)* (tecnología utilizada en los dispositivos *Kinect2*) (Lange et al., 2000), modulan la frecuencia de un elemento de iluminación activo y miden su desviación en captura para computar su tiempo de vuelo. Se usan más comúnmente para medir profundidad, requiriendo modificaciones de hardware para obtener resultados satisfactorios en imagen transitoria (Heide et al., 2013; Kadambi et al., 2013; Lin et al., 2014). Finalmente, los sistemas *gated* realizan multiples exposiciones de un mismo evento, desplazando los instantes de inicio y fin de la captura en una cámara rápida, efectivamente recogiendo en cada exposición un instante temporal diferente. Aunque se concibió originalmente para captura de imágen de superficies a distancias determinadas (*range imaging*), reordenando las capturas de las distintas exposiciones es posible reconstruir imágenes transitorias (Busck and Heiselberg, 2004; Laurenzis and Velten, 2014). Todos los sistemas de captura sufren un problema común: con tiempos de exposición tan cortos, la cantidad de fotones que alcanzan el sensor es muy baja, incluso cuando se emplean elementos de iluminación de gran potencia. Explotando la repetibilidad del transporte de luz en escenas estáticas, una solución recurrente consiste en realizar múltiples exposiciones independientes y acumularlas para mejorar la relación señal/ruido. La iluminación debe consistir de pulsos ultrarrápidos de forma que no se alcance un equilibrio en el sistema entre pulsos.

En definitiva, en estos momentos la captura de imagen transitoria es una tecnología en etapas tempranas de desarrollo. Las posibilidades son enormes, pero la tecnología todavía está lejos de soportar aplicaciones prácticas, tanto por su coste como por su capacidad limitada para proporcionar datos con tasas de ruido aceptables en tiempos razonables.

2.2. Simulación del estado transitorio de la luz

A diferencia del render más clásico, en el que el producto final es una imagen, el render transitorio produce un vídeo de la trayectoria que sigue la luz por la escena. Añadir esta dimensión temporal hace que el render transitorio sea mucho más complejo de evaluar de forma satisfactoria. Adicionalmente, las técnicas más sofísticadas diseñadas a lo largo de los años para mejorar la eficiencia en render clásico no son necesariamente aplicables al render transitorio, que representa un cambio

drástico de paradigma.

La reciente introducción de la captura de imagen transitoria a los campos de gráficos y visión por computador realzan la necesidad de disponer de herramientas para simularlo de forma eficiente. Con simulación podemos obtener datos de imagen transitoria de referencia robustos y de alta calidad, supliendo las carencias de la captura de este tipo de eventos en el mundo real, y es la base de modelos directos (*forward models*) para resolver diversos problemas inversos. En este contexto, Smith et al. (2008) formalizan la idea de render transitorio generalizando la formulación clásica del render estacionario (Kajiya, 1986). Varios autores proponen alternativas para obtener simulaciones físicamente correctas del transporte transitorio de la luz. Una opción es tratar la simulación como un vídeo, en el que cada fotograma corresponde con un instante de tiempo diferente y renderizarlos independientemente. En la práctica, esta opción es muy ineficiente debido a la dificultad de encontrar caminos entre fuentes de luz y sensores que cumplan la restricción temporal impuesta en cada fotograma. Los trabajos de Jarabo (2012); Marco (2013); O'Toole et al. (2014); Ament et al. (2014); Pitts et al. (2014); Adam et al. (2016) resuelven este problema reutilizando las muestras en todos los fotogramas, y agrupándolas en el tiempo. Estos métodos tienen la ventaja de ser sencillos de implementar partiendo de un motor de render estacionario pero, por contra, son ineficientes, convergiendo muy lentamente. Más adelante, Jarabo et al. (2014) extienden la formulación de la integral de caminos (Veach, 1997), y presentan un *framework* para su simulación proponiendo nuevas estrategias de muestreo específicas al estado transitorio de la luz para mejorar el orden de convergencia. Recientemente, Pan et al. (2019) extienden la técnica clásica de *instant radiosity* (Keller, 1997) para simular el transporte de luz en estado transitorio en tiempos interactivos. Otros trabajos mejoran más la eficiencia de las simulaciones a cambio de ignorar algunos aspectos del transporte transitorio de la luz, como simular solo el primer rebote de la luz en aceleradores gráficos (*GPU*) (Keller et al., 2007; Keller and Kolb, 2009; Hullin, 2014; Klein et al., 2016).

2.3. *NLOS Imaging*

Como aplicación relevante de la captura de luz en estado transitorio, la imagen de escenarios sin línea de visión (*Non-Line-of-Sight, o NLOS Imaging*) busca recuperar propiedades de una escena ocluida en base a información de iluminación indirecta. El escenario básico que se plantea en este problema es el que representamos en la Figura 1.1, donde se dispone de una cámara y una fuente de luz pero parte de la escena se encuentra fuera de su línea de visión. La única fuente de información de la región

ocluida se encuentra, por tanto, en las interacciones indirectas de la luz que se emite sobre una región visible de la escena (típicamente un plano), donde rebota alcanzando la región ocluida, y rebota de nuevo de vuelta al muro y, finalmente, a la cámara. Kirmani et al. (2009) proponen por primera vez el uso de imagen transitoria para ver *a través de las esquinas*, recuperando la forma de elementos ocluidos o no visibles. Velten et al. (2012) demostraron empíricamente este concepto, construyendo un sistema capaz de capturar la información necesaria de la iluminación indirecta para, junto con técnicas de imagen computacional basadas en *backprojection*, recuperar la forma de objetos ocultos. Veremos su algoritmo en mayor profundidad en la Sección 5.2.1, donde realizamos una implementación eficiente en CUDA y la utilizamos para verificar la validez de las escenas del dataset. Los trabajos sucesivos de Laurenzis and Velten (2014); Heide et al. (2014a); Hullin (2014); Kadambi et al. (2016); Arellano et al. (2017); Buttafava et al. (2015),utilizan diferentes sistemas de captura y/o de reconstrucción, para alcanzar mejoras en el coste del sistema, su eficiencia o la nitidez de los resultados. Sin embargo, los escenarios que utilizan para validar sus métodos no divergen del que se usa desde el principio, presentando, en general, figuras planas enfrentadas a un plano visible.

Recientemente, el trabajo de Xin et al. (2019) presenta un método diferente para reconstruir superficies en escenarios significativamente más complejos, soportando la reconstrucción de múltiples superficies curvas y materiales variados de forma robusta. Tsai et al. (2019) presentan un método basado en optimización estocástica, realizando simulaciones con un render transitorio sobre una malla de triángulos que ajustan iterativamente, aproximando la geometría oculta. Ambos trabajos validan sus métodos utilizando objetos con superficies más complejas que los anteriores, pero continúan asumiendo entornos aislados muy sencillos. Por contra, Liu et al. (2019b) y Lindell et al. (2019) buscan expandir el uso de *NLOS imaging* presentando técnicas que permiten reconstruir escenarios mucho más complejos. Aunque demuestran mayor robustez y precisión que métodos anteriores en escenas mucho más complejas en entornos reales, el problema de la falta de datos continúa siendo de relevancia, y es muy difícil realizar comparaciones objetivas entre ellos. Es en estos casos donde resulta más interesante utilizar un conjunto de datos con escenas complejas, ya que las escenas sencillas son demasiado simples para detectar diferencias significativas entre ellos.

El trabajo de Klein et al. (2018) es próximo a este en cuanto a que ambos tienen en cuenta la falta de datos de imagen transitoria y liberan un dataset sintético de imagen transitoria. Sin embargo, la extensión y variedad de los datos que contiene el primero es comparable a la ya explorada en los primeros trabajos, manteniendo el foco en objetos aislados, y, aunque propone varios retos para reconstrucción de propiedades sin línea

de visión, su extensión es muy limitada. Por contra, el presente trabajo se centra en reconstrucciones geométricas, proponiendo más escenarios con objetos ocultos variados y entornos más complejos, llegando a incluir situaciones de uso para la tecnología, y no solo escenarios prototipo para su evaluación (ver Figuras 3.1 y 3.3).

Capítulo 3

Diseño y simulación escenas *NLOS*

El objetivo del trabajo es lidiar con la carencia de datos que limita el avance de la investigación creando un conjunto de escenas *NLOS*. En este capítulo justificamos la necesidad de este dataset, repasamos las necesidades que buscamos cubrir y finalmente describimos el tipo de escenas con el que trabajamos.

3.1. Justificación y necesidades

Otros problemas en los campos de visión por computador y de informática gráfica han empleado datasets y benchmarks para mejorar sus resultados drásticamente. Este ha sido el caso, por ejemplo, en problemas de visión estéreo (Geiger et al., 2013), reconstrucción de profundidad con línea de visión (Nathan Silberman and Fergus, 2012) y corrección de interferencia por caminos múltiples (Marco et al., 2017; Su et al., 2018), entre otros. Buscamos seguir estos ejemplos para crear un dataset que cubra las necesidades de datos de alta calidad que existe en *NLOS imaging*, dado que, hasta la fecha, los datos disponibles son escasos y de variedad insuficiente (Klein et al., 2018). Este dataset reducirá la barrera de entrada a la investigación en el campo, permitirá realizar prototipados rápidos, comparar diferentes métodos de forma objetiva, detectar situaciones problemáticas para reconstrucción, e incluso incitará a competir en la mejora de resultados. Entre las escenas debemos poder encontrar ejemplos que exhiban las características siguientes:

Formas geométricas con complejidad creciente Los trabajos previos se centran en escenas con objetos planos simples como letras planas. Con geometría más detallada (criaturas, bustos, árboles, etc.), los fallos en reconstrucción cobran mayor relevancia, y son más fáciles de detectar restringiendo el rango de operabilidad de capturas o métodos.

Las superficies convexas y cóncavas presentan retos adicionales concretos de cara a

la reconstrucción. Las primeras tienen, por definición, diferentes vectores normales a lo largo de las superficies, afectando la intensidad de la radiancia que alcanzará al sensor (las regiones perpendiculares tendrán contribuciones mayores que las tangenciales). Las segundas presentan, además del problema anterior, interreflexiones múltiples, donde la luz reflejada en una región alcanza otra próxima, añadiendo ambigüedad al origen de la señal.

Entornos Usar objetos aislados en un entorno que no refleja luz es otro recurso recurrente en trabajos previos. Dado que la mayor parte de la luz que alcanza el sensor proviene de objetos que se pretende reconstruir, evitando la ambigüedad de tener múltiples superficies. Sin embargo, la presencia de objetos aislados no es común en escenas del mundo real; para que nuestro dataset sea usable como prueba de técnicas robustas fuera del laboratorio, tiene que incluir casos con dichas ambigüedades, múltiples superficies, etc.

Diferentes materiales La mayoría de métodos diseñados hasta la fecha, con las excepciones de (Liu et al., 2019b; Lindell et al., 2019), asumen materiales difusos o retroreflectivos en el plano visible y las superficies ocultas. Otros materiales rompen las asunciones del modelo de transporte de luz que invierten estos métodos, así como reducen parte de la señal en algunos casos.

Escenas con complejidad muy elevada Si buscamos aplicar *NLOS imaging* en entornos como conducción autónoma, operaciones de rescate o exploración médica, debemos aumentar la complejidad de los escenarios con los que trabajamos. El reto de reconstruir geometría variada con oclusiones, interreflexiones, objetos varios próximos, etc. es grande para métodos de reconstrucción actuales, pero no es imposible. Tener este tipo de escenas permite, además, comparar la robustez de los algoritmos ante distintas situaciones, y no solo casos aislados.

Diferentes patrones de captura de señal Como veremos con mayor detalle en la Sección 3.3, diferentes métodos de reconstrucción requieren trabajar con datos capturados con patrones de captura incompatibles entre sí. Incluir varios de estos patrones aumentará la flexibilidad de los datos y permitirá que el dataset sea de utilidad para un público mayor, e incluso realizar comparaciones de los propios patrones de captura en sí mismos en base a la calidad de las reconstrucciones que es posible obtener con ellos.

Rebotes desambiguados Cuando la luz alcanza un sensor en el mundo real sabemos de qué dirección proviene, pero no sabemos qué camino ha seguido desde su emisión en una fuente de luz. Existen múltiples caminos válidos que puede seguir la luz que lo alcanza en un instante, incluyendo caminos en los que ha rebotado en varias ocasiones. Estos caminos son problemáticos para las reconstrucciones *NLOS*, causando ruido o generando geometría falsa en las reconstrucciones. Aprovechando que trabajamos con simulaciones, buscamos que esta ambigüedad sea opcional en el dataset, proporcionando los datos provenientes de diferente número de rebotes por separado. Con esta información, los investigadores pueden probar el caso ideal considerando tan solo el tercer rebote, utilizar técnicas de análisis por síntesis para tratar de realizar la separación en captura real (Wu et al., 2014), o incluso reconstruir geometría a partir de órdenes altos de interreflexiones, es decir, ver a través de dos esquinas (Liu et al., 2019b).

Machine Learning Las técnicas de aprendizaje automático han demostrado ser muy útiles para resolver problemas tanto en visión por computador como en gráficos (Voulodimos et al., 2018; Mitra et al., 2018), pero su potencia está ligada a la disponibilidad de grandes cantidades de datos. Buscamos contar con un número suficientemente elevado de datos que comience a hacer posible el uso de algoritmos de aprendizaje para mejorar algún aspecto de la calidad de las reconstrucciones.

3.2. Agrupación de escenas

Para cubrir las necesidades de forma sistemática, identificamos dos tipos de escenas: las que siguen un patrón básico con un único objeto oculto y las más complejas, que no siguen patrones concretos sino que representan entornos no controlados más similares a los que se pueden encontrar en el mundo.

3.2.1. Escenas básicas

En una escena básica, consideramos la estructura de *NLOS* que se ha seguido en trabajos previos: un objeto oculto frente a un muro difuso. Aunque argumentamos a favor del uso de escenarios complejos, la regularidad con la que cambian los diferentes parámetros que afectan a la reconstrucción ayudan a evaluar sus efectos en las reconstrucciones, permitiendo tratar sus deficiencias de forma independiente.

Además, extendemos estas escenas con superficies con las que interacciona la luz, añadiendo ambigüedad en la información capturada que dificulta las reconstrucciones precisas. En la Figura 3.1 vemos la parte oculta de este tipo de escenas con las

extensiones de suelo y caja. Estas escenas son lo suficientemente sencillas como para generarlas automáticamente sin intervención humana en base a una descripción de sus características. Esto es ideal para obtener el mayor número de escenas posible en un tiempo razonable tan solo variando alguno de los parámetros las definen.

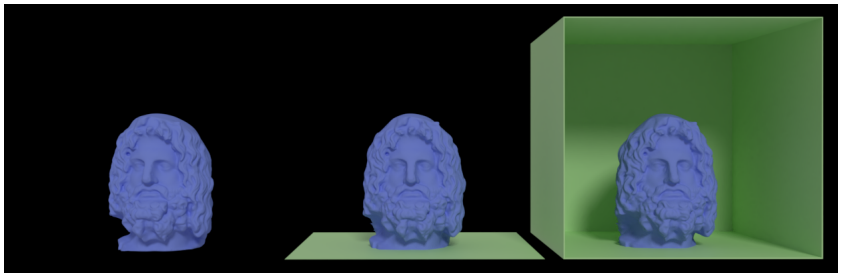
Esperamos que estas escenas, junto con técnicas de *data augmentation*, sean clave para aplicar *machine learning* en *NLOS imaging*, ya sea ideando métodos de reconstrucción nuevos, creando nuevos filtros más robustos para los ya existentes o reduciendo los efectos de las interferencias por camino múltiple, como se ha demostrado ya en trabajos previos (Marco et al., 2017; Su et al., 2018).

Objetos ocultos

En la Figura 3.2 representamos los diferentes objetos que situamos en nuestras escenas. Se trata de geometrías que creamos con este objetivo o que están disponibles de forma pública. Las complejidad variable que presentan será de utilidad para demostrar la precisión de los diferentes algoritmos de reconstrucción, encontrando qué características son más relevantes en la calidad final de los resultados.

Parámetros de escena

En una escena básica controlamos los parámetros: objeto oculto, distancia del plano visible al objeto, tamaño del objeto, entorno en el que se coloca, y su material. Mantenemos el tamaño del objeto oculto alrededor de 80cm en todos los objetos de forma que sus diferentes características se puedan comparar de forma independiente. Aunque parezca poco intuitivo que tanto un árbol como un conejo tengan el mismo tamaño, en la práctica podemos considerar que el sistema no tiene unidades y es independiente de la escala. En la Tabla A.5 de los anexos, resumimos los parámetros utilizados para todas las escenas básicas.



Flotando En suelo En caja

Figura 3.1: Parte oculta de los tres tipos de escena básica, de izquierda a derecha: flotando, con suelo y en una caja. Las tres imágenes están tomadas desde la posición del plano difuso que se encuentra en la parte visible.

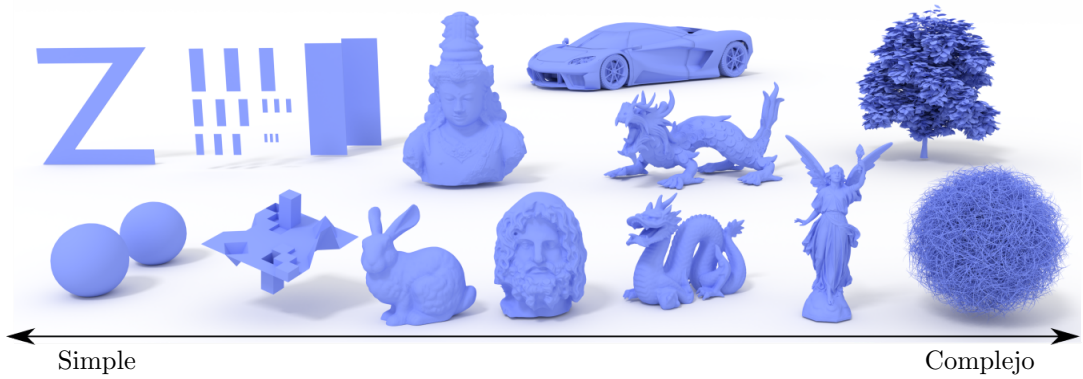


Figura 3.2: Objetos ocultos en las escenas básicas con complejidad creciente de izquierda a derecha.

3.2.2. Escenas complejas

Utilizar escenas simples para demostrar la validez de los métodos, como se ha hecho hasta ahora en trabajos previos, es útil, pero no permite evaluar con precisión su comportamiento ante la complejidad de escenarios que es posible encontrar en el mundo. Por tanto, preparamos un conjunto de escenas complejas con las que realizar este tipo de evaluaciones y validar nuevos algoritmos de reconstrucción.

Las escenas complejas que preparamos provienen de recursos online, tanto de acceso libre[1] como de pago. Las escenas que es posible encontrar están típicamente pensadas para resultar estéticamente agradables desde un punto de vista concreto, y no necesariamente resultan de interés como escenarios *NLOS*. Por tanto, modificamos escenas tanto de interior como de exterior situando elementos de interés para la reconstrucción, como mobiliario, elementos arquitecturales o personas en las primeras, y vehículos, edificios y viandantes en las segundas. Ambos representan situaciones en las que el uso de *NLOS imaging* es claro, como situaciones de rescate en interiores o seguridad en vehículos autónomos. En la Figura 3.3 mostramos las regiones ocultas de las escenas complejas que incluimos en el dataset. La variedad de los escenarios complejos obliga además a realizar las capturas sobre planos con diferentes orientaciones o con obstáculos adicionales en la región visible, suponiendo retos adicionales en la reconstrucción y reforzando las posibles diferencias entre algoritmos de reconstrucción.

[1] Varios artistas de `blendswap.com` y recursos de `benedikt-bitterli.me/resources/` (Bitterli, 2016)

Figura 3.3: Escenas complejas del dataset. Incluimos simulaciones de entornos de interior y exterior de diversa complejidad.

3.3. Iluminación y captura

La forma canónica de obtener información en *NLOS imaging* consiste en iluminar y capturar el perfil transitorio de la radiancia en puntos de la región visible de la escena. Para maximizar la información adquirida sobre la escena oculta, debemos obtener cuantos más pares de iluminación y captura del muro visible como sea posible. Cada par proporciona nueva información útil para la reconstrucción y mejora la robustez ante superficies ocluidas dentro de la propia región oculta. Por supuesto, existen limitaciones en el número de puntos que es posible o práctico obtener, tanto por tiempo como por espacio. Una captura exhaustiva combina todos los puntos de iluminación y de captura en una cuadrícula $N \times N$, produciendo una imagen transitoria 5-dimensional. El número de puntos que se puede capturar de forma exhaustiva es limitado por el crecimiento exponencial de capturas necesarias, alcanzando un millón de capturas con una cuadrícula de tamaño 32×32. Además, realizar capturas en el mundo real siguiendo estos patrones conlleva problemas adicionales, siendo necesario mover de forma precisa tanto la iluminación como el punto capturado, lo que requiere calibraciones precisas y complejas. En lugar de limitar las simulaciones del dataset a este tipo de patrones exhaustivos, nos fijamos en las estrategias seguidas por distintos métodos de reconstrucción, que no siempre son compatibles entre sí, para proporcionar los datos más completos y variados posibles.

Una estrategia recurrente en trabajos previos consiste en capturar un único punto de la cuadrícula, iluminando el resto (Buttafava et al., 2015; Liu et al., 2018). Es una opción relativamente simple de capturar utilizando *SPADs* unidimensionales y redirigiendo el foco del laser por medio de un espejo para realizar la captura, limitando así las partes móviles del sistema. Ya es posible encontrar *SPAD arrays* capaces de

capturar decenas de puntos de manera simultánea, y se espera alcanzar los cientos de puntos en 2019. Utilizando tales dispositivos, la velocidad de las capturas será mucho mayor, requiriendo de una única captura con un solo punto de iluminación, equivalente a iluminar varios puntos y capturar uno por la reciprocidad de Helmholtz. El principal problema de esta estrategia es que captura una variedad de información angular reducida, siendo especialmente débil frente a oclusiones como representamos en la Figura 3.4.

La última estrategia que añadimos se basa en patrones confocales, en los que el láser y la cámara enfocan al mismo punto. La ventaja principal de este tipo de capturas es que existe una solución cerrada para obtener reconstrucciones de forma eficiente realizando una convolución en espacio de Fourier (O'Toole et al., 2018), lo que denominaremos *Fourier-based backprojection*. Esta estrategia tiene la ventaja de ser capaz de recuperar superficies con materiales retroreflectantes por el incremento de cantidad de luz que regresa directamente al sensor, algo especialmente útil, por ejemplo, para visión en vehículos, donde las señales de tráfico o los chalecos retroreflectantes destacan más.

Otra consideración con respecto a los patrones de reconstrucción es su tamaño y su posición. Sus dimensiones definen una apertura virtual que delimita las superficies que pueden ser reconstruidas correctamente con métodos basados en *backprojection* (Liu et al., 2019a). En general, sólo será posible recuperar superficies cuya normal intersecta la región capturada. En métodos que utilizan *Fourier-based backprojection* tienen otra limitación aún mayor, reconstruyendo únicamente el volumen producido por la extrusión de la región capturada, que debe ser lo suficientemente grande como para cubrir la región oculta. El tamaño del plano visible en nuestras escenas básicas es de 1,6m, pero limitamos las regiones de captura a 1,0m de forma que contengan los objetos ocultos sin demasiado espacio sobrante independientemente del método de

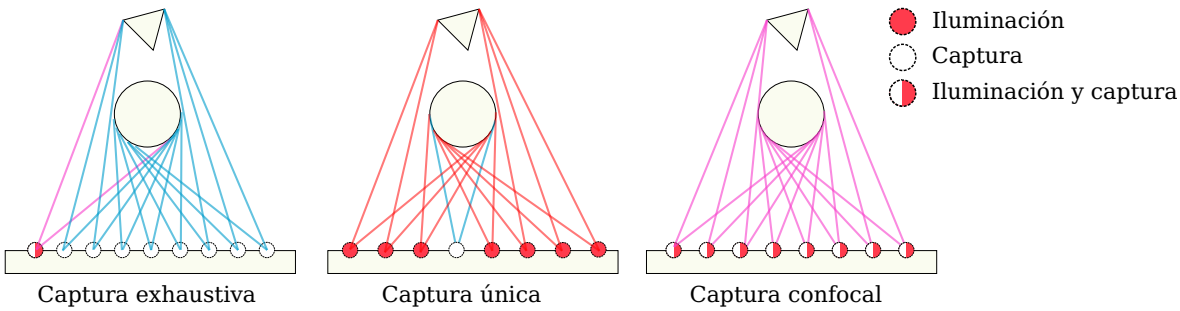


Figura 3.4: Vista alzada de los patrones de captura utilizados en el dataset. En captura única, aunque múltiples puntos iluminados alcanzan la región ocluida triangular, la luz no puede alcanzar el punto de captura, haciendo imposible recuperar esa información. El patrón de captura exhaustiva contiene todas las combinaciones de punto iluminado y capturado, incluyendo capturas confocales. En captura confocal, solo se capturan los puntos que se iluminan simultáneamente, permitiendo detectar información de la región oculta ocluida.

recontrucción. En las escenas básicas capturamos una región frente al objeto oculto de forma que lo cubra completamente. Sin embargo, en escenas en las que el objeto se encuentra sobre el suelo, la región capturada no puede cubrirlo completamente, dejando un margen con el suelo que será imposible reconstruir usando *Fourier-based backprojection*.

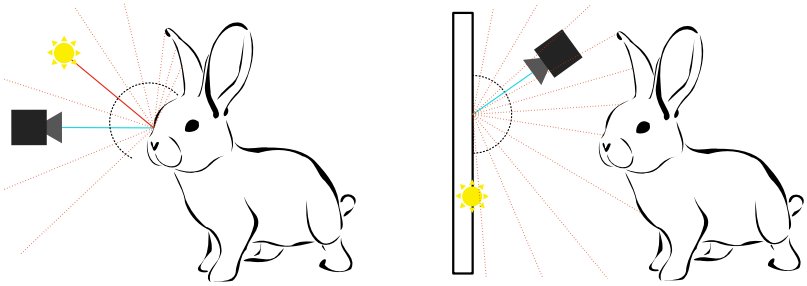
En escenarios complejos no es posible capturar regiones ideales, o incluso apropiadas, para reconstrucción en todos los casos. Como buscamos complejidad y generalidad esto es deseable, aportando un reto adicional en reconstrucción que además es representativo de situaciones reales. Elegimos regiones de captura apropiadas teniendo en cuenta estas consideraciones para recuperar información en cada escena, pero frecuentemente no son ideales. Especialmente para métodos basados en *Fourier-based backprojection*, para los que será imposible recuperar información útil en algunas de las escenas.

Finalmente, decidimos el número de puntos que simulamos en cada escena, independientemente del tamaño de la región o la estrategia de la captura. Establecemos el límite en 65.536 pares, proporcionando la suficiente información de cada escena sin exceder tiempos de computación y espacio de almacenamiento razonables. Esto supone utilizar cuadros de 16x16 puntos en iluminación y captura con patrones exhaustivos, 256x256 puntos de captura, o iluminación, en patrones con punto único y 256x256 puntos confocales en patrones confocales.

3.4. Simulación

La herramienta principal que usamos en este proyecto es el motor de renderizado transitorio de Jarabo et al. (2014). Este software utiliza la API de trazado de rayos de alto rendimiento *Embree* (Wald et al., 2014) junto con técnicas de muestreo específicas para transporte de luz transitorio para obtener simulaciones físicamente correctas de las escenas de la forma más eficiente posible. Sin embargo, su uso está enfocado a escenas en las que el sujeto que se intenta observar se encuentra en la línea de visión de la cámara, típicamente para simular vídeos del transporte de luz en la escena. La luz que alcanza el sensor virtual en escenas *NLOS* proviene de fuentes indirectas, más complejas de simular con precisión debido a la necesidad de emplear métodos de integración numérica.

Típicamente, cuando simulamos una escena en estado estacionario, utilizamos una cámara virtual con un sensor y campo de visión similares a los de una real. En escenas sin línea de visión, nos interesa iluminar y capturar con precisión los puntos de la región capturada en la escena. En la Figura 3.8 representamos esta diferencia. Podemos



(a) Escena LOS. *(b) Escena* NLOS.

Figura 3.5: Las escenas con línea de visión suelen tener un elemento que las ilumina de forma directa, haciéndolo más sencillo de simular. Sin embargo, en escenas NLOS *toda la iluminación proviene de fuentes indirectas (la luz rebota primero en la parte visible y luego en las partes ocultas), haciéndolas mucho más costosas de simular.*

simularlo fácilmente utilizando cámaras virtuales con un sensor de tamaño 1×1 y campo de visión de $0°$, capturando un punto diferencial, y situando una fuente de luz virtual sobre el plano visible, simulando el efecto de lanzar un láser sobre su superficie. Esto es ineficiente utilizando la versión pública del software, que fuerza a realizar múltiples simulaciones, cada una de un par de puntos independiente. Para simular todas las escenas de las que consta el dataset de forma sencilla y eficiente, extendemos el motor de renderizado para que tome como entrada valores específicos a escenas *NLOS*. De media, cada escena requiere 17 horas de cómputo, variando con la complejidad. Especificamos más detalles en el Anexo B.

Capítulo 4

Análisis de los datos

En esta sección analizamos el dataset generado, evaluando su validez en tiempo de respuesta temporal, así como comparando los datasets generados con capturas reales de escenarios NLOS:

4.1. Corrección de las simulaciones

Para asegurar la corrección de las simulaciones que produce el motor de render, creamos una escena muy sencilla, con tan solo dos planos enfrentados, como describimos en la Figura 4.1. Simulamos un punto centrado en el muro visible, de forma que conocemos los instantes de tiempo en los que el sensor deberá recibir información. Observamos que la simulación se corresponde con lo esperado, recibiendo el primer impulso a los 2,5m y recibiendo radiancia del resto del plano durante unos instantes, al inicio provenientes del punto central del muro y al final de las esquinas. La luz entonces viaja de vuelta al plano oculto, alcanzándolo a los 3,25m, donde rebota de vuelta al muro visible, alcanzando el sensor de nuevo a los 4,5m. El rango de tiempo en el que encontramos radiancia de este rebote es mayor debido a que en vez de recibir luz de una fuente puntual, la fuente de este rebote es toda la energía que alcanzó el muro visible en el tercer rebote. Esto se repite en los rebotes sexto y séptimo. La señal recibida en los rebotes 4º y 5º es 0 debido a que la escena presenta dos planos paralelos que no pueden recibir luz reflejada sobre sí mismos. La respuesta de la señal disminuye exponencialmente con el paso del tiempo con la ley de la inversa del cuadrado al distribuir la energía en ángulos sólidos cada vez mayores (nótese que el eje vertical tiene una escala logarítmica). En la cola de los rebotes quinto (entre 6 y 6,5 segundos) y séptimo (entre 8 y 9 segundos) vemos como la señal oscila rápidamente, recibiendo radiancia de forma intermitentemente. Este ruido es común en las simulaciones por Monte Carlo del render, que convergen a soluciones correctas muy lentamente. En este caso, aún con un número muy elevado de muestras (8388608), apreciamos que el ruido

sigue siendo muy relevante en la simulación.

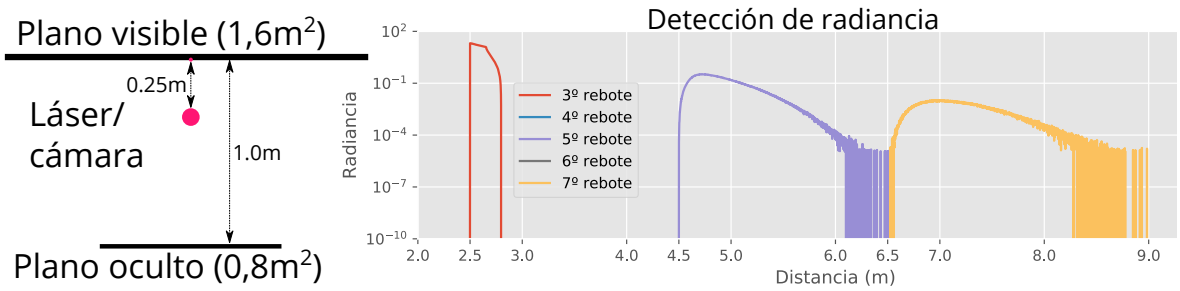


Figura 4.1: Escena simplificada con un único punto de captura e iluminación centrado. En la gráfica, indicamos la radiancia detectada en cada instante de tiempo en escala logarítmica. Al aumentar el número de rebotes la intensidad decae, pero se recibe durante más tiempo debido a los múltiples caminos de los que proviene.

4.2. Datos reales

Capturar información del transporte transitorio de la luz del mundo real presenta un gran reto. Cualquier sistema de medición físico tiene errores en la lectura, y la escala con la que se trabaja es extremadamente sensible a ellos. En particular, sufren de un ratio señal/ruido muy bajo debido a los tiempos de exposición extremadamente cortos con los que trabajan y al ruido provocado por la sensibilidad de la propia electrónica. Comparando la información en la Figura 4.2, vemos que los datos siguen un patrón similar, como es de esperar en una simulación fisicamente correcta. Principalmente vemos como con la distancia (tiempo) la intensidad decae. Sin embargo, en captura real no es posible detectar diferencias de radiancia en valores muy bajos, mientras que en simulación la precisión es arbitraria (excepto por límites de precisión de máquina y tiempo de cómputo).

En la Figura 4.3 comparamos imágenes transitorias 2D (incluyendo únicamente una columna de los puntos capturados sobre el muro) en captura y simulación con patrones confocales. Destacamos que la relación señal/ruido en capturas reales de escenas ocultas difusas presentan un ruido de sal y pimienta muy pronunciado, tanto que enmascara la información del tercer rebote si no se realizan más repeticiones de la captura para incrementar la señal. Con materiales retroreflectantes, las capturas confocales reciben una información significativamente más limpia, haciéndolo interesante en situaciones de conducción donde los materiales retroreflectantes son más comunes. Sin embargo, las simulaciones presentan datos mucho más claros que las capturas, mientras que en simulación encontramos el límite en la precisión de la máquina y el tiempo de cómputo que estemos dispuestos a utilizar..

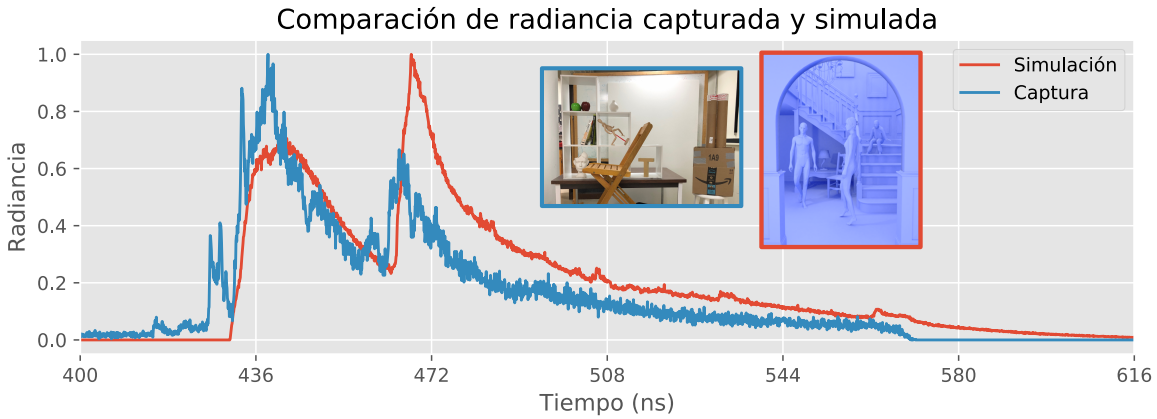
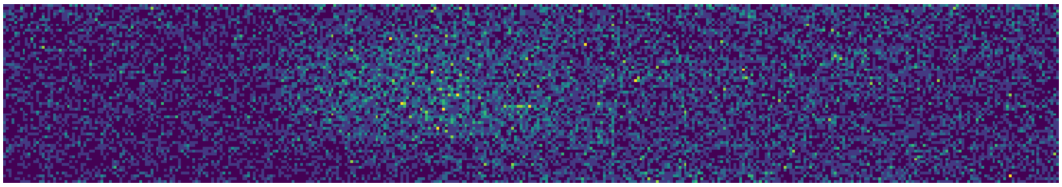
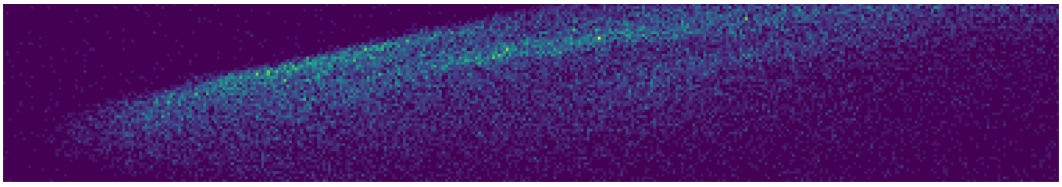


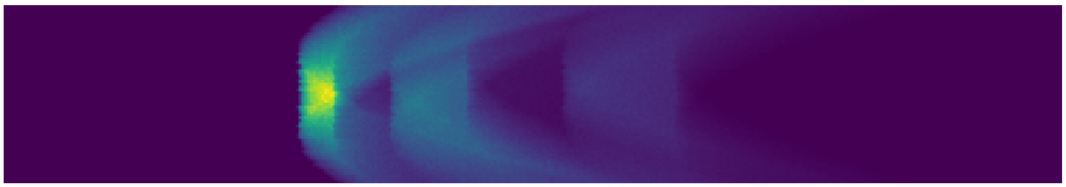
Figura 4.2: Perfiles temporales de la radiancia para una escena compleja en simulación y captura. Cualitativamente, observamos que la distribución que sigue la radiancia es similar, con mayor intensidad al principio y decayendo con el paso del tiempo. También observamos que la captura real es más ruidosa que la simulación, algo que se debe tener en cuenta al utilizar datos simulados. Datos capturados de Liu et al. (2018). Nótese que las escenas son distintas.



(a) Captura confocal con un objeto oculto plano difuso



(b) Captura confocal con un objeto oculto plano retroreflectante



(c) Simulación confocal con un objeto oculto plano difuso

Figura 4.3: Comparativa del ruido en imágenes transitorias confocales de un objeto difuso y uno retroreflectante y una simulación de un objeto difuso.

Capítulo 5

Validación con reconstrucciones

En este capítulo analizamos nuestro dataset como benchmark para contrastar los dos métodos más relevantes en el estado del arte de *NLOS imaging* y un filtro nuevo para *backprojection* que presentamos. Primero presentamos un set de métricas de error para evaluar la calidad de la reconstrucción de geometría no visible. Una vez presentadas dichas métricas, evaluamos dichos métodos con nuestro dataset.

5.1. Métricas de error

Existen multitud de métricas de error diseñadas para distintos aspectos de la imagen por computador. Si bien suelen estar orientadas a la comparación de imágenes 2D, sus extensiones 3D son triviales, por lo que merece la pena analizarlas. Idealmente, querríamos poder aprovechar todos los años de experiencia que se tiene en el campo de visión para imagen 2D y aplicar sus métricas a volúmenes para evaluar la calidad de las reconstrucciones. Las métricas más sencillas se basan en la media de los errores cuadráticos medios de los píxeles de dos imágenes (*MSE*) o en la proporción máxima de señal a ruido (*PSNR*) basada también en la anterior. Sus expresiones, en tres dimensiones, siendo $\hat{Y}$ un volumen de vóxeles reconstruido con dimensiones x, y, z, e Y su volumen de referencia, son:

$$\mathrm{MSE}(Y, \hat{Y}) = \frac{1}{xyz} \sum_{i=1}^{x} \sum_{j=1}^{y} \sum_{k=1}^{z} \left(Y(i,j,k) - \hat{Y}(i,j,k)\right)^2 \tag{5.1}$$

$$\mathrm{PSNR}(Y, \hat{Y}) = 10 \log_{10} \left(\frac{1}{\mathrm{MSE(Y, \hat{Y})}} \right) \tag{5.2}$$

Los volúmenes de referencia contienen valores binarios representando los vóxeles que se encuentran *ocupados* por una superficie con valor 1 y los vacios con 0. Por tanto, para utilizar estas expresiones es necesario binarizar las reconstrucciones, que generalmente contienen valores reales por vóxel. El problema con este tipo de métricas para *NLOS*

imaging es que están planteadas para evaluar diferencias entre valores individuales, y no qué posiciones se encuentran ocupadas. Esto se hace aparente cuando, por ejemplo, los vóxeles de una reconstrucción están desplazados una unidad en el eje Y. En esos casos, las métricas fallan, indicando valores muy elevados cuando el error real puede ser muy pequeño.

Una alternativa que evita esta problemática consiste en evaluar directamente las reconstrucciones como nubes de puntos utilizando métricas como la distancia de Hausdorff, que intuitivamente representa cómo de lejos se encuentran dos sets de ser el mismo:

$$d_H(c_x, g_t) = \text{máx}\left\{p_H(c_x, g_t), p_H(g_t, c_x)\right\} \tag{5.3}$$

$$p_H(X, Y) = \sup_{x \in X} \inf_{y \in Y} d(x, y) \tag{5.4}$$

donde $d(x, y)$ es la distancia euclídea entre x e y, ínf es el valor ínfimo, sup el supremo, $p_H(X, Y)$ la distancia mayor de los puntos más cercanos del set X al Y, la nube de puntos reconstruida c_x y la de referencia g_t. Como $p_H(c_x, g_t)$ no es una distancia simétrica, se considera el máximo de las dos distancias parciales. El trabajo de Klein et al. (2018) propone una métrica basada en esta distancia pero para mallas de triángulos en reconstrucciones *NLOS*. Aunque tiene la ventaja adicional de ser independiente de la escala de la reconstrucción, esta métrica requiere del uso de algoritmos de triangularización sobre los volúmenes reconstruidos, añadiendo una nueva fuente de error. Además, para obtener resultados significativos con los métodos de reconstrucción actuales, la malla de referencia debería contener tan solo las superficies orientadas a la región capturada, requiriendo un paso adicional para descartar las superficies no deseadas. Finalmente, no consideramos este tipo de métricas por la complejidad de obtener valores de referencia de las superficies, sin los cuales carece de sentido utilizarla ya que proporciona resultados muy elevados en cualquier caso, y la sensibilidad a valores espúreos en la reconstrucción que hacen de palanca afectando al resultado en exceso.

Buscando un término medio que permita evaluar reconstrucciones sin incurrir en errores adicionales por triangularización, proponemos el uso de métricas cuantitativas y cualitativas basadas en la profundidad de la reconstrucción desde el plano capturado. Esto reduce las dimensiones de las reconstrucciones de un volumen 3D a una imagen de profundidad 2D, ignorando oclusiones en la reconstrucción:

$$\text{DM}(Y, i, j) = \begin{cases} \arg\max_{k \epsilon [0,z]} Y(i, j, k), & \text{si} \sum_{k=0}^{z} Y(i, j, k) > 0 \\ \infty, & \textit{sino} \end{cases} \tag{5.5}$$

Establecemos distancia infinita cuando no hay ningún vóxel mayor que cero en

profundidad, indicando que no hay reconstrucción. Obtenemos imágenes de referencia a partir de volúmenes voxelizados de las escenas ocultas con la misma resolución que la misma resolución que las reconstrucciones. Consideramos el error en la reconstrucción entre las dos imágenes como el valor absoluto de su diferencia, teniendo en cuenta los valores infinitos por separado:

$$\text{DMError}(Y, \hat{Y}, i, j) = \begin{cases} \infty, & \text{si } DM(Y, i, j) = \infty \\ z, & \text{si } DM(\hat{Y}, i, j) = \infty \\ |DM(Y, i, j) - DM(\hat{Y}, i, j)|, & \text{sino} \end{cases} \quad (5.6)$$

De esta forma tenemos en cuenta el error de las reconstrucciones solo en las regiones definidas en la referencia, asignando error máximo (z) en regiones no reconstruidas. El error absoluto da una idea cualitativa de la calidad de la reconstrucción que, al ser una imagen, permite ver con facilidad el resultado en diferentes regiones de interés. En base a este error podemos utilizar la métricas como las descritas inicialmente (MSE) para compararlas de forma cuantitativa, teniendo especial cuidado con los valores infinitos que indican que no hay reconstrucción y, por tanto, no deberán afectar al resultado. Vemos ejemplos de los resultados a continuación en las Figuras 5.3, 5.5 y 5.8.

Un problema común a todos los métodos actuales es la elección del valor umbral a partir del cual se asume que un vóxel *existe* en una reconstrucción (ver Figura 5.2). Los mejores resultados se obtienen con intervención humana para maximizar las superficies reconstruidas correctamente y minimizar vóxeles erróneamente clasificados como superficies. Esto es posible porque sabemos qué se espera encontrar en el volumen, dando resultados subjetivos. En general, esto no es posible, haciendo necesario establecer el valor de forma automática, o emplear métodos que obtengan resultados directamente sobre geometría, como ya hacen los trabajos de Iseringhausen and Hullin (2018); Tsai et al. (2019). En definitiva, utilizar métricas para comparar reconstrucciones tridimensionales de escenas *NLOS* es muy complejo, y será necesario encontrar métricas cuantitativas más apropiadas según mejoren las reconstrucciones, por ejemplo, considerando oclusiones que no permitan reducir la dimensionalidad de los resultados.

5.2. Resultados en reconstrucción

La reconstrucción de geometría no visible es un problema complejo, costoso de computar y todavía en desarrollo. Utilizamos el dataset para comparar los dos métodos de reconstrucción más usados hasta la fecha y caracterizamos las diferencias que encontramos.

5.2.1. *Filtered backprojection*

Filtered backprojection (Velten et al., 2012; Arellano et al., 2017) es el método más utilizado para recuperar información geométrica a partir de capturas de transporte de luz transitorio. Está basado en las técnicas de tomografía computacional utilizadas frecuentemente en medicina para obtener imágenes del interior de nuestro cuerpo. El algoritmo reconstruye un mapa de probabilidades 3D en base al tiempo de vuelo de la señal en la imagen transitoria. Después, es necesario aplicar un filtrado sobre el resultado para obtener la reconstrucción final (ver Sección C.2 en los anexos). Es robusto al ruido en la señal, y eficiente en memoria (aunque tiene coste $\mathcal{O}(n^3)$ se puede subdividir en bloques pequeños fácilmente). Sin embargo, es muy lento, sumando para cada vóxel del mapa de probabilidades 3D el valor que le corresponde de cada captura individual de la imagen transitoria (complejidad $\mathcal{O}(n^5)$). En la Figura 5.1 incluimos un resumen del método.

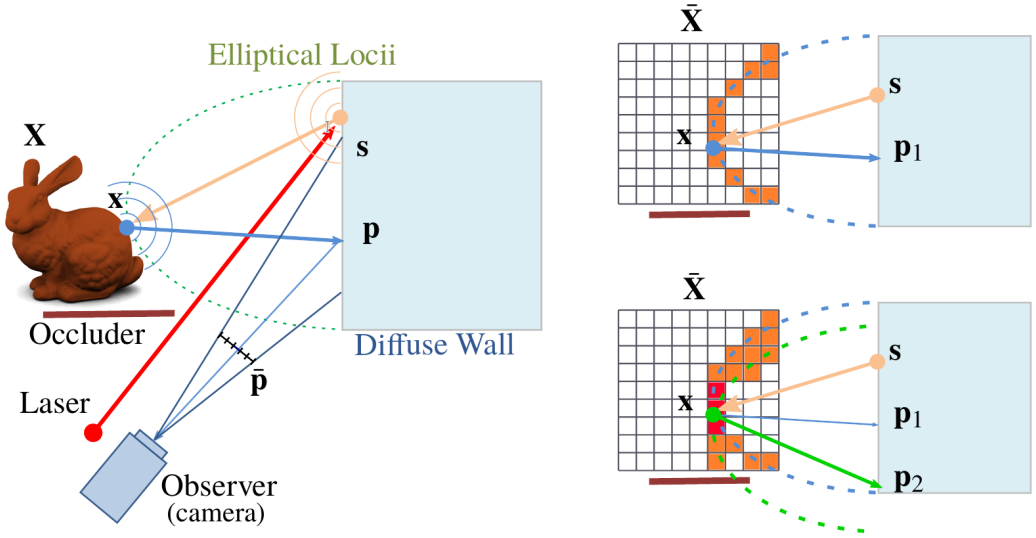


Figura 5.1: Resumen del método de backprojection (figura de Arellano et al. (2017)). A la izquierda: ilustración de la escena. Se emite un pulso láser hacia el muro visible, creando una nueva fuente de luz que ilumina la escena oculta. El reflejo de la luz en las superficies no visibles viajan de vuelta al muro visible, que observamos con la cámara. El tiempo de propagación desde un punto x de una superficie oculta forma un elipsoide con puntos focales en s y p. A la derecha: la intersección de varios de estos elipsoides define el mapa de probabilidad 3D de la geometría oculta. Posteriormente se filtra y se establece un umbral para obtener la reconstrucción final.

El trabajo de Arellano et al. (2017) convierte el problema en uno de intersección de elipsoides, y lo resuelve haciendo uso del pipeline gráfico de las *GPUs*. Por este motivo, no la utilizamos en nuestras pruebas, ya que no es sencillo diferenciar la fuente de errores en reconstrucción, que podrían provenir de la simulación, el algoritmo o la implementación.

En la Figura 5.3 mostramos los mapas de profundidad de reconstrucciones realizadas mediante una implementación propia del algoritmo básico en *GPU* (ver Anexo C para más detalles sobre la reconstrucción con este algoritmo y nuestra implementación en *CUDA*). Como filtro, en nuestras reconstrucciones utilizamos principalmente el laplaciano, pero en la Figura 5.4 incluimos resultados con el laplaciano de Gauss que, teóricamente logra resultados superiores en escenas reales (Laurenzis and Velten, 2014), y vemos que obtienen volúmenes muy similares. Incluimos más resultados en el Anexo A.

5.2.2. *Light Cone Transform*

Es una formulación del problema más eficiente en tiempo debido a que es posible computarla como una convolución en el dominio de Fourier O'Toole et al. (2018). Al contrario que *backprojection*, que es una aproximación, *LCT* es una forma cerrada para invertir el problema, y obtiene resultados más limpios cuando sus asunciones se cumplen. Por contra, tiene unos requisitos de memoria mucho mayores, es menos flexible en cuanto al volumen que puede reconstruir (como vimos en la Sección 3.3), y es menos robusto en general cuando sus asunciones no se cumplen, como en presencia de ruido e interferencias por caminos múltiples.

Para obtener reconstrucciones con este método, utilizamos la implementación pública incluida en el trabajo original de O'Toole et al. (2018) con pequeñas modificaciones para aceptar nuestros datos como entrada. En la Figura 5.5 incluimos ejemplos de resultados para las mismas escenas que en la Figura 5.3 (con captura confocal). Destacamos que, aunque este obtiene resultados más limpios al considerar un número bajo de rebotes, las interferencias por caminos múltiples causan artefactos no presentes en *backprojection* filtrado. Además, el volumen reconstruido es diferente en ambos casos, el de *LCT* estando limitado a un espacio encima del suelo correspondiente con la posición de la región capturada en el muro visible.

5.2.3. Filtro *LCT* para *backprojection*

Derivamos un nuevo filtro para *backprojection* en *NLOS imaging* en base a un análisis de la *Light Cone Transform*, la solución de forma cerrada para medidas confocales. El filtro que proponemos está basado en el filtrado de *Wiener*, y es teóricamente óptimo para medidas con ruido blanco. Aunque no demostramos este nuevo filtro para medidas no confocales, en la Figura 5.6 mostramos empíricamente que es posible obtener resultados válidos aplicándolo en volúmenes reconstruidos con otro tipo de medidas. Incluimos derivaciones detalladas del filtro en el Anexo D.

Idealmente, con este filtro obtendríamos resultados con las características de LCT a partir de volúmenes provenientes de *backprojection*, con la flexibilidad en la reconstrucción que conlleva. En la Figura 5.7 mostramos un problema de este filtro con las discontinuidades presentes en los volúmenes que, al realizar cálculos en espacio de frecuencias, genera *bloques* de gran intensidad en los extremos del volumen, haciendo que la reconstrucción fracase. En la Figura 5.8 comparamos los tres métodos que consideramos en el análisis en una escena compleja.

5.3. Discusión

En esta sección demostramos que los datos simulados son cualitativamente válidos en comparación con los que es posible obtener con captura real. Idealmente, dispondríamos de un dataset de capturas reales con algunos escenarios equivalentes a los que presentamos en simulación para realizar una comparación directa de sus valores y caracterizar más aspectos de la calidad de las simulaciones. Sin embargo, esto solo sería posible en escenas muy sencillas por la dificultad de digitalizar clones idénticos de la realidad. La calidad de los datos es, sin embargo mucho mayor, habiendo evitado el ruido que produce la electrónica de los sistemas de captura y la iluminación natural de los entornos, algo que, por contra, se deberá tener en cuenta al utilizar el dataset para evitar presentar métodos frágiles al ruido presente en la realidad.

También nos hemos cerciorado de que nuestro dataset es válido como herramienta para validar reconstrucciones, permitiendo evaluar diferentes parámetros sin tener que realizar capturas reales. Aunque no podemos realizar conclusiones definitivas en cuanto a cual de los dos pudiera ser superior, caracterizamos algunas diferencias de los métodos FBP y LCT y nuestro nuevo filtro. Aunque FBP es una aproximación y LCT una solución cerrada al problema, vemos que ambos consiguen reconstrucciones con una calidad similar. En general, LCT consigue volúmenes más limpios, y mantiene una calidad de reconstrucción constante en la escena. Por contra, los volúmenes de FBP tienen una gran cantidad de ruido que es necesario eliminar estableciendo un valor umbral apropiado que, sobre todo en escenas profundas, elimina valores que es deseable mantener. Nuestro nuevo filtro es interesante en cuanto a que presenta una formulación cerrada del problema. Sin embargo, no solo es computacionalmente costoso por estar basado en *backprojection*, sino que también es más caro de computar que los filtros laplaciano y laplaciano de Gauss utilizados típicamente sin conseguir resultados significativamente mejores y presenta el problema de las discontinuidades.

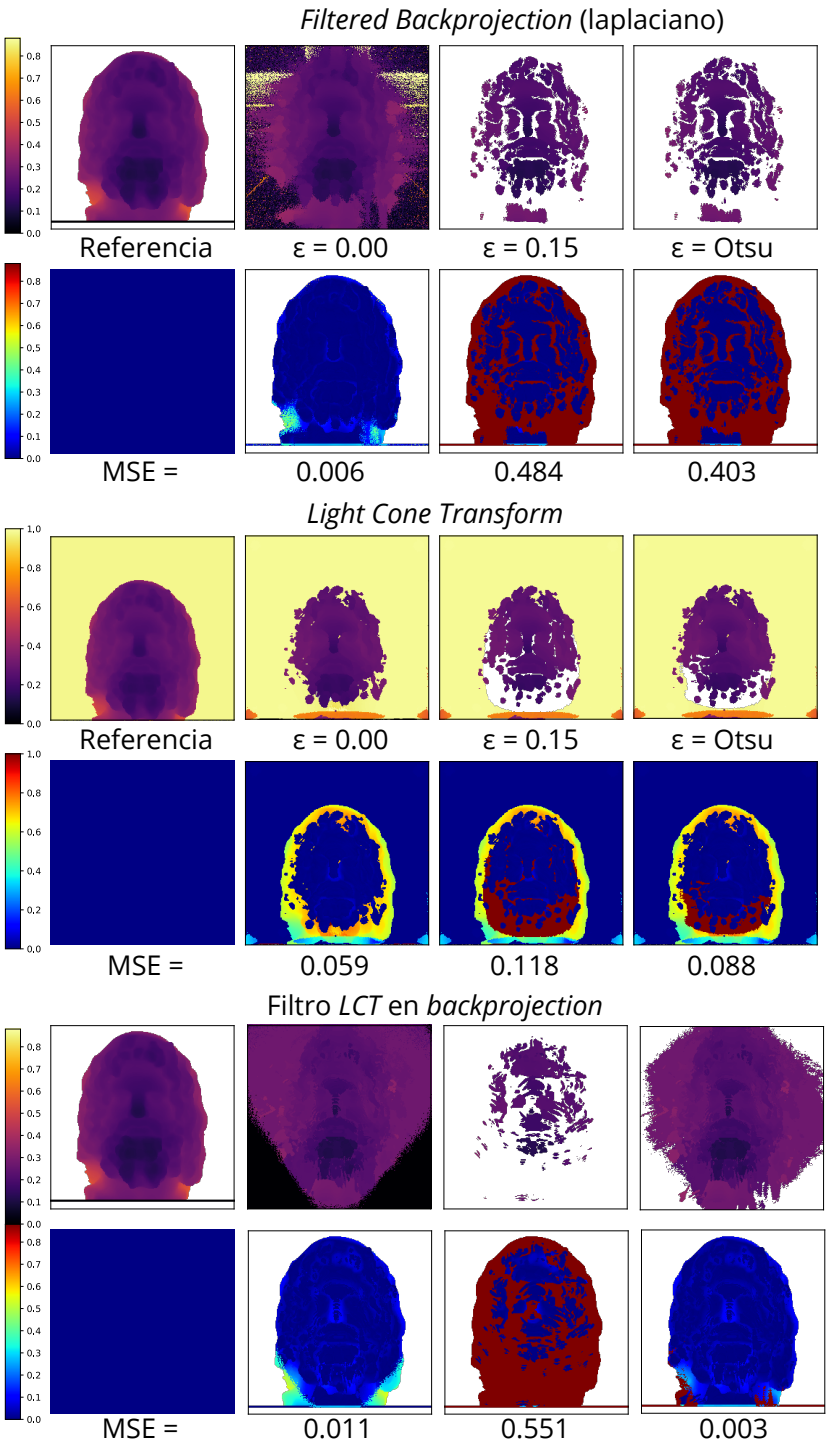


Figura 5.2: Busto de Serapis en una caja reconstruido con diferentes valores umbral ϵ. El valor de umbral 0 proporciona las reconstrucciones más detalladas, pero también las más ruidosas. Aumentarlo mejora la calidad de los resultados, pero son menos completos. Los métodos LCT y FBP no se comportan del mismo modo ante cambios en el umbral. En general, en nuestros experimentos, observamos que LCT da resultados más completos con un valor de umbral mayor que 0 mientras que FBP siempre da el resultado más completo con este umbral. Los resultados con Otsu tampoco son consistentes, dependiendo de la escena, patrón de captura, etc.

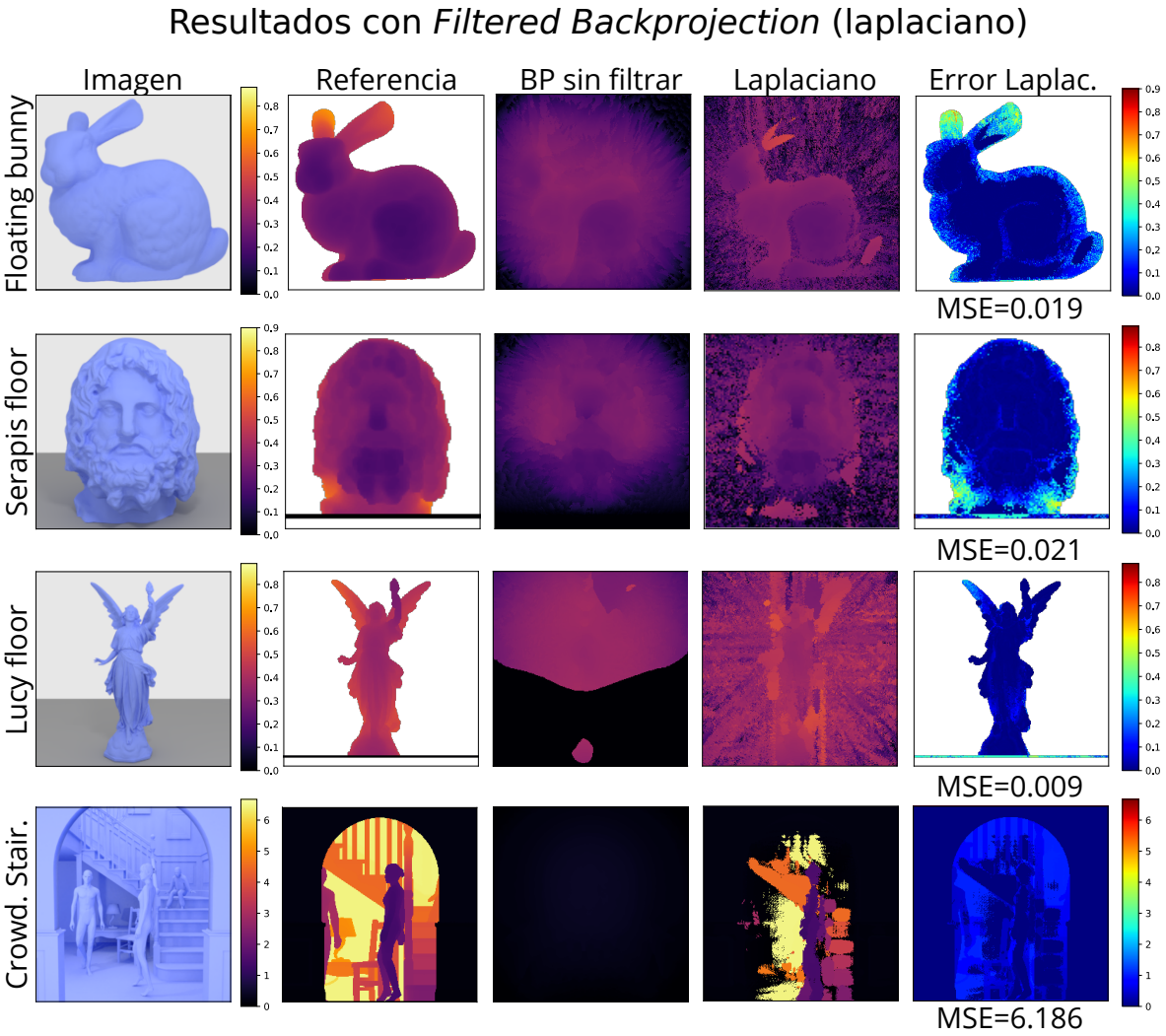


Figura 5.3: Ejemplos de reconstrucción de profundidad en varias escenas con valor de umbral $\epsilon = 0$. *El color blanco indica valores infinitos, es decir, no hay ningún vóxel en la dirección correspondiente, mientras que los valores negros indican valor* 0. *En la primera fila tenemos un conejo flotando en el escenario, en la segunda y tercera dos figuras sobre el suelo y finalmente una sala con gente y mobiliario tras un arco. Observamos como* backprojection *sin filtrar está desenfocado y no se puede discernir geometría correctamente. Una vez filtrado comienza a ser posible discernir las distintas superficies, aunque la calidad sea pobre. En los mapas de error vemos cómo de completas son las reconstrucciones, y las zonas más problemáticas en reconstrucción, como regiones cóncavas (e.g. orejas del conejo) u oclusiones (e.g. hombre tras el arco).*

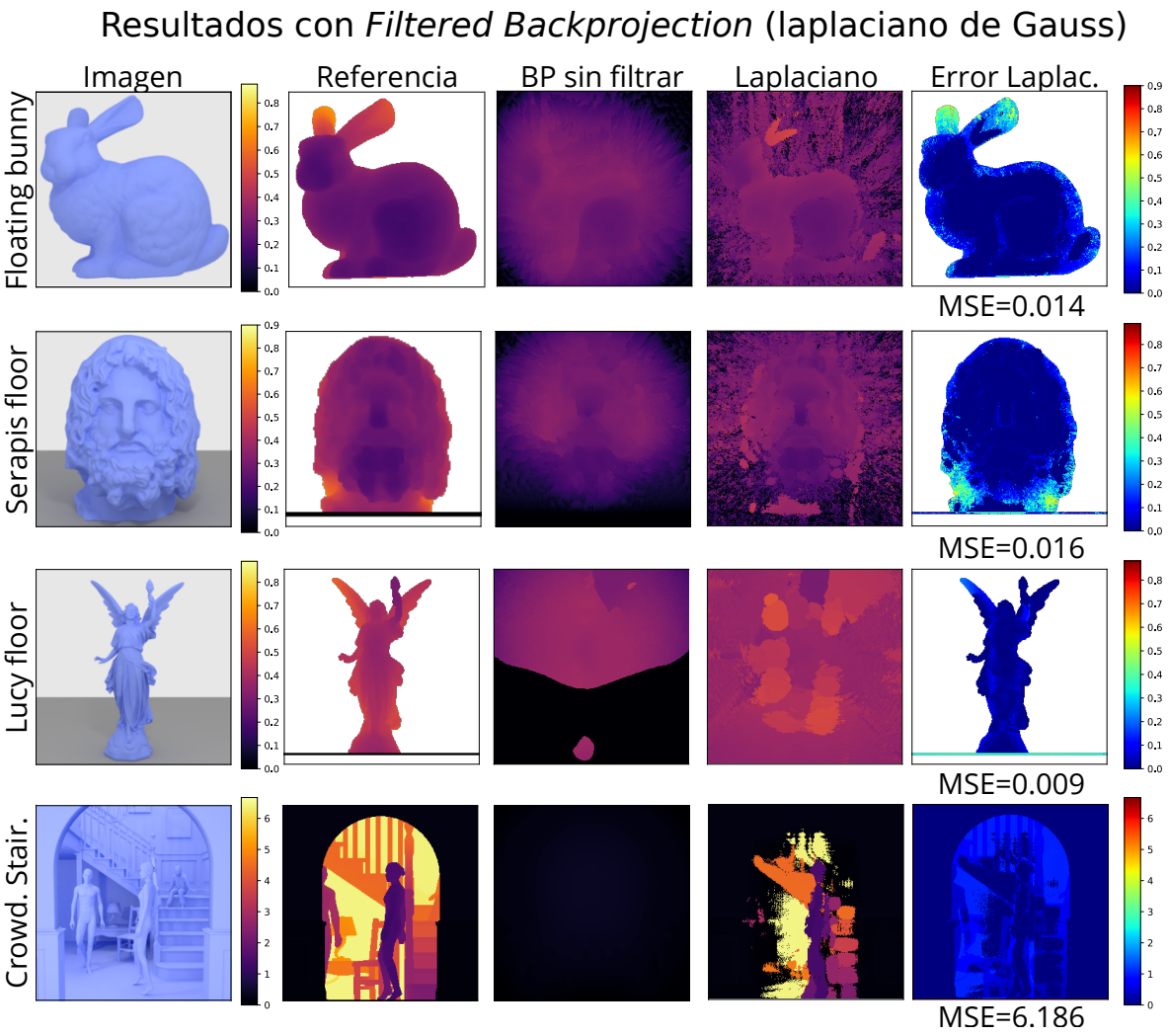


Figura 5.4: Reconstrucciones con el filtro laplaciano de Gauss. El valor umbral es $\epsilon = 0$.

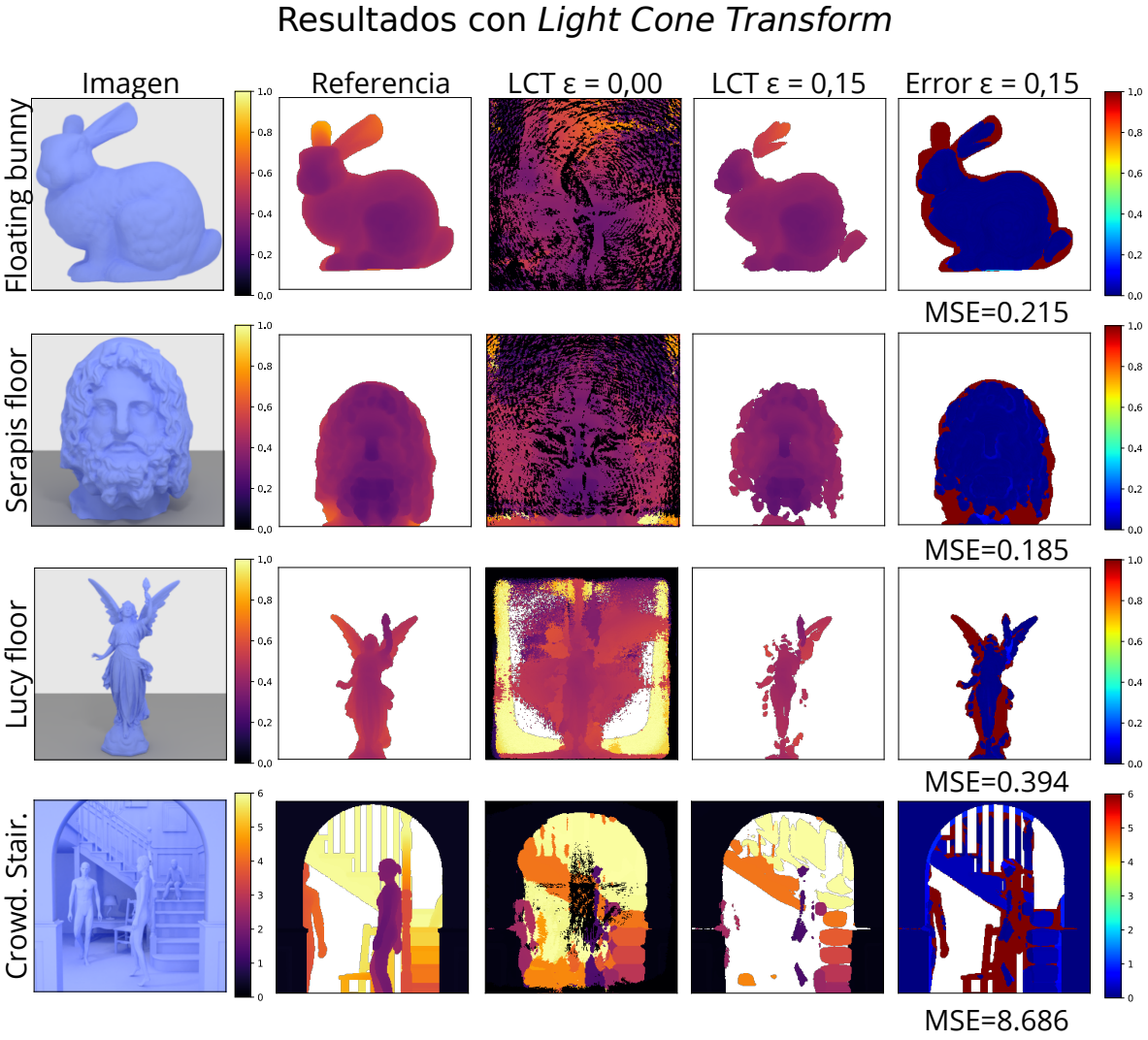


Figura 5.5: Ejemplos de reconstrucción de profundidad en varias escenas con el método basado en LCT de O'Toole et al. (2018). Incluimos reconstrucciones considerando dos umbrales, indicando que aunque este método da resultados más limpios que el anterior, sigue resultando necesario para evitar valores espurios. Las referencias en este caso son diferentes a las de la Figura 5.3, reconstruyendo el volumen delimitado por el área capturada, lo que excluye la parte más baja de las escenas con suelo.

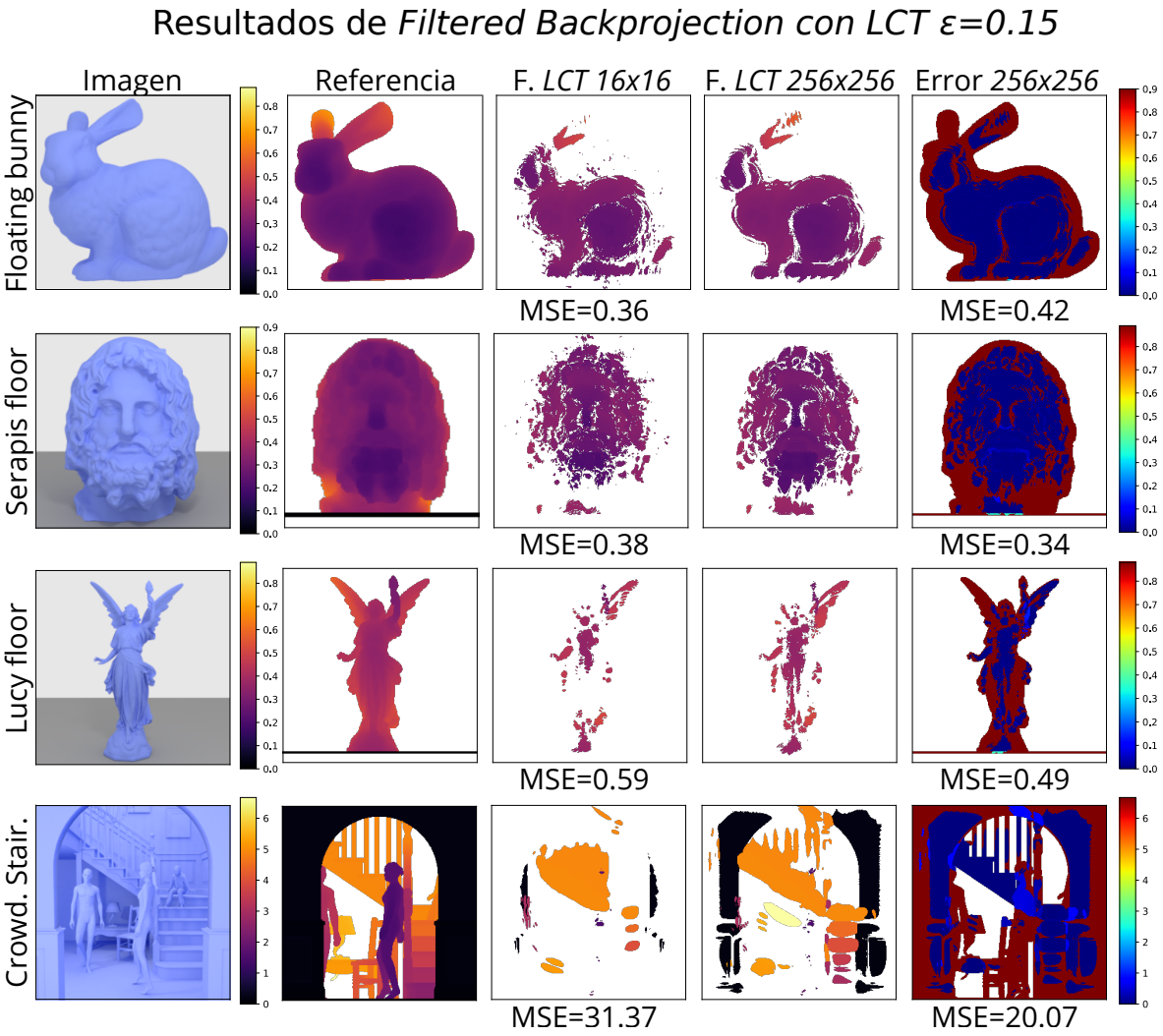


Figura 5.6: Resultados de backprojection utilizando el filtro LCT con dos patrones de captura diferentes. Solo demostramos su uso para patrones confocales, sin embargo, demostramos empíricamente que es válido en otros patrones. Solo incluimos el mapa de error para los patrones confocales, pero calculamos la métrica en ambos casos. En la escena compleja recortamos el fondo para evitar la discontinuidad que vemos en la Figura 5.7.

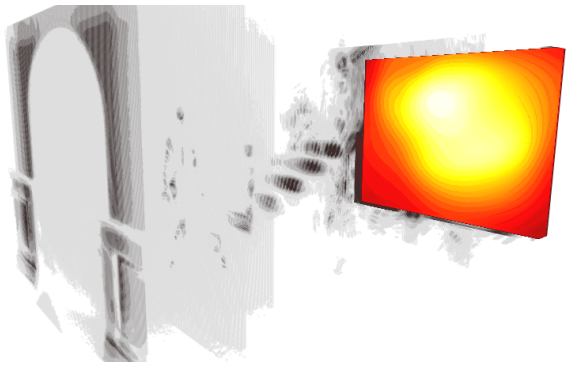


Figura 5.7: Reconstrucción de la escena "Crowded Staircase" con el filtro LCT para backprojection en la que se observa el problema de las discontinuidades. Si no se eliminan los extremos de la reconstrucción, al calcular el mapa de profundidades solo se ve el bloque del fondo debido a su elevada intensidad.

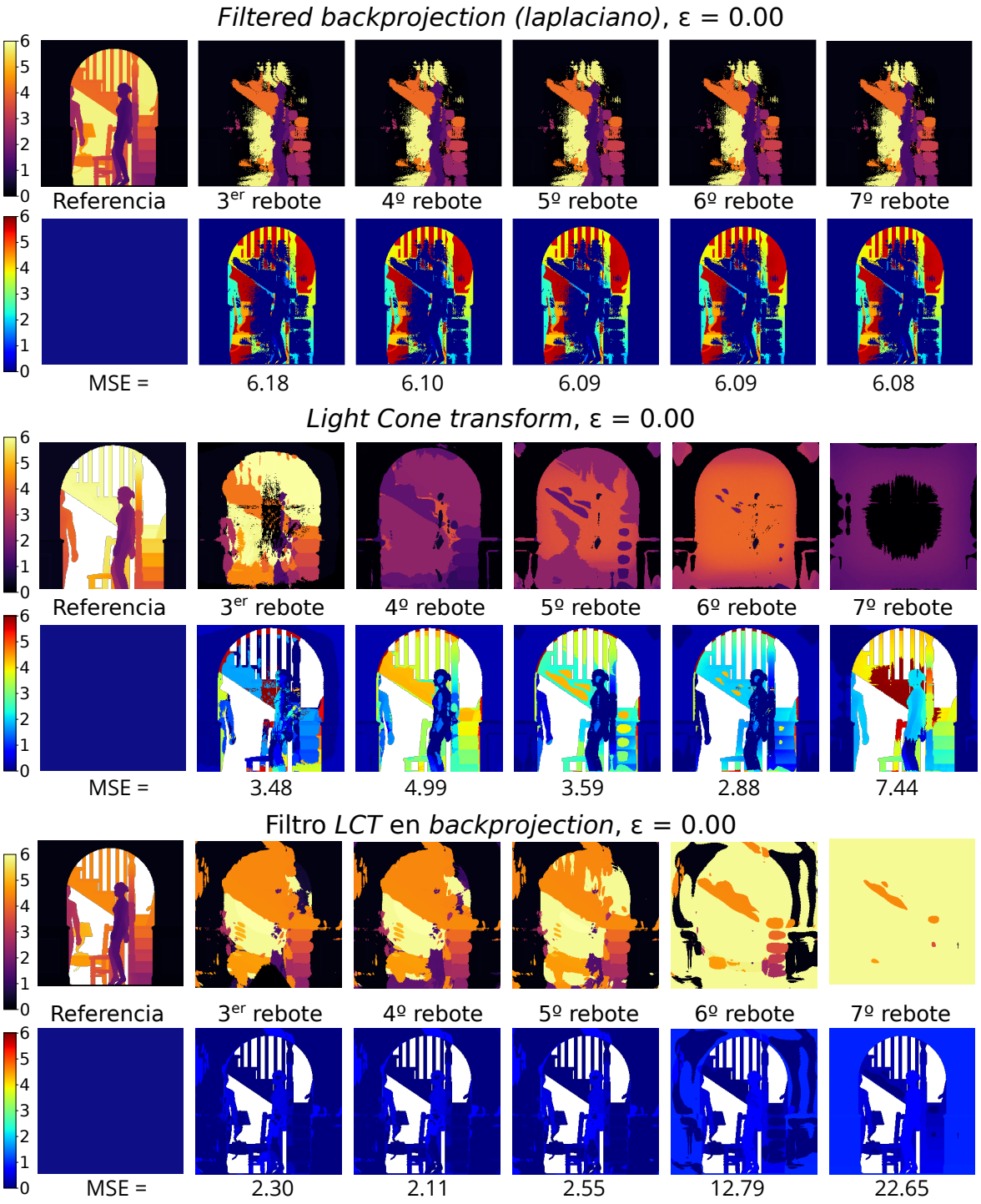


Figura 5.8: Comparación de resultados de los dos métodos que utilizamos sobre medidas confocales incrementando el número de rebotes que se consideran en la reconstrucción. Aunque las reconstrucciones del método LCT con pocos rebotes son más claras, FBP es mucho más robusto frente al ruido añadido por los rebotes de alto orden.

Capítulo 6

Conclusiones

Aportamos un dataset sintético de imagen transitoria para reconstrucción de escenarios sin línea de visión con una escala y complejidad sin precedentes en la comunidad. Este dataset servirá como herramienta para el desarrollo de nuevas técnicas de reconstrucción *NLOS*. Además, proponemos métricas para utilizar el dataset como evaluación de la calidad de métodos de reconstrucción de geometría en base a la profundidad. Mantenemos estos datos en una interfaz web pública a la comunidad, donde ya se está utilizando en nuevas publicaciones.

Reconocemos que utilizar datos simulados puede ser arriesgado si no se tiene en cuenta el comportamiento de los dispositivos de captura reales. Por el momento, trabajamos con los datos simulados limpios para obtener resultados en situaciones ideales, pero consideramos la posibilidad de añadir ruido similar al de captura en la simulaciones. El trabajo de Hernandez et al. (2017) analiza el comportamiento de los SPADs y crean un modelo dando indicaciones para emular los diferentes tipos de ruido que presentan. Utilizando este modelo, las evaluaciones que se obtengan del dataset deberán ser cercanas a las que se obtendrían utilizando datos capturados.

Los últimos métodos de reconstrucción en escenarios *NLOS* publicados durante la realización de este trabajo obtienen resultados sin precedentes (Liu et al., 2019b; Lindell et al., 2019; Xin et al., 2019). Se planea utilizar el dataset para probar estas nuevas técnicas de reconstrucción realizando un análisis más profundo de sus características. Es probable que este análisis nos lleve a reconocer los límites de nuestro trabajo, que idealmente presentaría una variedad de escenarios, parámetros de captura, y materiales mayor. No consideramos que el trabajo haya finalizado, sino que debe seguir en continua expansión, incluyendo nuevas escenas para seguir el avance del campo.

Todavía no han surgido métodos que realmente traten de explotar grandes cantidades de datos de imagen transitoria con aprendizaje automático. Una línea de trabajo futura tratará de explorar esta posibilidad, utilizando *deep learning* para mejorar la calidad o la eficiencia de las reconstrucciones, detectar superficies

automáticamente para evitar la elección manual de valores de umbral, o, quizás, desambiguar la trayectoria de los diferentes rebotes difusos de la luz en las escenas para recuperar regiones ocultas a las regiones ocultas o, *ver a través de la segunda esquina*. Este tipo de métodos requerirá de métricas de error más complejas que las que proponemos en este trabajo, donde se tengan en cuenta reconstrucciones de superficies no sólo frente a la escena visible, sino regiones ocluidas del objeto que hasta ahora no ha sido posible reconstruir.

Estos resultados se han presentado ya como póster en la conferencia de fotografía computacional *ICCP 2019* en Tokio, y se presentarán en la conferencia de gráficos por ordenador *SIGGRAPH 2019* en Los Ángeles (ver Anexo F). Finalmente, en verano de 2019 se hará una estancia en la Universidad de Wisconsin-Madison, continuando esta línea de investigación bajo la supervisión del profesor Andreas Velten centrándonos en el análisis y caracterización del comportamiento de las técnicas de reconstrucción más recientes frente a diferentes *BRDFs*.

Capítulo 7

Bibliografía

Adam, A., Dann, C., Yair, O., Mazor, S., and Nowozin, S. (2016). Bayesian time-of-flight for realtime shape, illumination and albedo.

Ament, M., Bergmann, C., and Weiskopf, D. (2014). Refractive radiative transfer equation. *ACM Transactions on Graphics*, 33(2):17:1–17:22.

Antipa, N., Oare, P., Bostan, E., Ng, R., and Waller, L. (2019). Video from stills: Lensless imaging with rolling shutter. In *IEEE International Conference on Computational Photography (ICCP)*. IEEE.

Arellano, V., Gutierrez, D., and Jarabo, A. (2017). Fast back-projection for non-line of sight reconstruction. 25(10):11574–11583.

Bitterli, B. (2016). Rendering resources. https://benedikt-bitterli.me/resources/.

Busck, J. and Heiselberg, H. (2004). Gated viewing and high-accuracy three-dimensional laser radar. 43(24):4705–4710.

Butler, D. J., Wulff, J., Stanley, G. B., and Black, M. J. (2012). A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *European Conf. on Computer Vision (ECCV)*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag.

Buttafava, M., Zeman, J., Tosi, A., Eliceiri, K., and Velten, A. (2015). Non-line-of-sight imaging using a time-gated single photon avalanche diode. 23(16).

Gariepy, G., Krstajić, N., Henderson, R., Li, C., Thomson, R. R., Buller, G. S., Heshmat, B., Raskar, R., Leach, J., and Faccio, D. (2015). Single-photon sensitive light-in-fight imaging. *Nature Communications*, 6.

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237.

Hamamatsu (2012). Guide to streak cameras.

Heide, F., Hullin, M. B., Gregson, J., and Heidrich, W. (2013). Low-budget transient imaging using photonic mixer devices. 32(4):45:1–45:10.

Heide, F., O'Toole, M., Zang, K., Lindell, D., Diamond, S., and Wetzstein, G. (2018). Non-line-of-sight imaging with partial occluders and surface normals. arXiv:1711.07134.

Heide, F., Xiao, L., Heidrich, W., and Hullin, M. B. (2014a). Diffuse mirrors: 3D reconstruction from diffuse indirect illumination using inexpensive time-of-flight sensors.

Heide, F., Xiao, L., Kolb, A., Hullin, M. B., and Heidrich, W. (2014b). Imaging in scattering media using correlation image sensors and sparse convolutional coding. 22(21).

Hernandez, Q., Gutierrez, D., and Jarabo, A. (2017). A computational model of a single-photon avalanche diode sensor for transient imaging. arXiv:1703.02635.

Heshmat, B., Gariepy, G., Leach, J., Raskar, R., and Faccio, D. (2016). Spad cameras for biomedical imaging: Promise and problems. In *2016 Conference on Lasers and Electro-Optics (CLEO)*, pages 1–2.

Hullin, M. B. (2014). Computational imaging of light in flight. In *SPIE/COS Photonics Asia*.

Iseringhausen, J. and Hullin, M. B. (2018). Non-line-of-sight reconstruction using efficient transient rendering. arXiv:1809.08044 [cs.GR].

Jarabo, A. (2012). *Femto-Photography: Visualizing Light in Motion*. M.Sc. Thesis, Universidad de Zaragoza.

Jarabo, A., Marco, J., Munoz, A., Buisan, R., Jarosz, W., and Gutierrez, D. (2014). A framework for transient rendering. 33(6):177:1–177:10.

Jarabo, A., Masia, B., Marco, J., and Gutierrez, D. (2017). Recent advances in transient imaging: A computer graphics and vision perspective. *Visual Informatics*, 1(1):65–79.

Kadambi, A., Whyte, R., Bhandari, A., Streeter, L., Barsi, C., Dorrington, A., and Raskar, R. (2013). Coded time of flight cameras: Sparse deconvolution to address multipath interference and recover time profiles. 32(6).

Kadambi, A., Zhao, H., Shi, B., and Raskar, R. (2016). Occluded imaging with time-of-flight sensors. *ACM Transactions on Graphics*, 35(2).

Kajiya, J. T. (1986). The rendering equation. 20(4):143–150.

Keller, A. (1997). Instant radiosity. Computer Graphics Proceedings, Annual Conference Series, pages 49–56.

Keller, M. and Kolb, A. (2009). Real-time simulation of time-of-flight sensors. *Simulation Modelling Practice and Theory*, 17(5).

Keller, M., Orthmann, J., Kolb, A., and Peters, V. (2007). A simulation framework for time-of-flight sensors. In *International Symposium on Signals, Circuits and Systems 2007*.

Kirmani, A., Hutchison, T., Davis, J., and Raskar, R. (2009). Looking around the corner using transient imaging.

Kirmani, A., Venkatraman, D., Shin, D., Colaço, A., Wong, F. N., Shapiro, J. H., and Goyal, V. K. (2014). First-photon imaging. *Science*, 343(6166).

Klein, J., Laurenzis, M., Michels, D. L., and Hullin, M. B. (2018). A quantitative platform for non-line-of-sight imaging problems. In *British Machine Vision Conference 2018 (BMVC)*, page 104.

Klein, J., Peters, C., Martín, J., Laurenzis, M., and Hullin, M. B. (2016). Tracking objects outside the line of sight using 2D intensity images. *Scientific Reports*, 6.

La Manna, M., Kine, F., Breitbach, E., Jackson, J., Sultan, T., and Velten, A. (2018). Error backprojection algorithms for non-line-of-sight imaging. *IEEE transactions on pattern analysis and machine intelligence*.

Lange, R., Seitz, P., Biber, A., and Lauxtermann, S. C. (2000). Demodulation pixels in CCD and CMOS technologies for time-of-flight ranging. In *Electronic Imaging*.

Laurenzis, M. and Velten, A. (2014). Nonline-of-sight laser gated viewing of scattered photons. 53(2).

Lin, J., Liu, Y., Hullin, M. B., and Dai, Q. (2014). Fourier analysis on transient imaging with a multifrequency time-of-flight camera.

Lindell, D. B., Wetzstein, G., and O'Toole, M. (2019). Wave-based non-line-of-sight imaging using fast f-k migration. *ACM Trans. Graph. (SIGGRAPH)*, 38(4):116.

Liu, X., Bauer, S., and Velten, A. (2019a). Analysis of feature visibility in non-line-of-sight measurements. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Liu, X., Guillen, I., Manna, M. L., Nam, J. H., Reza, S. A., Le, T. H., Gutierrez, D., Jarabo, A., and Velten, A. (2018). Virtual wave optics for non-line-of-sight imaging. arXiv:1810.07535.

Liu, X., Guillen, I., Manna, M. L., Nam, J. H., Reza, S. A., Le, T. H., Gutierrez, D., Jarabo, A., and Velten, A. (2019b). Non-line-of-sight imaging using phasor field virtual wave optics.

Marco, J. (2013). *Transient Light Transport in Participating Media*. Ph.D. Thesis, Universidad de Zaragoza.

Marco, J., Hernandez, Q., Muñoz, A., Dong, Y., Jarabo, A., Kim, M., Tong, X., and Gutierrez, D. (2017). DeepToF: Off-the-shelf real-time correction of multipath interference in time-of-flight imaging. 36(6).

Mitra, N., Ritschel, T., Kokkinos, I., Guerrero, P., Kim, V., Rematas, K., and Yumer, E. (2018). Deep learning for graphics. In *Eurographics 2018 Courses*, volume 39, pages 13–15. Eurographics Association.

Naik, N., Barsi, C., Velten, A., and Raskar, R. (2014). Estimating wide-angle, spatially varying reflectance using time-resolved inversion of backscattered light. 31(5).

Nathan Silberman, Derek Hoiem, P. K. and Fergus, R. (2012). Indoor segmentation and support inference from rgbd images. In *ECCV*.

O'Toole, M., Heide, F., Xiao, L., Hullin, M. B., Heidrich, W., and Kutulakos, K. N. (2014). Temporal frequency probing for 5D transient analysis of global light transport. 33(4):87:1–87:11.

O'Toole, M., Lindell, D. B., and Wetzstein, G. (2018). Confocal non-line-of-sight imaging based on the light-cone transform. 555(7696):338–341.

Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66.

O'Toole, M., Heide, F., Lindell, D., Zang, K., Diamond, S., and Wetzstein, G. (2017). Reconstructing Transient Images from Single-Photon Sensors. *Proc. IEEE CVPR*.

Pan, X., Arellano, V., and Jarabo, A. (2019). Transient instant radiosity for efficient time-resolved global illumination. In *Computers & Graphics (CEIG)*.

Pandharkar, R. (2011). *Hidden Object Doppler: Estimating Motion, Size and Material Properties of Moving Non-Line-of-Sight Objects in Cluttered Environments*. Ph.D. Thesis, Massachusetts Institute of Technology.

Pitts, P., Benedetti, A., Slaney, M., and Chou, P. (2014). Time of flight tracer. Technical report, Microsoft.

Serrano, A., Garces, E., Masia, B., and Gutierrez, D. (2017). Convolutional sparse coding for capturing high-speed video content. *Computer Graphics Forum*.

Smith, A., Skorupski, J., and Davis, J. (2008). Transient rendering. Technical Report UCSC-SOE-08-26, School of Engineering, University of California, Santa Cruz.

Su, S., Heide, F., Wetzstein, G., and Heidrich, W. (2018). Deep end-to-end time-of-flight imaging. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6383–6392.

Tsai, C.-Y., Sankaranarayanan, A. C., and Gkioulekas, I. (2019). Beyond volumetric albedo—a surface optimization framework for non-line-of-sight imaging. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2019)*.

Veach, E. (1997). *Robust Monte Carlo Methods for Light Transport Simulation*. Ph.D. Thesis, Stanford University, United States – California.

Velten, A., Willwacher, T., Gupta, O., Veeraraghavan, A., Bawendi, M. G., and Raskar, R. (2012). Recovering three-dimensional shape around a corner using ultrafast time-of-flight imaging. *Nature Communications*, (3).

Velten, A., Wu, D., Jarabo, A., Masia, B., Barsi, C., Joshi, C., Lawson, E., Bawendi, M. G., Gutierrez, D., and Raskar, R. (2013). Femto-photography: Capturing and visualizing the propagation of light. *ACM Transactions on Graphics (SIGGRAPH 2013)*, 32(4).

Voulodimos, A., Doulamis, N., Doulamis, A., and Protopapadakis, E. (2018). Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018.

Wald, I., Woop, S., Benthin, C., Johnson, G. S., and Ernst, M. (2014). Embree: A kernel framework for efficient CPU ray tracing. 33(4).

Wetzstein, G., Ihrke, I., Lanman, D., and Heidrich, W. (2011). Computational plenoptic imaging. *Computer Graphics Forum*, 30(8).

Wu, D., Velten, A., O'Toole, M., Masia, B., Agrawal, A., Dai, Q., and Raskar, R. (2014). Decomposing global light transport using time of flight imaging. 107.

Xin, S., Nousias, S., Kutulakos, K. N., Sankaranarayanan, A. C., Narasimhan, S. G., and Gkioulekas, I. (2019). A theory of fermat paths for non-line-of-sight shape reconstruction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2019)*.

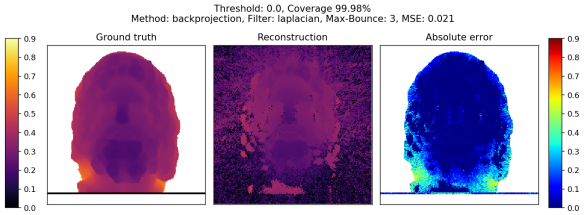
Lista de Figuras

Anexos

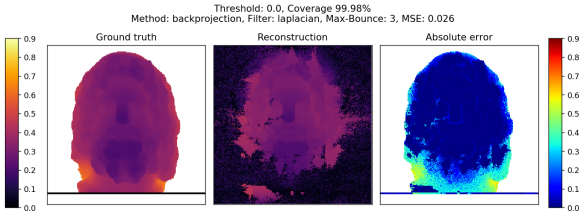
Anexos A

Desglose de escenas y resultados

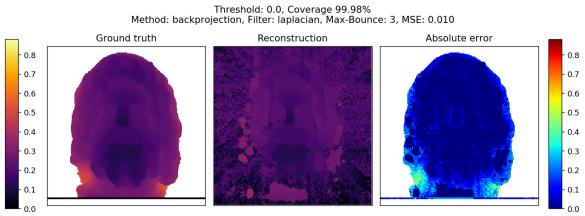
En este anexo mostramos más resultados que por su tamaño no incluimos en el documento principal.



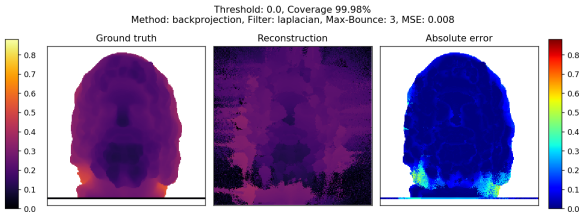
(a) Serapis 0,5m, 16×16 *exhaustivo*



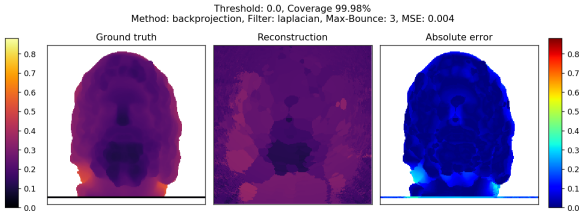
(b) Serapis 0,5m, *256x1 cap. única*



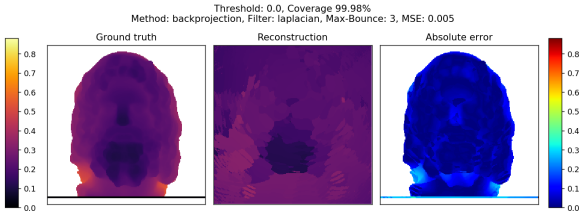
(c) Serapis 1,0m, 16×16 *exhaustivo*



(d) Serapis 1,0m, *256x1 cap. única*

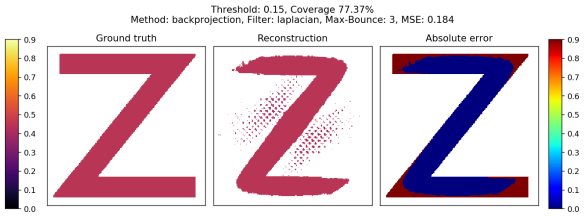


(e) Serapis 4,0m, 16×16 *exhaustivo*

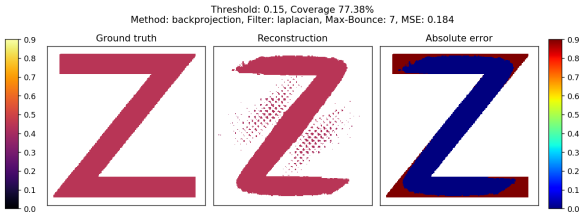


(f) Serapis 4,0m, *256x1 cap. única*

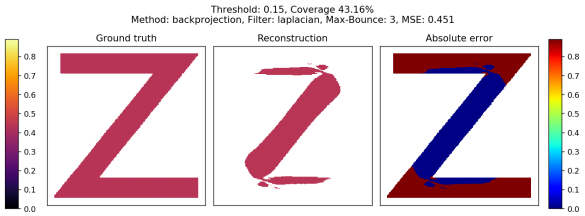
Figura A.1: Comparativa de reconstrucciones del busto de serapis a diferentes distancias y con patrones de captura 16×16 *y* 256×1 *utilizando filtered backprojection con valor de umbral* 0. *Vemos como la distancia tiene un gran efecto en la calidad de la reconstrucción, fallando en la reconstrucción a 4 metros. El patrón exhaustivo reconstruye mejor la base del busto a la distancia más corta debido a la auto-oclusión que sufre el patrón con un solo punto.*



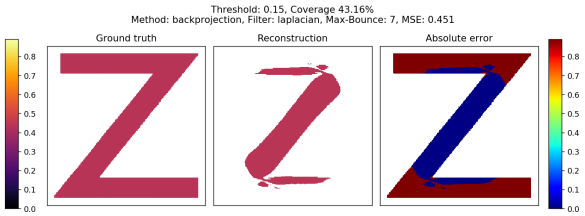
(a) Z 1,0m, 16×16, *3 rebotes*



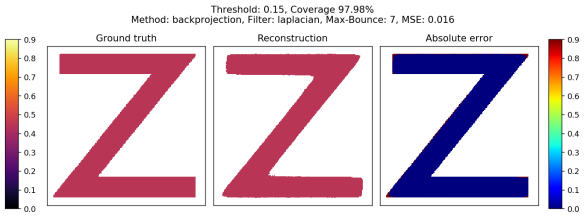
(b) Z 1,0m, 16×16, *7 rebotes*



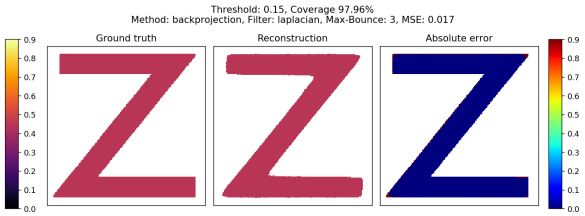
(c) Z 1,0m, 256×1, *3 rebotes*



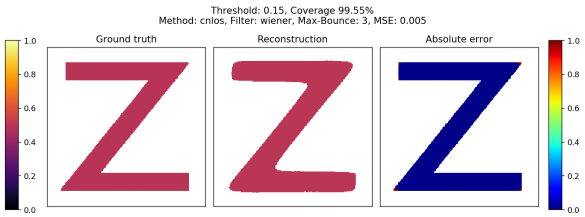
(d) Z 1,0m, 256×1, *7 rebotes*



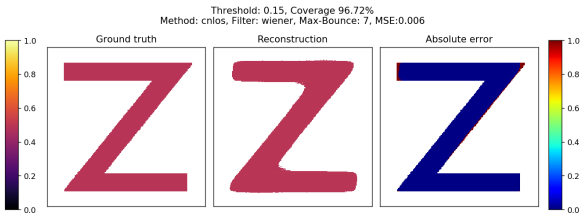
(e) Z 1,0m, 256×256, *3 rebotes*



(f) Z 1,0m, 256×256, *7 rebotes*

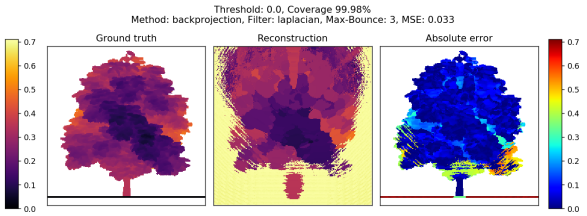


(g) Z 1,0m, 256×256, *3 rebotes*

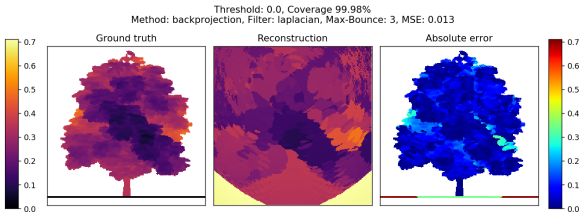


(h) Z 1,0m, 256×256, *7 rebotes*

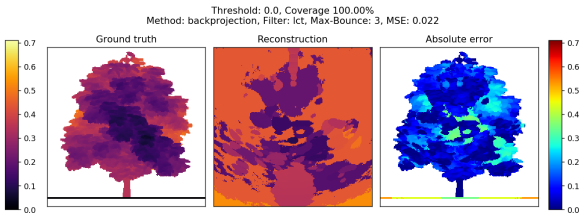
Figura A.2: Reconstrucciones de una escena sencilla con una letra plana flotando frente al muro visible. Vemos que la calidad de las reconstrucciones varía con el patrón de captura (en los casos con captura única se pierde gran parte de la reconstrucción al elegir un valor umbral demasiado elevado), pero el número de rebotes no afecta al resultado. La mayoría de métodos de reconstrucción asumen que la luz proviene únicamente de un tercer rebote, reforzando el uso de este tipo de escenas en trabajos previos.



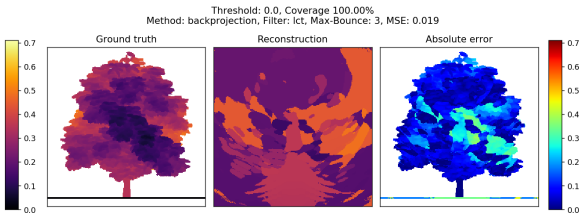
(a) *Chestnut* 4,0m, 16×16



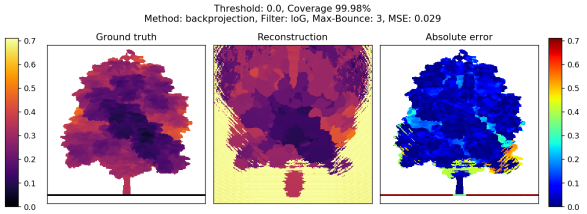
(b) *Chestnut* 4,0m, 256×1



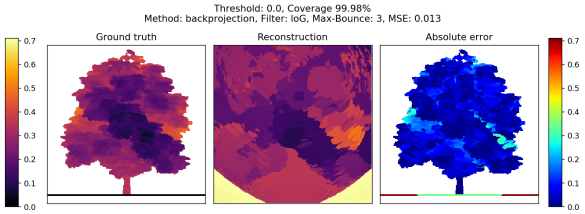
(c) *Chestnut* 4,0m, 16×16



(d) *Chestnut* 4,0m, 256×1

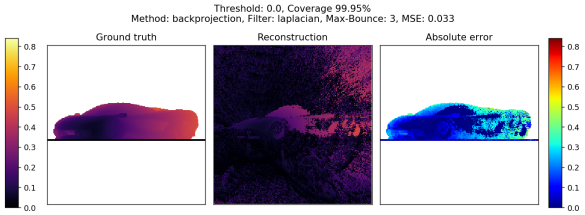


(e) *Chestnut* 4,0m, 16×16

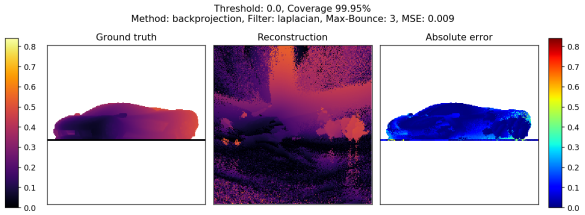


(f) *Chestnut* 4,0m, 256×1

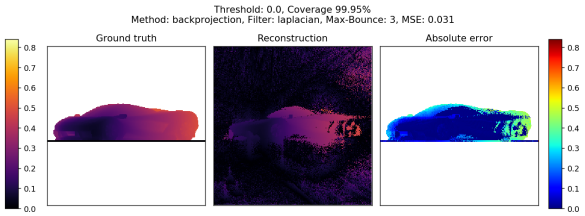
Figura A.3: Reconstrucciones del castaño dentro de una caja con FBP con dos tipos de patrones y filtrado con laplaciano (arriba), filtro LCT (medio) y laplaciano de Gauss (abajo).



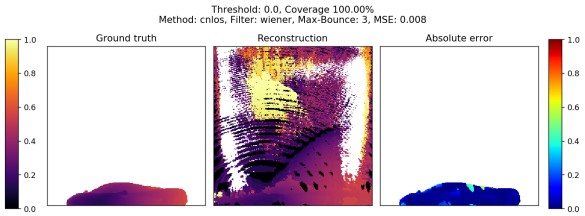
(a) Sports car, $\epsilon = 0$, 16×16



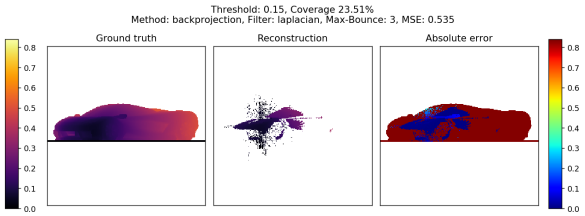
(b) Sports car, $\epsilon = 0$, 256×1



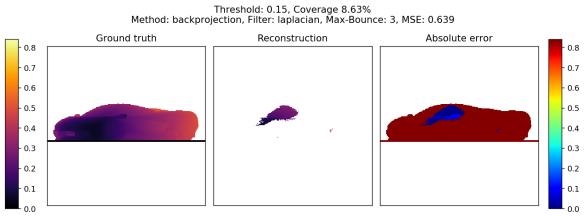
(c) Sports car, $\epsilon = 0$, 256×256



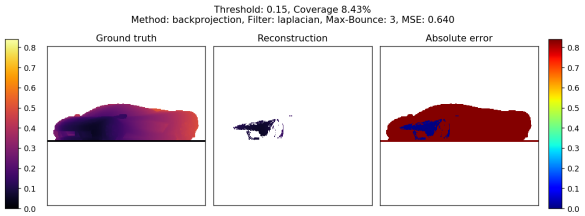
(d) Sports car, $\epsilon = 0$, 256×256



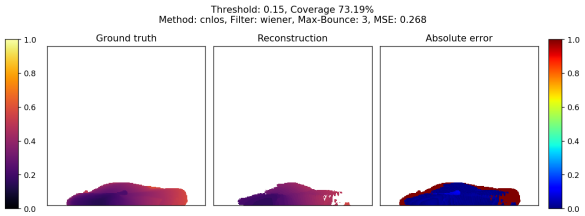
(e) Sports car, $\epsilon = 0{,}15$, 16×16



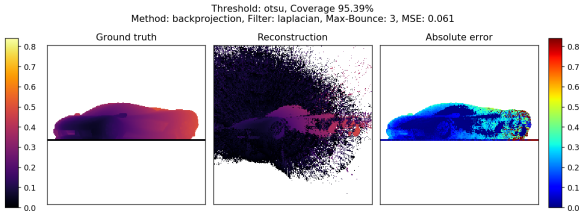
(f) Sports car, $\epsilon = 0{,}15$, 256×1



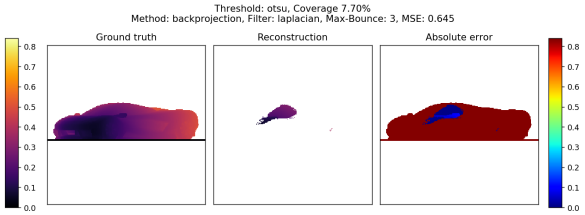
(g) Sports car, $\epsilon = 0{,}15$, 256×256



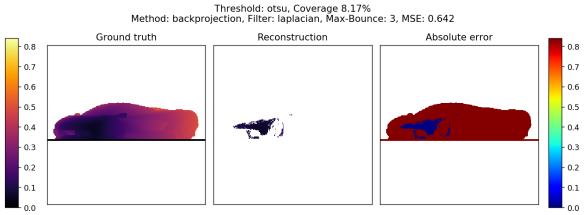
(h) Sports car, $\epsilon = 0{,}15$, 256×256



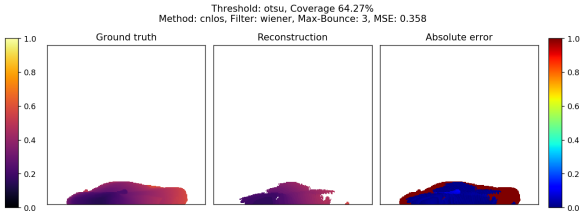
(i) Sports car, $\epsilon = Otsu$, 16×16



(j) Sports car, $\epsilon = Otsu$, 256×1



(k) Sports car, $\epsilon = Otsu$, 256×256



(l) Sports car, $\epsilon = Otsu$, 256×256

Figura A.4: Reconstrucciones del coche deportivo a $-0{,}5$m del plano visible. Observamos que FBP falla al reconstruir regiones que no están orientadas en dirección a la región capturada. LCT obtiene reconstrucciones significativamente más completas, pero está limitado a una región pequeña del coche que se encuentra sobre el suelo debido a los límites de Fourier-based backprojection.

Objeto oculto	Entorno	Distancia	Material
Z	flotando	0.5	Lambertian 1
Z	flotando	1.0	Lambertian 1
Z	en suelo	0.5	Lambertian 1
Z	en suelo	1.0	Lambertian 1
Z	en caja	0.5	Lambertian 1
Z	en caja	1.0	Lambertian 1
bunny	flotando	0.5	Lambertian 1
bunny	flotando	1.0	Lambertian 1
bunny	en suelo	0.5	Lambertian 1
bunny	en suelo	1.0	Lambertian 1
bunny	en caja	0.5	Lambertian 1
bunny	en caja	1.0	Lambertian 1
preoccl.	flotando	0.5	Lambertian 1
preoccl.	flotando	1.0	Lambertian 1
preoccl.	en suelo	0.5	Lambertian 1
preoccl.	en suelo	1.0	Lambertian 1
preoccl.	en caja	0.5	Lambertian 1
preoccl.	en caja	1.0	Lambertian 1
semiocc.	flotando	0.5	Lambertian 1
semiocc.	flotando	1.0	Lambertian 1
semiocc.	en suelo	0.5	Lambertian 1
semiocc.	en suelo	1.0	Lambertian 1
semiocc.	en caja	0.5	Lambertian 1
semiocc.	en caja	1.0	Lambertian 1
occluded	flotando	0.5	Lambertian 1
occluded	flotando	1.0	Lambertian 1
occluded	en suelo	0.5	Lambertian 1
occluded	en suelo	1.0	Lambertian 1
occluded	en caja	0.5	Lambertian 1
occluded	en caja	1.0	Lambertian 1
cavities	flotando	0.5	Lambertian 1
cavities	flotando	1.0	Lambertian 1
cavities	en suelo	0.5	Lambertian 1
cavities	en suelo	1.0	Lambertian 1
cavities	en caja	0.5	Lambertian 1
cavities	en caja	1.0	Lambertian 1
usaf	flotando	0.5	Lambertian 1
usaf	flotando	1.0	Lambertian 1
usaf	en suelo	0.5	Lambertian 1
usaf	en suelo	1.0	Lambertian 1
usaf	en caja	0.5	Lambertian 1
usaf	en caja	1.0	Lambertian 1
cornell	en suelo	0.5	Lambertian 1
cornell	en suelo	1.0	Lambertian 1
cornell	en suelo	4.0	Lambertian 1
cornell	en caja	1.0	Lambertian 1
cornell	en caja	4.0	Lambertian 1
serapis	en suelo	0.5	Lambertian 1
serapis	en suelo	1.0	Lambertian 1
serapis	en suelo	4.0	Lambertian 1
serapis	en caja	1.0	Lambertian 1
serapis	en caja	4.0	Lambertian 1
xyzdragon	en suelo	0.5	Lambertian 1
xyzdragon	en suelo	1.0	Lambertian 1
xyzdragon	en suelo	4.0	Lambertian 1
xyzdragon	en caja	1.0	Lambertian 1
xyzdragon	en caja	4.0	Lambertian 1
xyzdragon	en suelo	1.0	Ward 1
xyzdragon	en suelo	4.0	Ward 1
xyzdragon	en caja	1.0	Ward 1
xyzdragon	en caja	4.0	Ward 1
sports car	en suelo	0.5	Lambertian 1
sports car	en suelo	1.0	Lambertian 1
sports car	en suelo	4.0	Lambertian 1
sports car	en caja	1.0	Lambertian 1
sports car	en caja	4.0	Lambertian 1
sports car	en suelo	1.0	Ward 1
sports car	en suelo	4.0	Ward 1
sports car	en caja	1.0	Ward 1
sports car	en caja	4.0	Ward 1
hairball	en suelo	0.5	Lambertian 1
hairball	en suelo	1.0	Lambertian 1
hairball	en suelo	4.0	Lambertian 1
hairball	en caja	1.0	Lambertian 1
hairball	en caja	4.0	Lambertian 1
chestnut tree	en suelo	0.5	Lambertian 1
chestnut tree	en suelo	1.0	Lambertian 1
chestnut tree	en suelo	4.0	Lambertian 1
chestnut tree	en caja	1.0	Lambertian 1
chestnut tree	en caja	4.0	Lambertian 1
lucy	en suelo	0.5	Lambertian 1
lucy	en suelo	1.0	Lambertian 1
lucy	en suelo	4.0	Lambertian 1
lucy	en caja	0.5	Lambertian 1
lucy	en caja	1.0	Lambertian 1
lucy	en caja	4.0	Lambertian 1
chinese dragon	en suelo	1.0	Lambertian 1
chinese dragon	en suelo	4.0	Lambertian 1
chinese dragon	en caja	1.0	Lambertian 1
chinese dragon	en caja	4.0	Lambertian 1
chinese dragon	en suelo	1.0	Ward 1
chinese dragon	en suelo	4.0	Ward 1
chinese dragon	en caja	1.0	Ward 1
chinese dragon	en caja	4.0	Ward 1
stanf bunnny	en suelo	1.0	Ward 1
stanf bunnny	en suelo	4.0	Ward 1
stanf bunnny	en caja	1.0	Ward 1
stanf bunnny	en caja	4.0	Ward 1
indon. statue	en suelo	0.5	Lambertian 1
indon. statue	en suelo	1.0	Lambertian 1
indon. statue	en suelo	4.0	Lambertian 1
indon. statue	en caja	1.0	Lambertian 1
indon. statue	en caja	4.0	Lambertian 1
MHDesktop1	compleja	—	Lambertian 1
MHDesktop2	compleja	—	Lambertian 1
MHGryphon	compleja	—	Lambertian 1
MHLivingRoom	compleja	—	Lambertian 1
MHDesktop1	compleja	—	Lambertian 1
victorian	compleja	—	Lambertian 1
crowdedstaircase	compleja	—	Lambertian 1
citycars	compleja	—	Lambertian 1
citycars	compleja	—	Varied

Figura A.5: Resumen de escenas incluidas en el dataset. Cada una de estas escenas se simula con tres patrones de captura diferentes.

Anexos B

Simulación

Implementamos extensiones sobre el motor de renderizado de Jarabo et al. (2014), creando una nueva interfaz diseñada específicamente para simular escenas *NLOS* de forma sencilla y eficiente, sin interferir con la interfaz para render transitorio ya existente.

B.1. Parámetros de entrada

Definimos los patrones de captura que hemos visto anteriormente como elementos básicos en el motor de render. Describimos tanto la captura como la iluminación de las escenas con las posiciones de cada dispositivo y los puntos hacia los que mira su campo visual. Especificamos las dimensiones y número de puntos de superficies rectangulares centradas en la intersección del punto de mira con la primera superficie de la escena. Asumiendo que la superficie es plana y conociendo su dirección normal, propagamos las posiciones que deberán tener los puntos que se van a iluminar o capturar.

B.2. Formato de salida

Típicamente, la salida del proceso de render es una imágen. Sin embargo, en este caso, el render mínimo produce una imagen unidimensional representando el transporte de luz en un punto a lo largo del tiempo (ver Figura 4.2), como si se tratase de un vídeo de ese punto. Esto no es práctico a la hora de simular las escenas, ya que para cada una de ellas produciríamos decenas de miles de ficheros de imagen. La nueva interfaz produce un único fichero *HDF5* agregando la información de todos los puntos en un tensor N-dimensional (dependiendo del patrón de captura). Esto supone una mejora en el rendimiento, evitando las escrituras en disco anteriores. Además, los ficheros *HDF5* permiten el uso de compresión en lectura y escritura, ahorrando espacio en disco, y, al ser un formato estandarizado, se pueden utilizar en todo tipo de sistemas con facilidad.

B.3. Generación y simulación automática

Crear estas escenas manualmente no es viable, y sería una fuente de errores adicional, por lo que nos interesa automatizar el proceso. Aprovechamos la API del software de modelado 3D *blender* en *python* para generar las escenas de forma rápida a partir de un descriptor de escena sencillo que indica el objeto oculto, su posición, tamaño y rotación y el entorno en el que se coloca (flotando, en el suelo o en una caja). Una vez generada la escena, la simulamos con los parámetros indicados en el mismo fichero, ya que no es posible describir la iluminación y captura de la escena en ficheros estándar de escenarios 3D. Este proceso lo consideramos dentro de un pipeline de simulación, simplificando. Incluso teniendo en cuenta las optimizaciones anteriores, el tiempo de cómputo total del dataset ha sido de unas 6000 horas de máquina. Distribuimos el cómputo en máquinas de altas prestaciones del laboratorio a lo largo de varios meses de trabajo.

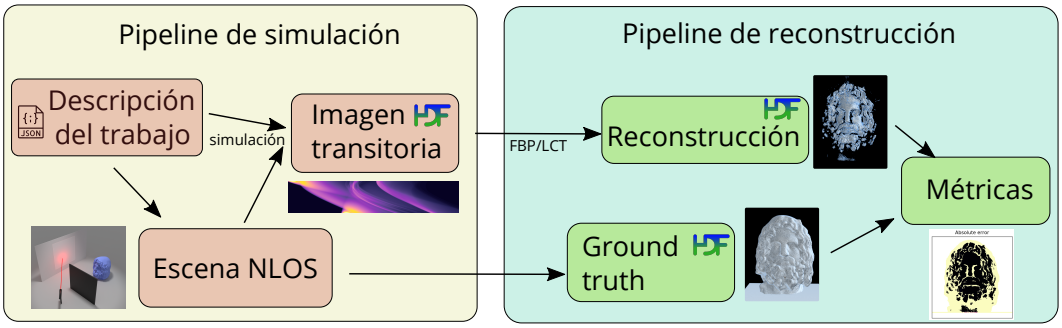


Figura B.1: Esquema del pipeline de generación, simulación y reconstrucción de escenas.

Anexos C

Backprojection: implementación en *CUDA*

C.1. Algoritmo de *Backprojection*

El Algoritmo 1 representa el algoritmo básico de *backprojection* descrito en el trabajo de Velten et al. (2012). Las operaciones realizadas en el algoritmo son muy sencillas: tan solo se calculan distancias entre puntos y se suman valores de la imagen transitoria sobre los vóxeles de un volumen. En definitiva, a cada vóxel se le asigna la suma de radiancias que le corresponde por la distancia a la que se encuentra de todos los puntos capturados.

En otra forma de plantear el problema, cada instante discreto capturado, proyecta sobre el volumen un elipsoide con su valor, como vimos en la Figura 5.1. Este elipsoide corresponde con todas las posiciones del volumen que podrían haber dado lugar a ese valor en la captura de existir en la región oculta. En general, esto es cierto para uno solo de esos vóxeles si no existen interferencias por caminos múltiples, es decir, la luz no ha rebotado más que una vez en la escena oculta. Sin embargo, el algoritmo es robusto frente a este tipo de errores en la asunción debido a que los valores erróneos no tienen la frecuencia e intensidad suficientes para superar los correctos. Además, los rebotes de alto orden recorren, por fuerza, caminos más largos por la escena, perdiendo intensidad y generando elipsoides detrás de las superficies reconstruidas correctamente, reduciendo su efecto a halos con ruido de valores bajos con poco impacto visual en la reconstrucción final.

Algoritmo: Backprojection
Data: images: transient images of each pair,
pairs: capture and lighting positions,
camera: camera position,
laser: laser position,
Δt: per pixel exposure distance (in meters)
Result: volume
initialize volume with zeros;
foreach *voxel in volume* **do**
  foreach *pair in pairs* **do**
    $distance \leftarrow dist(laser, pair.laser) + dist(pair.laser, voxel) + dist(voxel, pair.capture) + dist(pairs.capture, camera)$;
    $volume[voxel] \leftarrow volume[voxel] + images[pair][\text{distance}/\Delta t]$;
  end
end

Algoritmo 1: *Backprojection* de Velten et al. (2012). Aunque su implementación es sencilla, el coste computacional es elevado, con el bucle externo recorriendo dimensiones x, y, z , y el interno los pares capturados, típicamente, en x, y ($\mathcal{O}(n^5)$).

C.2. Filtrado

La calidad de la reconstrucción de una escena *NLOS* utilizando *backprojection* depende en gran medida del filtro que se utiliza para obtenerla. Sin filtrado, los resultados son demasiado borrosos debido a que el método amplifica los componentes de baja frecuencia más que los de alta. Por tanto, una heurística común para mejorar los resultados es utilizar filtros de paso alto, como el laplaciano o laplaciano de Gauss, para contrarrestar este efecto enfatizando los componentes de alta frecuencia. El coste de aplicar estos filtros es muy bajo, tan solo una convolución 3D sobre el mapa de probabilidades, permitiendo probar diferentes filtros sobre un mismo volumen para obtener mejores resultados. Como resultado del filtro, las superficies ocultas descritas en el mapa de probabilidades quedan significativamente más definidas.

Sin embargo, filtrar los volúmenes no elimina todas las regiones erróneas, cuyos valores persisten y que pueden llegar a ser comparables a los reconstruidos correctamente. Aunque idealmente el volumen sería binario, diferenciando los vóxeles en los que hay una superficie de los que están vacíos, sus valores dan una aproximación de la confianza que se da a que un vóxel exista en la escena. Por tanto, es necesario establecer un umbral a partir del cual se considerarán los valores del mapa de probabilidades como existentes, eliminando ruido al mismo tiempo. En escenas grandes, esto causará problemas por la intensidad lumínica reducida de las superficies más

alejadas con respecto a las cercanas, causando falsos positivos o negativos según cómo se ajuste el valor del umbral. El ajuste de este valor es un proceso manual, algo en lo que se debe mejorar el algoritmo si se busca una solución integral de utilidad en el mundo real. Pruebas preliminares con métodos para obtener umbrales de forma automática como Otsu (1979) dan resultados mixtos según la escena.

C.3. Implementación en *GPU*

Como parte del trabajo, realizamos una implementación del algoritmo original de Velten et al. (2012) en *CUDA*. Si la reconstrucción corresponde con la escena simulada sabemos que el pipeline es correcto y los datos no se han corrompido en algún paso de la simulación y reconstrucción. Nos ha interesado, por tanto, que sea un proceso rápido, por lo que decidimos realizar una implementación que aproveche el paralelizado masivo de los aceleradores gráficos para computarlo, al mismo tiempo que evitamos errores adicionales causados por la propia reconstrucción, como es el caso de Arellano et al. (2017).

Identificamos como tarea mínima en el algoritmo sumar todos los valores de la imagen transitoria que corresponden a un vóxel del mapa de probabilidades. Programamos la ejecución de cada tarea en un bloque de hilos, repartiendo los pares que conforman la imagen transitoria entre los hilos equitativamente (bucle interno del Algoritmo 1). En tiempo de ejecución, la API distribuye los vóxeles de un volumen de dimensiones $X \times Y \times Z$ entre los multiprocesadores de la *GPU*, donde se ejecutan en paralelo según dicta su planificador.

El principal obstáculo para implementar el algoritmo en *GPU* es el *watchdog* temporizado que expulsa procesos largos (5 segundos) de *GPU* si se utiliza simultáneamente su función gráfica. La solución es subdividir la reconstrucción en bloques pequeños que puedan calcularse en menos tiempo y no ser expulsados. Para evitar que esto suponga un impacto en el tiempo de ejecución, todo el cálculo se sigue realizando en *GPU*, estableciendo una ventana sobre el volúmen de reconstrucción completo en la que se trabaja en cada llamada y que avanza en la ejecución de cada bloque. Trabajamos con ventanas de tamaño $16 \times 16 \times 16$, calculando 4096 vóxeles en cada llamada al *kernel*, con el trabajo de cada vóxel repartido entre 256 hilos.

Reconstruir un volumen de 256^3 vóxeles usando una implementación paralelizada en *CPU* tarda aproximadamente 40 minutos, mientras que nuestra implementación tarda aproximadamente 7 minutos[1], con un *speed-up* de 5,77. Al contrario que el trabajo

[1] Ambos resultados en una máquina con un intel i7-7700 en 8 hilos de ejecución y una NVIDIA GTX 980.

de Arellano et al. (2017), nuestra implementación no presenta errores medibles más allá de los que cabe esperar al realizar operarciones en punto flotante con valores muy pequeños en paralelo. Además, en comparación con métodos basados en LCT O'Toole et al. (2018), el coste en memoria se puede amortizar fácilmente dividiendo el trabajo en bloques más pequeños, de modo que es posible escalar el sistema a grandes cantidades de datos sin necesidad de utilizar hardware de *workstation* con mayor cantidad de memoria gráfica.

C.4. Posibles mejoras

Como es común en computación de altas prestaciones, este algoritmo presenta cuellos de botella dados por las latencias de acceso a memoria y tasas de fallo en cache. Para óbtener mejoras de rendimiento, hay varios puntos en los que podríamos trabajar:

1. Reducir accesos y espacio en memoria modificando la imagen transitoria para evitar calcular distancias muro-laser y muro-camara.

2. Alinear los accesos a la imagen transitoria, almacenándola por columnas (dimension temporal).

3. Aprovechar correlaciones en los accesos a memoria desde cada vóxel para hacer mejor uso de las caches.

4. Utilizar estructuras de datos más compactas o que sitúen próximos los elementos que se acceden secuencialmente con mayor probabilidad.

5. Cambiar el orden en el que se computan los vóxeles, agrupandolos por distancia al muro para incrementar los aciertos en cache.

Anexos D

Filtro *LCT* para *Backprojection*

D.0.1. Modelo de formación de imagen *LCT*

El modelo de formación de imagen estándar en *NLOS imaging* requiere de una matriz grande y de alta dimensión, compleja de almacenar e invertir, por lo que se recurre a aproximaciones como *backprojection*. La *LCT*, descrita en el trabajo de O'Toole et al. (2018), evita estos problemas al realizar una transformación no-uniforme de las medidas confocales y expresa el modelo de formación de imagen como una convolución tridimensional.

$$\underbrace{v^{\frac{3}{2}}\ \tau\left(x', y', \frac{2}{c}\sqrt{v}\right)}_{\mathcal{R}_t\{\tau\}(x',y',v)} = \iiint_{\Omega} \underbrace{\frac{1}{2\sqrt{u}}\ \rho\left(x, y, \sqrt{u}\right)}_{\mathcal{R}_z\{\rho\}(x,y,u)}\ \underbrace{\delta\left(\left(x'-x\right)^2 + \left(y'-y\right)^2 + u - v\right)}_{h(x'-x,y'-y,v-u)}\ dx\, dy\, du.$$

o, simplificando:

$$\mathcal{R}_t\left\{\tau\right\} = h * \mathcal{R}_z\left\{\rho\right\}, \tag{D.1}$$

donde h es un kernel de desenfoque conocido, $\mathcal{R}_t$ es un operador de transformación aplicado sobre los datos τ, y $\mathcal{R}_z$ es un operador de transformación aplicado sobre el volumen desconocido ρ.

Expresar el modelo como una convolución tiene múltiples ventajas, incluyendo una reducción en el tiempo de cómputo necesario para evaluar la expresión.

D.0.2. Análisis de Fourier del kernel de desenfoque

Con la transformada de Fourier podemos realizar convoluciones tridimensionales entre dos funciones de forma eficiente. A continuación, derivamos analíticamente la transformada de Fourier del kernel h.

Primero, atendemos a la identidad[1]:

$$\int_{-\infty}^{\infty} e^{-ax^2-bx}\ dx = \sqrt{\frac{\pi}{a}} e^{\frac{b^2}{4a}} \quad \text{where} \quad a > 0 \tag{D.2}$$

[1]Obtenida de `https://en.wikipedia.org/wiki/List_of_integrals_of_exponential_functions`

La transformada de Fourier del kernel h viene dada por la expresión:

$$\begin{aligned} H(f_x, f_y, f_v) &= \mathcal{F}\{h\}(f_x, f_y, f_v) \\ &= \iiint \delta(x^2 + y^2 - v) e^{-2\pi i(xf_x + yf_y + vf_v)}\, dx\, dy\, dv \\ &= \iint e^{-2\pi i(xf_x + yf_y + (x^2+y^2)f_v)}\, dx\, dy \\ &= \int e^{-2\pi i(xf_x + x^2 f_v)}\, dx \int e^{-2\pi i(yf_y + y^2 f_v)}\, dy \end{aligned} \tag{D.3}$$

Combinándola con la identidad de la Ecuación (D.2), esto produce:

$$\begin{aligned} &= \left(\sqrt{\frac{\pi}{2\pi i f_v}} \exp\left(\frac{(2\pi i f_x)^2}{4(2\pi i f_v)}\right)\right) \cdot \\ &\qquad \left(\sqrt{\frac{\pi}{2\pi i f_v}} \exp\left(\frac{(2\pi i f_y)^2}{4(2\pi i f_v)}\right)\right) \end{aligned} \tag{D.4}$$

$$= -\frac{i}{2f_v} \exp\left(\pi i \left(\frac{f_x^2 + f_y^2}{2f_v}\right)\right) \tag{D.5}$$

Nótese que esta expresión solo es válida para $f_v > 0$, como resultado de la Ecuación (D.2).

D.0.3. Filtro de *Wiener* sobre *backprojection*

El algoritmo de *LCT* estándar resuelve *NLOS imaging* interpretándolo como una deconvolución tridimensional. Una forma de hacerlo consiste en computar un filtro de *Wiener* del kernel h donde la señal y el espectro del ruido se asumen conocidos.

El filtro de *Wiener* g se puede describir en el dominio de frecuencias como:

$$G = \mathcal{F}\{g\} = \frac{H^*}{|H|^2 + \frac{1}{\alpha}} \tag{D.6}$$

donde α representa el ratio señal/ruido espectral (SNR). Para valores grandes de α, la calidad de la señal mejora, y el filtro de *Wiener* converge al filtro inverso, $G = 1/H$. Para valores pequeños de α, la calidad de la señal es pobre y el filtro resultante es equivalente a realizar *backprojection*, $G = H^*$.

Combinando la Ecuación (D.5) y el filtro de *Wiener* para diferente SNR obtenemos:

$$\frac{H^*}{|H|^2 + \frac{1}{\alpha}} = i\frac{2f_v\alpha}{4f_v^2 + \alpha} \exp\left(-\pi i \left(\frac{f_x^2 + f_y^2}{2f_v}\right)\right) \tag{D.7}$$

Nótese que el parámetro complejo representado por la exponencial se mantiene igual para los diferentes filtros. El único componente que varía es la amplitud en la frecuencia, y esta es una función de f_v.

Esto sugiere que, dado un resultado de *backprojection* proveniente de medidas confocales sin filtrar, podemos seguir tres pasos básicos para aplicar el filtro inverso

que describe la LCT: 1) realiza la operación de estirado no-uniforme del volumen, 2), en el dominio de frecuencias, atenuar los coeficientes de Fourier por un factor de $4f_v^2$, 3) deshacer el estirado no-uniforme del volumen para obtener el resultado final.

Anexos E

Web

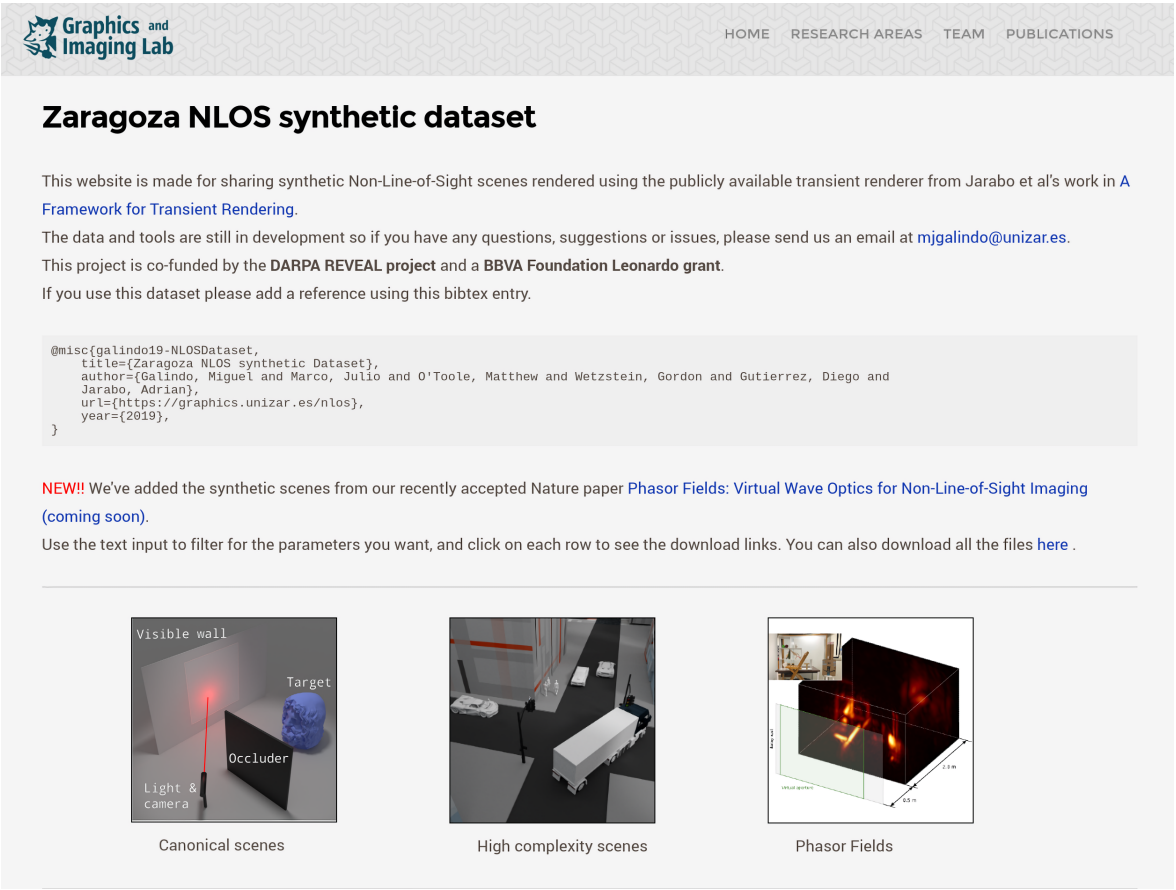


Figura E.1: Web donde publicamos el dataset `graphics.unizar.es/nlos`.

Publicamos los datos en la web de la Figura E.1. Cuenta con un buscador de escenas para hacer más sencillo encontrar aquellas que interesen al investigador, ya que descargar el dataset completo (500GB) no es práctico en general, explicaciones de los formatos que utilizamos en el dataset, y el código necesario para utilizarlo con facilidad en *Python, MATLAB* y *C++*.

Anexos F

Póster

A continuación, incluímos el póster presentado en *ICCP 2019* en Tokio y que presentaremos en *SIGGRAPH 2019* en Los Ángeles.

A Dataset for Benchmarking Time-Resolved Non-Line-of-Sight Imaging

Miguel Galindo[1] Julio Marco[1] Matthew O'Toole[2] Gordon Wetzstein[3] Diego Gutierrez[1] Adrian Jarabo[1]

Universidad de Zaragoza, I3A[1] Carnegie Mellon University[2] Stanford University[3]

Transient imaging [1] has made it possible to look around corners by exploiting information from indirect light bounces. While most previous works have only demonstrated this technique in limited and controlled scenarios [2-6], recent works have proven that this technology can be applied to image very complex occluded scenes [7,8]. We present a public dataset of synthetic time-resolved Non-Line-of-Sight (NLOS) scenes to validate new research, benchmark reconstruction methods, and serve as training data for imaging methods based on machine learning. Our dataset is an order of magnitude larger than any other available data, including over 300 scenes. These scenes include an increasing level of complexity, from simple isolated objects with varying complexity in geometry and reflectance, to scenes with significant multibounce, and to scenes simulating real-world indoor and outdoor scenarios. With this dataset, we hope to boost research on NLOS imaging and to help bringing it closer to real-world applications.

Time-resolved NLOS imaging

Time-resolved NLOS Imaging exploits indirect diffuse reflections on a wall to estimate the geometry of hidden objects.

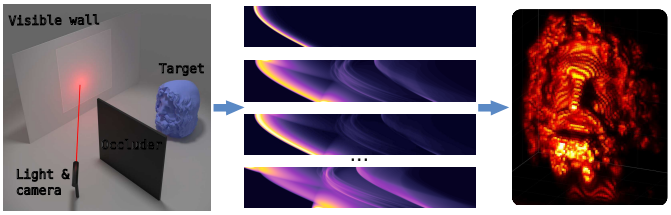
The canonical NLOS setup contains an isolated hidden object, and a diffuse wall in the line of sight of a capture device as in the figure on the right.

Capturing many samples in the wall results in volumes representing radiance in time with valuable information on the hidden object.

Since the seminal work by Velten et al. [2], there have been many improvements in NLOS imaging [2,3]. However, the available data for validation remains very limited, which makes it difficult to compare results.

Having an extensive public dataset for benchmarking will be a great tool to compare works, and even to attempt data-driven reconstruction methods as is common in other computer vision tasks.

Overview of the NLOS pipeline

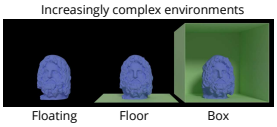


NLOS setup | Time-resolved slices from the captured volume | Reconstruction

Canonical scenes

- Hidden objects with varied complexity
- Placed at varying distances from the diffuse wall
- Increasingly complex environments that contribute multiple path interference
- Different BRDF models: Lambertian and Ward

Increasingly complex environments



Each scene is simulated using three capture patterns, giving us a total of 276 reconstructible scenes for benchmarking, validation, and data-driven research.

Obtaining the simulations

To render time-resolved NLOS scenes efficiently, we rely on the latest advances on transient rendering [9].

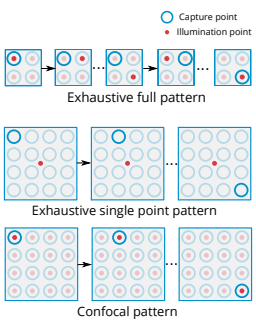
We use the following three illumination and capture patterns, making the dataset valid for benchmarking all current reconstruction methods.

All simulations contain 65,536 time-resolved captures, regardless of the pattern used.

Exhaustive: more spatial information, complex capture.

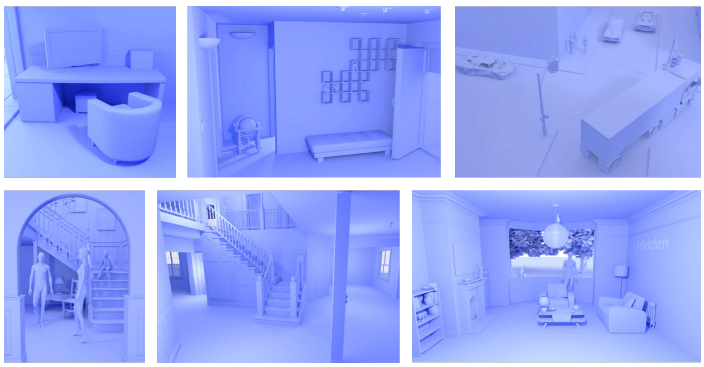
Single: less spatial information, specially for occlusions, simple to capture.

Confocal: spatial information, noisy data in the real world. Some methods only work with this kind of captures.



High-Complexity scenes

Real world scenes are much more complex than any canonical setup.
Testing NLOS reconstructions on these complex scenarios is necessary to detect their strengths and weaknesses, and to keep on improving them.
We include high-complexity, realistic scenes, considering both indoor and outdoor spaces.



Sample of the high-complexity scenarios in our dataset

Recently, the work in [7,8] proved the use of NLOS technology to reconstruct scenarios of this level of complexity, further highlighting the need for more examples of scenarios exhibiting high-complexity features.

Benchmarking

To compare reconstructions, we propose using error metrics based on distance to the relay wall. This avoids the issues with full 3D volume comparisons and additional errors caused by triangulation algorithms.
We simulate each light bounce separately, allowing researchers to study the the adverse effects of higher-order bounces, or even to exploit them.

Reconstructed depth using the method described in [3].

Ground truth | 3rd bounce | 5th bounce | 6th bounce | 7th bounce

Dataset available at: ***graphics.unizar.es/nlos***

Contact: ***mjgalindo@unizar.es***

References

[1] A. Jarabo et al. Recent Advances in Transient Imaging: A Computer Graphics and Vision Perspective. Visual Informatics. 2017.
[2] A. Velten et al. Recovering three-dimensional shape around a corner using ultrafast time-of-flight imaging. Nature communications. 2012.
[3] M. OToole et al. Confocal Non-Line- of-Sight Imaging Based on the Light-Cone Transform. Nature. 2018.
[4] V. Arellano et al. Fast Back-Projection for Non-Line of Sight Reconstruction. Optics Express. 2017.
[5] F. Heide et al. Diffuse mirrors: 3D reconstruction from diffuse indirect illumination using inexpensive time-of-flight sensors. CVPR. 2014.
[6] J. Klein et al. A Quantitative Platform for Non-Line-of-Sight Imaging Problems. BMVC. 2018
[7] X. Liu et al. Phasor Fields: Virtual Wave Optics for Non-Line-of-Sight Imaging. Nature. 2019.
[8] D. Lindell et al. Wave-Based Non-Line-of-Sight Imaging using Fast f-k Migration. ACM Transactions on Graphics. 2019.
[9] A. Jarabo et al. A Framework for Transient Rendering. ACM Transactions on Graphics. 2014.

Graphics and Imaging Lab

This project has been funded by DARPA (project REVEAL), the European Research Council (ERC) under the EU's Horizon 2020 research and innovation programme (project CHAMELEON, grant No 682080), the Spanish Ministry of Economy and Competitiveness (project TIN2016-78753-P), and the BBVA Foundation through a Leonardo grant.