PEDECIBA INFORMÁTICA

Instituto de Computación, Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay

TESIS DE MAESTRÍA EN INFORMÁTICA

Resolución de Problemas Inversos de Iluminación Considerando Datos Fotométricos

Rodrigo Leira
rodrigoleira10@hotmail.com.uy

6 de Abril de 2018

Dr. Eduardo Fernández	Director académico y tutor de tesis
Dr. Gonzalo Besuievsky	Co-tutor de tesis
Dr. Francisco José Serón Arbeloa	Revisor
Dr. Pedro Piñeyro	
Dr. Marcos Viera	

Publicaciones relacionadas

Esta tesis comprende las siguientes publicaciones y expande algunas de estas:

- Rodrigo Leira y Eduardo Fernández y Gonzalo Besuievsky *Calculation of Optimal Luminaires for Architectural Design.* Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 1: GRAPP, (VISIGRAPP 2017), Oporto, Portugal, Páginas 203-211.
- Ignacio Decia and Rodrigo Leira and Martín Pedemonte and Eduardo Fernández and Pablo Ezzatti *A VNS with Parallel Evaluation of Solutions for the Inverse Lighting Problem.* Applications of Evolutionary Computation - 20th European Conference, EvoApplications 2017, Amsterdam, The Netherlands, April 19-21, 2017, Proceedings, Part I, Páginas 741-756.

Resumen

En el diseño arquitectónico, a los objetivos estéticos y de confort visual se han agregado aquellos relacionados con la eficiencia energética y el cumplimiento de estándares de calidad en la iluminación. Este cambio de paradigmas hace relevante el estudio y desarrollo de técnicas computacionales que ayuden en la búsqueda de buenas configuraciones de luminarias. Considerar estos nuevos objetivos durante el proceso de diseño sin las herramientas adecuadas resulta ineficiente, porque está basado principalmente en el proceso de prueba y error y en la experiencia del diseñador. Las herramientas de CAD existentes generan resultados a partir de configuraciones proporcionadas por el diseñador, sin brindar nuevas soluciones. Debido a esto surge la necesidad de otro tipo de herramienta que se base en las intenciones del diseñador para generar configuraciones de luminarias adecuadas.

Esta tesis se centra en el desarrollo de nuevas y eficientes heurísticas que tengan en cuenta las propiedades fotométricas de luminarias reales, así como su ubicación y orientación. Las intenciones de iluminación (LI) del diseñador se definen como objetivos y restricciones a satisfacer, y son tratadas como problemas de optimización denominados problemas inversos de iluminación (ILP). Las configuraciones obtenidas son el punto de partida para el diseñador, dado que podrá modificarlas para contemplar otros aspectos más difíciles de modelar matemáticamente.

Desde el punto de vista de los métodos de iluminación global, las técnicas propuestas utilizan la ecuación de radiosidad. Se simula la emisión de la luz de las luminarias y a través de la ecuación de radiosidad se calcula cuánta luz llega a cada parche de la escena.

Se realizaron experimentos centrados en la mejora de la iluminación en el edificio Palacio de los Tribunales (Poder Judicial), donde se comparan los resultados obtenidos con aquellos propuestos por diseñadores y se muestra que las heurísticas desarrolladas tienen el potencial de facilitar el proceso de diseño de iluminación. Un análisis general muestra que las técnicas implementadas son capaces de obtener buenas soluciones en el conjunto de problemas estudiado, y de obtener tiempos de ejecución adecuados para este tipo de problemas. Por tanto, estas técnicas podrían ser utilizadas como herramientas de apoyo al proceso de diseño arquitectónico.

Palabras Clave: optimización, problema inverso de iluminación, intenciones de iluminación, radiosidad, fotometría, archivos fotométricos, luminarias.

Índice general

Índice de figuras

Índice de cuadros

Agradecimientos

Quiero agradecer enormemente a mis tutores por toda la paciencia que me tuvieron en este largo camino que ha sido el desarrollo de esta tesis. A Gonzalo por su buena onda y por siempre estar en las charlas de como encaminar el trabajo, siempre dando consejos. Y a Eduardo, por todo, por estar siempre cuando tenía una pregunta, siempre con paciencia y mucha sabiduría. Sin toda su ayuda el desarrollo de esta maestría no habría sido posible.

También quiero agradecer a la Universidad de la República Oriental del Uruguay y a la Agencia Nacional de Investigación por financiar el viaje a España y Portugal para participar en el congreso donde se presentó parte del trabajo de esta tesis.

También quiero agradecer a todos los amigos que me dieron una mano para que todo esto salga adelante, siempre sacándome una sonrisa y haciendo los momentos más llevaderos.

Muy especialmente, y por último pero no menos importante, estoy inmensamente agradecido a toda mi familia por aguantarme todo este tiempo, a mi madre por haberme dado tanto en todo no solo durante estos años y también mi hermano por haber bancado todos mis malos humores y acompañarme todo este tiempo.

Rodrigo Leira

Capítulo 1

Introducción

En la actualidad las computadoras son ampliamente utilizadas para modelar y computar todo tipo de fenómenos, especialmente los físico-matemáticos, complementando la teoría y experimentos con la resolución de problemas [1]. El principal aporte de estas es facilitar el estudio de dichos fenómenos llevando a cabo experimentos (simulaciones) comprobando así las teorías en que se basan. Lo interesante de las simulaciones realizadas por medio de computadores es la capacidad de realizar experimentos que de otra forma resultarían muy costosos, ya sea debido a su alto costo en recursos materiales, o al tiempo requerido para realizarlos. Sin embargo estos experimentos no son necesariamente precisos, debido a una de las características inherentes a los computadores: sus capacidades de cómputo y de almacenamiento finitos. A pesar de esto, debido al incremento exponencial de la potencia de las computadoras —que se han mantenido desde sus comienzos [2] — y al desarrollo de nuevos algoritmos, es cada vez más fácil realizar simulaciones que brinden resultados más exactos, y simular fenómenos físico-matemáticos que antes no eran posibles. En este campo, la computación gráfica es una de las áreas más relevantes, debido a la facilidad que poseen las personas de comprender elementos visuales y a la complejidad inherente en los cálculos detrás de los mismos. En el área de la computación gráfica la luz es un fenómeno que ha sido objeto de mucho estudio. La luz resulta especialmente interesante de estudiar debido a su complejidad y a la gran cantidad de fenómenos asociados a la misma (como la refracción, reflexión, dispersión, propagación, difracción, interferencia, entre otros) haciendo de esta una área de gran interés académico. A su vez dentro de los distintos métodos y algoritmos desarrollados en el estudio de la luz, aquellos que se enfocan en estudiar la iluminación global de una escena (GI por sus siglas en inglés) son de mayor trascendencia. Estos simulan los efectos de la reflexión de la luz en objetos de una escena, y cómo es esta conducida a través de los medios involucrados [3]. Entre los algoritmos que estudian la GI se encuentran aquellos que intentan obtener resultados físicamente realistas o aquellos que intentan convencer al observador por medio de artificios o heurísticas psicológicamente satisfactorios.

1.1. Problemas de optimización

Como en la mayoría de los problemas importantes, la optimización es también objeto de estudio en lo referente a la luz. En esta área, algoritmos cuyo objetivo es la búsqueda de soluciones óptimas han sido propuestos para distintos tipos de problemas relativos a la luz [4, 5, 6]. En los problemas de optimización combinatoria se buscan soluciones óptimas

dentro de un espacio de búsqueda dado. Dentro de esta área existen los problemas de optimización donde el objetivo del problema es el de encontrar la solución con el mejor valor retornado para la función objetivo [7]. Estos problemas se caracterizan por poseer un espacio de búsqueda. Para la mayoría de este tipo de problemas, el espacio de búsqueda crece de forma exponencial en relación a sus variables. Esto causa que la evaluación de todas las soluciones no sea posible y trae la necesidad de la utilización de heurísticas.

Si bien los principios detrás de la interacción entre la luz y los objetos son bien comprendidos, la eficiencia computacional de los algoritmos de GI es objeto de estudio aún en la actualidad. Muchos modelos y algoritmos han sido creados y mejorados en el correr del tiempo [8, 9, 10, 11] para resolver este tipo de problemas de manera eficiente. Cuando se intenta resolver un problema de optimización es importante que los cálculos sean eficientes, de otra manera los algoritmos están sujetos a grandes tiempos de respuesta.

1.2. Motivación y objetivos

El verbo iluminar refiere a la acción de proyectar luz sobre un cuerpo, o conjunto de cuerpos. Y se conoce como iluminación al conjunto de luces que se instala, en un determinado entorno, con la intención de afectarlo a nivel visual por medio de la luz. La correcta iluminación de un entorno es un problema de gran importancia en el diseño arquitectónico (conocido como diseño de iluminación) y es una de las principales áreas de aplicación de los algoritmos de GI. La luz y cómo esta se transmite es de vital importancia en el diseño arquitectónico debido a que es a través de la luz que las personas los perciben. En el diseño arquitectónico se deben considerar un conjunto amplio de objetivos y restricciones, conocidas como intensiones de iluminación (LI por sus siglas en inglés) [12]. Dicho conjunto de LI deben ser satisfechas con el propósito de asegurar condiciones de eficiencia, atractivo y aspectos funcionales en el ambiente resultante. Para llevar a cabo diseños de iluminación en ambientes arquitectónicos, los diseñadores utilizan una amplia gama de herramientas de software profesionales, llamadas herramientas de diseño asistido por computadora (CAD por sus siglas en inglés), que asisten en el diseño de la iluminación. Dialux [13] o Relux [14] son ejemplos de esas herramientas CAD. Sin embargo uno de los principales problemas de las herramientas CAD es que están basadas en estrategias directas, que retornan resultados a partir de una entrada dada. Retornan estadísticas, resultados visuales e informes, a partir de una configuración dada por el diseñador. Estas herramientas no buscan configuraciones óptimas ni facilitan su búsqueda sino que proporcionan resultados objetivos a diseños previamente ingresados.

La principal dificultad de las estrategias directas es que la cantidad de configuraciones a ser evaluadas es muy grande, por lo que resulta difícil evaluar todo el dominio de configuraciones posibles. Este proceso al ser llevado a cabo por diseñadores, resulta repetitivo y tedioso además de demandar mucho tiempo.

Siendo la simulación por computador una herramienta ampliamente utilizada para representar fenómenos físico-matemáticos, más específicamente la luz en su interacción con la materia, resulta importante el estudio y creación de un software que sea capaz de computar configuraciones de luminarias óptimas a partir de un conjunto de LI. Este tipo de problemas es conocido como problemas inversos de iluminación (ILP [15] por sus siglas en inglés). Debido a lo inmenso del dominio de configuraciones posibles, es necesario

utilizar heurísticas para realizar un análisis inteligente del dominio de soluciones de forma de encontrar buenas soluciones sin necesidad de evaluar todo el dominio (por fuerza bruta), lo cual resultaría en un consumo excesivo de recursos de cómputo. Si bien las heurísticas no proveen la solución óptima, el resultado de estas puede ser utilizado como una buena solución inicial que el diseñador puede luego ajustar. También puede realizar nuevos procesos de optimización cambiando los objetivos y restricciones de la búsqueda.

En este contexto, el objetivo de esta tesis es proponer nuevas técnicas de simulación y de iluminación global para resolver el problema de la selección óptima de luminarias como un ILP y de manera eficiente.

1.3. Contribución de la tesis

En el contexto de técnicas de asistencia al diseño de iluminación, se propone un método novedoso para el cálculo de configuraciones de luminarias. Este método busca soluciones óptimas considerando las restricciones (LIs) impuestas por el diseñador, considerando cientos de posiciones distintas a la vez de miles de luminarias. La información de la emisión de las luminarias es obtenido utilizando los datos reales de emisión de las luminarias. Estos datos son extraídos de los archivos fotométricos (que brindan los fabricantes de las mismas). Partiendo de una base de datos de luminarias y una geometría cualquiera, es posible dar como entrada al método un conjunto de restricciones que las soluciones deben cumplir y un conjunto de objetivos. A cambio, el algoritmo retorna de forma eficiente una buena solución (óptima en el subconjunto de soluciones evaluadas) dando como salida la posición, tipo de luminaria y rotación que cada luminaria debe tener. Algunos trabajos [16, 17, 18] han estudiado este problema y considerado estas variables pero sin considerar variedades grandes de tipos de luminarias ni su emisión real.

Además de la evaluación del algoritmo, se realiza la evaluación de las técnicas propuestas en un conjunto de problemas interesantes, probando su utilidad, y también se realiza el estudio de un caso real. Estas evaluaciones exploran su uso en el área de diseño arquitectónico por medio del cálculo de iluminaciones realistas para escenas complejas y de grandes dimensiones, de forma eficiente. Los resultados muestran buenas soluciones, que pueden ser utilizadas como punto de partida en el estudio de la iluminación. A la vez, el método puede ser utilizado para optimizar o evaluar propuestas concretas realizadas por expertos.

En resumen, las principales contribuciones del trabajo son:

- el estudio de las técnicas de iluminación global así como el análisis de sus trabajos relacionados;
- el estudio de las técnicas de resolución de problemas inversos de iluminación;
- la propuesta de varias técnicas y algoritmos que permiten el cálculo de la iluminación a partir de la información de los archivos fotométricos;
- la propuesta de mejoras a la eficiencia computacional de las técnicas planteadas que permiten realizar optimizaciones considerando dominios de grandes dimensiones;

- un profundo análisis experimental de las técnicas propuestas, que demuestra la capacidad de estas para hallar soluciones óptimas o satisfactorias de iluminación arquitectónica considerando objetivos y restricciones realistas.

1.4. Estructura de la tesis

La tesis se estructura de la siguiente forma:

El Capítulo 2 presenta los conceptos relacionados a todas las áreas de conocimiento, algoritmos, técnicas y conceptos relacionados a los objetos de estudio de la tesis. Estos están principalmente relacionados a la luz y las técnicas de iluminación global así como al área de optimización. En dicho capítulo se realiza un estudio del estado del arte de cada una de las áreas involucradas en esta tesis. Se comienza describiendo la iluminación global, detallando la ecuación utilizada para modelarla junto a sus aspectos físico-matemáticos y presentando los trabajos realizados en las últimas décadas. Luego se describe la radiosidad, principal método utilizado en esta tesis para calcular la iluminación resultante en las escenas, describiendo sus características principales, la forma en que esta se calcula y algunos de los métodos existentes en el estado del arte para realizar su cómputo de forma eficiente. En cuanto a la optimización, se introducen los problemas inversos y las heurísticas utilizadas en la tesis para realizar la búsqueda de soluciones, a la vez de presentar técnicas que permiten su cómputo de forma eficiente. Por último en este capítulo se realiza una introducción a la fotometría, un área de suma importancia para esta tesis, utilizada para obtener datos realistas de las emisiones de las luminarias.

En el Capítulo 3 se describen los conceptos, algoritmos y técnicas implementadas para realizar el cálculo de la iluminación, la arquitectura de la solución implementada y todo lo referente a los cálculos necesarios para realizar un cálculo realista de la iluminación final producida por las luminarias en la escena. Se comienza introduciendo la arquitectura, mostrando tanto la entrada como la salida del algoritmo y la comunicación de cada uno de los componentes de la misma. Luego se describe el proceso de pre-cómputo y se continúa detallando el proceso de optimización. A continuación se describe cómo se mapea la luz emitida por una luminaria en la escena, las técnicas utilizadas, las ecuaciones involucradas en su cálculo, y el proceso de construcción de la emisión a partir de los datos de un archivo fotométrico dado. Por último, se describen los detalles de implementación y cómo son utilizadas las heurísticas seleccionadas para realizar la búsqueda de soluciones.

Luego, en el Capítulo 4 se muestran los experimentos realizados para evaluar los algoritmos implementados, se realiza un análisis experimental que muestra distintos tipos de objetivos y restricciones de optimización y se exponen los resultados obtenidos para los mismos. Se comienza dando una breve introducción a los factores generales que aplican a todos los experimentos realizados, introduciendo las variables utilizadas en los casos de estudio, las escenas utilizadas, y el hardware utilizado para realizar los experimentos. Luego se analizan los resultados del algoritmo midiendo qué tan bien converge a una solución dada, minimizando la distancia de la iluminación producida para una configuración dada. El primer caso de estudio analiza la optimización referente al consumo eléctrico, siendo este uno de los principales casos de estudio de la tesis. Luego se realizan varios experimentos que intentan evaluar objetivos que son aplicados en estándares de iluminación, como la uniformidad de la iluminación producida o la cantidad de luz que

tienen ciertas superficies. Luego se evalúa la técnica de evaluación paralela de individuos como método para mejorar la eficiencia del algoritmo. A continuación se evalúa la eficiencia computacional del algoritmo, considerando las distintas variables y condiciones que controlan la eficiencia y eficacia del algoritmo (cantidad de luminarias por configuración, el uso de rotaciones y el tamaño de hemi-cubos utilizados). Luego se evalúan las heurísticas VNS y algoritmo evolutivo, realizando experimentos que permiten determinar cuál de las mismas retorna mejores resultados. El capítulo finaliza realizando el estudio de la aplicación del algoritmo en un caso real. En el caso estudiado es necesario actualizar el diseño de la iluminación en un edificio existente.

Por último en el Capítulo 5 se exponen las conclusiones finales de la tesis y se presentan las principales líneas de trabajo futuro que tienen la finalidad de la mejora de las técnicas implementadas.

Capítulo 2

Estado del arte

Este capítulo presenta una descripción general a los conceptos más relevantes relacionados con esta tesis. Se comienza con una breve introducción sobre la luz y se continúa explicando el concepto de iluminación global, luego radiosidad, se describien los problemas inversos de renderizado, problemas inversos de iluminación, optimización y por último se introduce la fotometría.

2.1. Introducción

Si bien los conceptos detrás de los fenómenos fisico-matemáticos de la interacción de la luz con la materia son bien comprendidos en la actualidad, sigue siendo un problema complejo de resolver eficientemente. Esto es debido a la múltiple cantidad de fenómenos visuales que la luz puede generar al interactuar con la materia, como por ejemplo: la reflexión, el sangrado y las causticas [3]. Esto lleva a que muchos de los algoritmos utilizados actualmente para modelar las interacciones de la luz con los objetos de su entorno, utilicen modelos que contemplan un subconjunto de todas las posibles interacciones o estas se modelen de manera parcial. En pocas palabras se utilizan modelos simplificados de los introducidos por la teoría que estudia la luz, siendo las primeras técnicas de computación gráfica utilizadas para generar imágenes realistas no más que un conjunto de ingeniosas técnicas (no necesariamente basadas en la teoría) que generaban, ante los ojos del que las visualizaba, la sensación de realismo. No es sino hasta las décadas de 1970 y 1980 cuando los avances en las técnicas y capacidad de cómputo permiten modelar la luz de mejor manera [19], siendo un ejemplo de técnicas basadas en, o más cercanas a, la física las de traza de rayos (*ray tracing* [8]) y radiosidad (*radiosity* [11]), existiendo también técnicas híbridas, utilizados para el cálculo de la iluminación global y la generación de imágenes foto realistas.

2.2. Iluminación global

El propósito principal de los modelos de iluminación es estudiar la emisión, reflexión y transmisión de la luz en el ambiente, con el objetivo de calcular el color, la intensidad y demás características de la luz en un punto dado, que pertenece al mismo, y contemplando los distintos fenómenos involucrados [19]. Algunos modelos simplificados no calculan el color para todo posible punto sino que lo hacen para puntos concretos

(visibles) que pueden ser por ejemplo determinados por los píxeles de la imagen y su proyección en la escena tomando el punto de vista de la cámara.

En general, la luz que llega al punto A desde otro punto B es considerada como luz directa si esta es emitida por B. Si dicha luz fue reflejada o transmitida antes de llegar a A, en otro punto C (o conjunto de puntos), entonces es considerada como luz indirecta [3]. La luz está compuesta por ambos tipos; directa e indirecta. En computación gráfica se consideran como modelos de iluminación local a aquellos que consideran únicamente la iluminación directa y modelos de iluminación global a aquellos que consideran también la iluminación indirecta. En el trabajo de Jensen [20] podemos encontrar una definición más formal que define a la iluminación global como la simulación, basada en las características físicas de la luz, que toma en cuenta toda la dispersión de la luz en un modelo sintético. Siendo su principal objetivo simular todas las reflexiones de la luz en dicho modelo y permitiendo predecir de manera precisa la intensidad de la luz en cualquier punto del modelo, i.e., cómo la luz es emitida por las fuentes luminosas y cómo interactúa con las superficies de la escena.

Esta tesis se basa en el cálculo de la iluminación global en un ambiente por medio de la técnica de radiosidad.

2.2.1. Ecuación de la luz

De acuerdo a Sakas et al. [21], la "ecuación completa de radiancia" relaciona la interacción global entre la luz y la materia, incluyendo fenómenos como el transporte del fotón en medios participativos, la polarización, la fosforescencia y la fluorescencia. Esta ecuación puede ser expresada brevemente en notación de operadores como sigue:

$$I = (\mathcal{M} + \mathcal{V})\left[\varepsilon + \mathcal{P}\mathcal{A}I + \mathcal{K}\mathcal{F}I\right] \tag{2.1}$$

donde I es la intensidad de la luz en la longitud de onda λ que llega al punto r desde la dirección $\vec{w}$ y en el tiempo t (ver Figura 2.1), $\mathcal{M}$ representa la atenuación de la radiancia desde el punto s, $\mathcal{V}$ es la atenuación de la radiancia en el punto a entre r y s, ε es la luz emitida en la longitud de onda λ en el punto s en dirección $\vec{w}$ y en el tiempo t, $\mathcal{P}$ es el operador de fosforescencia, $\mathcal{A}$ es el operador de absorción, $\mathcal{K}$ es la función de distribución bidireccional de superficie-dispersión (o BSSDF por sus siglas en inglés) y $\mathcal{F}$ es el operador de fluorescencia. La ecuación anterior resulta muy compleja, sin embargo

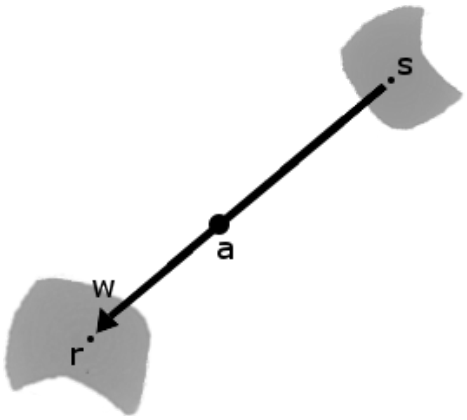


Figura 2.1: Luz que llega al punto r desde el punto s y con dirección $\vec{w}$, pasando a través del punto a.

existen las siguientes versiones simplificadas:

- "ecuación de rendering" planteada por Kajiya [22]
- "ecuación volumétrica de renderizado" planteada por Glassner [23]

En la ecuación de rendering la luz es transportada sin la presencia de un medio participativo, y la ecuación volumétrica de renderizado sí incluye un medio participativo. De acuerdo con Kajiya [22], la ecuación de rendering es definida como sigue:

$$I(x,x') = g(x,x')\left[\varepsilon(x,x') + \int_S \rho(x,x',x'')I(x',x'')dx''\right] \tag{2.2}$$

donde $I(x,x')$ representa la radiancia ($Wm^{-2}sr^{-1}$) que pasa de x' y llega a x (ver Figura 2.2), el término geométrico $g(x,x')$ es igual a 0 cuando x y x' están ocluidos entre sí y $1/r^2$ en otro caso, siendo r la distancia entre ellos. $\varepsilon(x,x')$ está relacionado con la intensidad de la luz que es emitida desde x' hacia x, y $\rho(x,x',x'')$ está también relacionado con la intensidad de la luz que es dispersada desde x'' hacia x por un parche en x' (esta es denominada función de distribución bidireccional de reflectancia, o BRDF por sus siglas en inglés). La ecuación de rendering es invariante en el tiempo (la difracción no es calculada), el medio posee un índice de refracción homogéneo y no participa por sí mismo en la dispersión de la luz. Finalmente, la ecuación no está influida por la longitud de onda o por fenómenos de polarización de la luz.

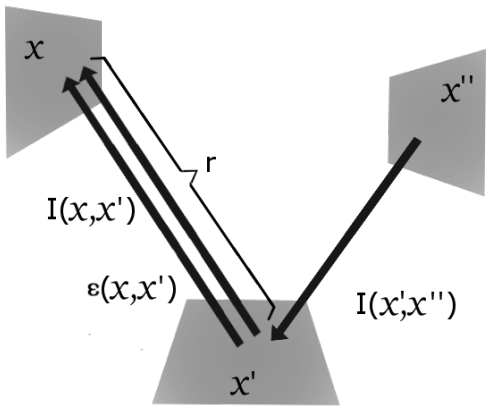


Figura 2.2: Algunos elementos de la ecuación de rendering.

Se han propuesto varias técnicas que toman diferentes enfoques a la hora de resolver estas ecuaciones, estos enfoques por lo general tratan un conjunto distinto de los fenómenos de la luz. Ritschel et al. [3] establecieron que los enfoques clásicos para computar la iluminación global, de forma interactiva, son: métodos discretos de ordenadas [24], elementos finitos (radiosidad) [10], traza de rayos de Monte Carlo [22], mapeado de fotones [9], radiosidad instantánea [25], iluminación basada en múltiples luces [26, 27], iluminación global basada en puntos [28, 29] y transferencia de radiancia pre-computada [30]. A través de estas técnicas se intenta implementar un amplio rango de fenómenos: iluminación directa (local) e indirecta (global), *ambient occlusion*, iluminación natural (aquella proveniente del sol), rebotes simple y múltiples, cáusticas, rebotes difusos y superficies brillosas ("*glossy*" en inglés), y difusión. Si bien la mayoría de las técnicas propuestas intentan resolver las ecuaciones de renderizado tomando distintos enfoques

existen algunos fenómenos que son estudiados por la mayoría. Estos fenómenos afectan significativamente el tiempo de cómputo necesario por estas y son los siguientes; varias fuentes de luz, distintos materiales para las superficies y varios rebotes de la luz. Además las técnicas antes mencionadas adolecen de los siguientes problemas abiertos; cantidad de rebotes, escalabilidad y el uso de fuentes luminosas complejas.

Esta tesis realiza el estudio de cómo calcular eficientemente la iluminación producida por fuentes luminosas complejas.

2.2.2. Trabajos relacionados

Aquí se indican algunas de las líneas de trabajo seguidas en los últimos años, y que históricamente son base de las técnicas actuales, con respecto a iluminación global que puedan ser utilizados para generar imágenes:

- Goral et al. [10], proponen la técnica de *radiosity* para el cálculo de la luz en la computación gráfica.
- Cohen et al. [31], proponen una técnica para calcular la radiosidad de manera eficiente utilizando niveles jerárquicos.
- Kajiya [22], propone una técnica basada en *Monte Carlo ray tracing*.
- Ward et al. [32], proponen una técnica denominada *irradiance caching*. Donde plantean una técnica eficiente para el cálculo de las interreflecciones entre superficies difusas y especulares utilizando métodos de Monte Carlo.
- Kuchkuda [33], propone una técnica basada en *ray tracing*.
- Jensen [9], propone una técnica denominada *photon mapping*.
- Keller[25], propone una técnica llamada *instant radiosity*. La técnica se basa en convertir la iluminación en un número enorme de puntos de luz generados automáticamente (de forma aproximada).
- Jensen [20], propone un avance sobre la técnica sobre *photon mapping*. En esta propuesta se permite calcular la iluminación global en escenas que no pueden ser manejadas por métodos sin sesgo basados en métodos de Monte Carlo.
- Sloan et al. [30], proponen la técnica *precomputed radiance transfer*. La técnica se basa en pre-computar y realizar la compresión de los efectos de iluminación y sombreado que luces distantes provocan en la escena.
- Dachsbacher y Stamminger [34], proponen la técnica de *reflective shadow maps*. En esta se considera a los mapas de sombra como fuentes de luz indirecta, con el objetivo de lograr la generación de iluminación indirecta, pero se niega la existencia de múltiples interreflexiones de luz entre las superficies.
- Walter et al. [26], proponen una técnica sobre *many-light-based global illumination*. Esta técnica utiliza un árbol binario de iluminación que por medio de cortes agrupa las fuentes de luz en clusters permitiendo reducir de forma significativa el costo necesario para calcular la iluminación de fuentes variadas. La técnica es capaz de

manejar geometrías arbitrarias, materiales no difusos y varias fuentes de iluminación, como por ejemplo luces puntuales, modelos de sol o cielo e iluminación indirecta.

- Nijasure et al. [35], proponen una técnica basada en el uso de GPUs para obtener un muestreo de la radiancia incidente en grillas uniformes tridimensionales que permite tomar ventaja del poder de cómputo y funcionalidades de la GPU para lograr mejoras significativas de performance. El algoritmo implementado no tiene dependencias más que el hardware utilizado, por lo que la aceleración obtenida depende de la aceleración del hardware.

- Hašan et al. [27], proponen una técnica que implementa *many-light-based global illumination* en GPUs. Este enfoque es capaz de computar de forma veloz, y con gran calidad, problemas con muchas fuentes de iluminación, donde tratan el problema como la suma de todas las columnas de una matriz. Su uso puede resultar muy útil para realizar una vista previa, rápida, en aplicaciones cinematográficas y arquitectónicas para el diseño de la iluminación.

- Christensen [28], propone una técnica llamada *point-based global illumination*. Esta técnica produce resultados 4 a 10 veces más rápido que técnicas basadas en *ray tracing*, utilizando menos memoria, no tiene ruido, y su tiempo de ejecución no aumenta debido al desplazamiento mapeado de superficies, sombreadores complejos o muchas fuentes de luz complejas. El método primero organiza las superficies en un *octree* (árbol octal) y calcula los datos para cada uno de sus nodos. Luego, para cada punto de recepción, se recorre el octree, rasterizando las contribuciones de color de los elementos de superficie y los clústeres, y el sangrado del color se calcula como una suma ponderada de los píxeles rasterizados. El método es significativamente más rápido y más eficiente en el uso de memoria que métodos precursores.

- Ritschel [29], propone un avance de la técnica *point-based global illumination*. El método se basa en la recolección final paralela, realizándola por completo en la GPU. La técnica realiza un micro renderzado, que a través de una representación de la escena basada en puntos jerárquicos realiza una recolección por muestras de importancia de BRDF, permitiendo la computación de la iluminación indirecta de forma eficiente (soportando superficies tanto difusas como brillantes).

- Gautron et al. [36], proponen una técnica sobre *radiance cache splatting* que es basada en la técnica de *radiance caching* y en el uso de imágenes para recoger la radiancia final de la escena. Este trabajo se enfoca en evitar la necesidad de complejos algoritmos y el uso de estructuras de datos para así poder ser aplicado en GPUs manteniendo una calidad de reproducción similar a la irradiación clásica y al *radiance caching*. Esto resulta en una mejora considerable en el tiempo de cómputo.

- Ritschel et al. [37], proponen una técnica sobre *imperfect shadow maps*. En esta técnica se evalúan mapas aproximados de sombra sobre muchos puntos de luz en cada paso de la GPU, lo que resulta en una mejora considerable en el tiempo de cómputo.

- Wang et al. [38], proponen la técnica de *photon mapping* en GPU.
- Fernández et al. [39], proponen la técnica de *low-rank Radiosity* para mejorar la velocidad de cómputo de la radiosidad.

No es el objetivo de este capítulo, ni de esta tesis, hablar en específico de cada una de estas técnicas, pero si se dan a continuación detalles generales de las técnicas más importantes y referencias que pueden ser consultadas para entender en profundidad los coneptos detrás de las mismas.

Es de destacar la existencia, entre las técnicas de iluminación global, de aquellas que dependen del punto de vista y aquellas que no. En las técnicas que dependen del punto de vista (ejemplos son *ray tracing* e *instant radiosity*) la luz no se calcula para todos los puntos de la escena sino que se calcula para aquellos que pueden ser vistos desde el punto de vista. Estas técnicas son mayormente utilizadas cuando se tienen superficies especulares. Ejemplos de técnicas que no dependen del punto de vista son *photon tracing* y *radiosity*. Estas técnicas son utilizadas cuando se tienen superficies difusas.

De los trabajos antes mencionados se destacan cuatro técnicas principales, las cuales son: *Ray tracing*, *Photon Mapping*, *Instant Radiosity* y *Radiosity*.

Ray tracing es una técnica muy utilizada en la computación gráfica basada en la traza del camino de los rayos de luz a través de píxeles en el plano de vista. Esta técnica es capaz de simular varios efectos, como la reflexión o la refracción, logrando generar imágenes que se asemejan en gran medida a las reales pero a un costo computacional realmente alto. Esta técnica fue introducida por Whitted [8] y se basa en el algoritmo de *Ray Casting* propuesto por Appel [40]. Sin entrar en gran detalle el algoritmo de *ray tracing* consiste en trazar rayos (representados por líneas de color rojo en la Figura 2.3) que parten del punto de vista (cámara en la imagen) y pasan a través del plano de vista (donde se genera la imagen) determinando un píxel. Ese píxel es pintado con el color correspondiente al que tiene la línea trazada en el punto donde intersecta con el objeto más cercano a ella en la escena. El cálculo del color de dicho píxel se realiza trazando también rayos (también representados por líneas, pero de color verde) hacia las fuentes de luz tomando en cuenta tanto oclusiones (y el tipo de superficie de los objetos que causan las oclusiones) a la vez del tipo de superficie del objeto intersectado en primer lugar. Las superficies pueden tener distintos tipos de propiedades (como por ejemplo ser superficies especulares o difusas), la complejidad de los distintos tipos de superficies pueden dar la necesidad de la traza de rayos adicionales (las reflexiones son un ejemplo donde se necesita la traza de rayos de forma recursiva para el punto dado). A su vez, a cada objeto se le es sumada la luz de ambiente.

Por otro lado, la técnica de *photon mapping* introducida por Jensen [9], es una técnica de traza de rayos basada en el concepto del fotón (*photon* en inglés) que consiste en dos pasos (ver Figura 2.4 [41]) :

1. Trazado del fotón (*Photon Tracing*) donde se construye un mapa de fotones salientes desde las luces que llegan a los distintos objetos en la escena. Este paso es independiente del punto de vista. El mapa de fotones es almacenado en la medida que los fotones colisionan con superficies difusas (o más bien superficies no especulares ya que estas no aportan información dado que la probabilidad de que un fotón incida por la dirección especular es muy pequeña y cercana a cero).

2. Radiación estimada (*Radiance Estimate*) donde se estima la radiación que recibe un punto en la escena por medio de la estimación de densidad. Este paso es

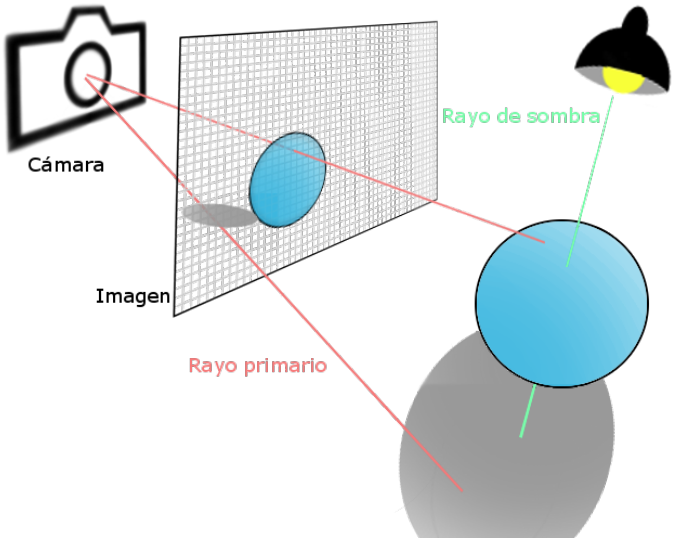
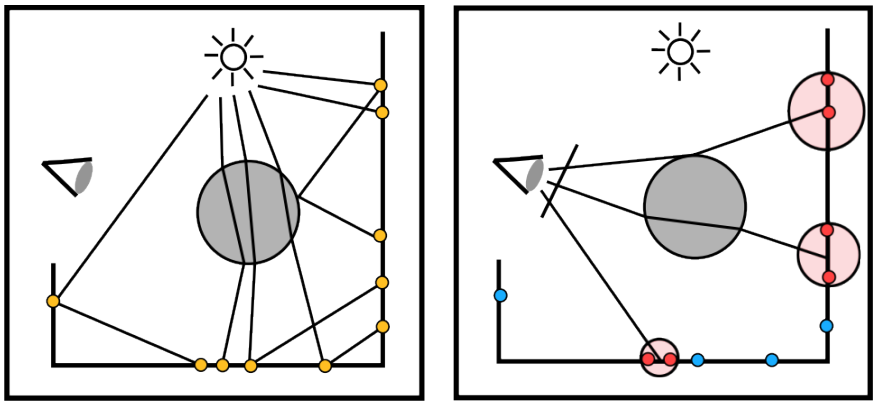


Figura 2.3: Representación simple de trazado de rayos. La imagen es generada en el plano de vista (que es dividido en píxeles). Se trazan rayos desde el punto de vista (cámara) que pasan a través de los píxeles en la imagen e intersectan con los objetos de la escena. El color del píxel es analizado calculando la luz en el objeto intersectado (tomando en cuenta su material, que puede provocar el trazado de nuevos rayos en la escena). Para el cálculo además se trazan rayos hacia las fuentes de luz para calcular la generación de sombras.

dependiente del punto de vista.

En esta técnica los fotones representan paquetes de energía, de volumen infinitesimal, que son transmitidos en la escena por las fuentes de luz que se transportan en línea recta desde su punto de partida a través de su dirección original.



(a) Trazado de fotones. (b) Radiación emitida.

Figura 2.4: Etapas de la técnica de photon mapping.

Por último tenemos la técnica de *Instant Radiosity* descrita por Keller [25]. Esta técnica también se basa en la traza de fotones y en dos pasos:

1. Una cantidad pequeña de fotones es trazado desde las fuentes de luz impactando en los distintos objetos de la escena. Estos puntos son llamados VPL por sus siglas

en inglés (*Virtual Point Lights*). Estos puntos son utilizados para calcular una primer aproximación de la radiancia difusa utilizando una recorrida casi aleatoria.

2. Se renderiza la escena una vez por cada VPL utilizando dicho VPL como la fuente de luz única y utilizando la técnica de Z-buffer [42] con mapas de sombra.

La imagen final es computada acumulando cada renderizado de cada VPL, teniendo como resultado la simulación de varios rebotes de la luz, en superficies lambertianas (ver Seccion 2.3), debido a la superposición de los mapas de sombras. Los resultados finales obtenidos son similares a los obtenidos por radiosidad.

Esta tesis se basa en la técnica de radiosidad y fuertemente en el trabajo realizado por Fernández et al. [39] extendiéndolo a fuentes de iluminación complejas, i.e., fuentes de iluminación para las cuales la magnitud de su emisión depende de la dirección que se considere (siendo el ejemplo más claro las luminarias y el contraejemplo más claro una luz difusa, que emite de forma constante la misma cantidad de luz en todas las direcciones).

2.3. Radiosidad

La radiosidad es la técnica de iluminación global utilizada en esta tesis para el cálculo de la iluminación. En esta sección se tratan los aspectos generales necesarios para comprender su utilización en el contexto del trabajo realizado. Esta técnica ha sido objeto de estudio en las últimas décadas y existen excelentes referencias en los trabajos presentados por Cohen et al. [11] y Glassner [23] que pueden ser consultadas para comprender en mayor profundidad los aspectos fisico-matématicos detrás de la misma.

El método de radiosidad es introducido por primera vez por Goral et al. [10], en el área de la computación gráfica, como un algoritmo de iluminación global, permitiendo el cálculo en escenas cuyos objetos poseen superficies lambertianas. Desde entonces esta técnica ha sido utilizada en gran diversidad de áreas de diseño y simulación por computadora [43]. El término "radiosidad" se refiere a la cantidad de energía lumínica, emitida y reflejada por la superficie, que parte de la misma y es medida por unidad de área (W/m^2) [31]. Las superficies lambertianas son aquellas que tienen reflexión difusa ideal. Estas aparentan tener un brillo constante independientemente del ángulo de vista del observador. En la Figura 2.5 se pueden ver tres tipos distintos de superficies y cómo estas reflejan la luz. Las superficies lambertianas simplifican el termino ρ de la ecuación de rendering (Ecuacion (2.2)) transformándola en la "ecuación de radiosidad" según la Ecuación 2.3).

$$B(x) = E(x) + \rho(x) \int_S B(x')G(x,x')dA' \tag{2.3}$$

En esta ecuación $B(x)$ representa la radiosidad en el punto x, $E(x)$ es la emisión de luz en x, $\rho(x)$ es la reflectividad lambertiana difusa del material en el punto x, y $G(x,x')$ es el factor geométrico que determina cuánto de la radiosidad en x' contribuye a la radiosidad de x [11]. La ecuación de radiosidad puede ser discretizada a través del uso de una metodología basada en elementos finitos, donde las superficies de la escena son discretizadas en un conjunto finito de polígonos o de "parches" y a su vez todos los términos de la Ecuación 2.3 son transformados en un conjunto finito de elementos

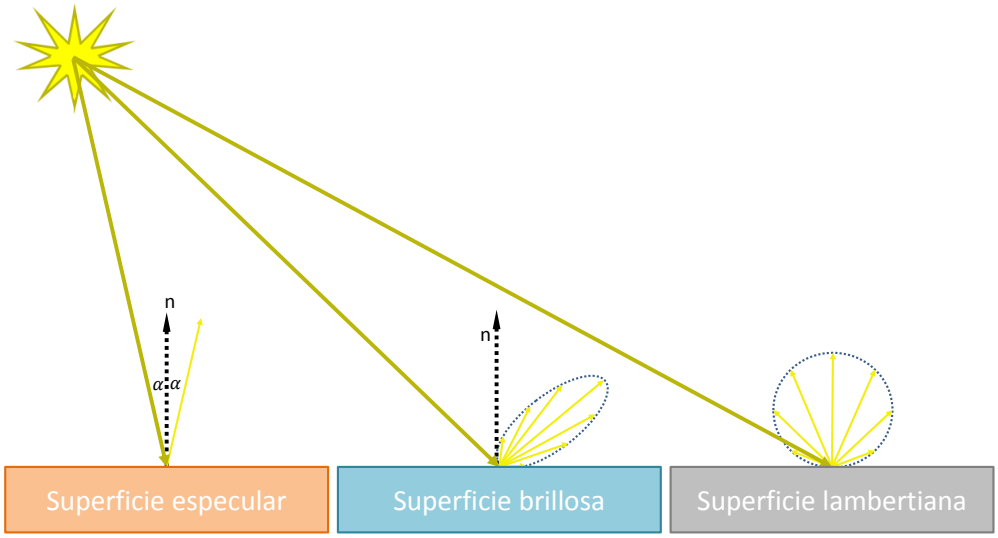


Figura 2.5: Tipos de reflexiones más comunes.

(discretos) donde para cada parche i se tendrá un único componente $B(x)$, $\rho(x)$ y $E(x)$. En otras palabras $B(x)$ es transformada en B_i, $E(x)$ en E_i, $\rho(x)$ en R_i y $G(x, x')$ en $\mathbf{F}(i, j)$ quedando así la ecuación transformada en el conjunto de ecuaciones lineales descrito en la Ecuación 2.4.

$$B_i = E_i + R_i \sum_{j=1\ldots n} B_j \mathbf{F}(i,j) \quad , \quad \forall i \in \{1 \ldots n\} \tag{2.4}$$

Este conjunto de ecuaciones es expresado de forma algebraica en la siguiente ecuación:

$$(\mathbf{I} - \mathbf{RF})B = E \tag{2.5}$$

donde $\mathbf{I}$ es la matriz identidad, $\mathbf{R}$ es una matriz diagonal conteniendo los valores R_i (con los índices de reflectancia de cada parche), $\mathbf{F}$ es la matriz de factores de forma, B es un vector conteniendo todas las radiosidades a ser encontradas, y E es otro vector conteniendo la emisión de la escena. El factor de forma $\mathbf{F}(i, j)$ es definido como un número entre 0 y 1, que indica la fracción de potencia luminosa emitida por el parche i que llega al parche j. Los factores de forma pueden ser calculados utilizando la Ecuación 2.6.

$$\mathbf{F}(i,j) = \frac{1}{A_i} \int_{A_i} \int_{A_j} G(x_{A_i}, x_{A_j}) dx_{A_j} dx_{A_i} \tag{2.6}$$

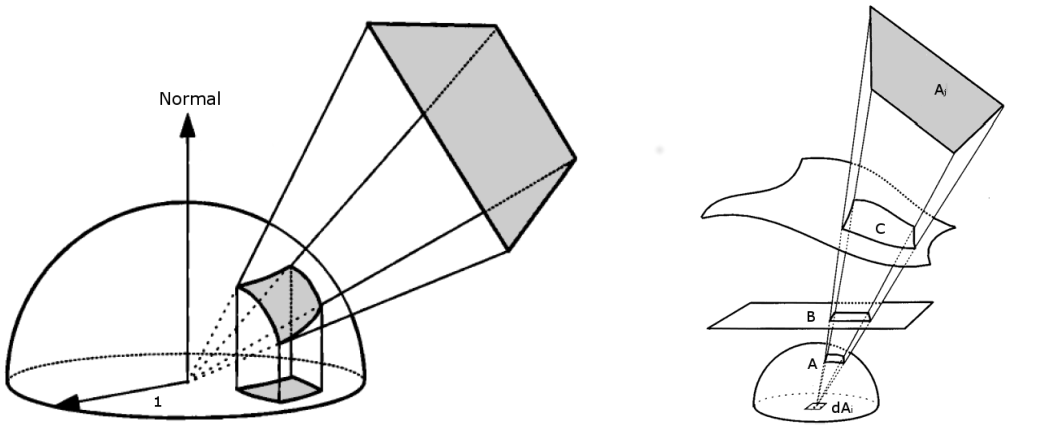
Este cálculo es complejo, debido a la necesidad de resolver la integral doble anterior y a la necesidad de calcular qué parches son visibles entre sí. Además es necesario calcular n^2 factores de forma (para una escena con n parches) lo que hace de este un proceso costoso computacionalmente. Debido a esto se han desarrollado varias técnicas que pretenden hacer más eficientes los cálculos de los factores de forma (ver Sección 2.3.1).

En el problema inverso la matriz de radiosidad $\mathbf{M} = (\mathbf{I} - \mathbf{RF})^{-1}$ contiene la información de la radiosidad global de la escena. En esta tenemos que cada elemento $\mathbf{M}(i, j)$ contiene la contribución de la emisión en el parche j de la radiosidad final en el parche i. Por lo tanto, es posible calcular la radiosidad final en el parche i por medio de un

único producto escalar entre la fila $\mathbf{M}(i,:)$ y el vector de emisión E. Sumado a esto, si se tiene una geometría (escena) fija, donde solo varía la luz emitida (posición o cantidad de luz emitida), entonces la matriz $\mathbf{M}$ no cambia. De esta manera es posible utilizar la matriz inversa de la radiosidad (o una aproximación de esta) para realizar cientos de cálculos de radiosidad por segundo y obteniendo resultados con una iluminación que considera infinitos rebotes. En la Sección 2.3.3 se dan más detalles de la matriz inversa de la radiosidad y su uso para el cálculo de la iluminación.

2.3.1. Factores de forma

El argumento geométrico utilizado para el cálculo de los factores de forma se llamada analogía de Nusselt. En esta se expresa la relación, sin tomar en cuenta la visibilidad, entre un diferencial de área y una superficie dada. Una representación gráfica de dicha analogía puede ser observada en la Figura 2.6 donde en particular en la Figura 2.6(a) tenemos una hemiesfera centrada en un diferencial de superficie cuyo radio es 1, donde se proyecta el parche. Luego de ser proyectado sobre la hemiesfera se vuelve a proyectar, de forma ortogonal, a la base de la hemiesfera (un círculo de radio 1). La fracción del área de la base que está cubierta por la proyección del parche es equivalente al valor del factor de forma entre el diferencial de área y el parche, o sea, es igual a la porción de potencia emitida del diferencial de área hacia el parche. Luego en la Figura 2.6(b) se muestra que cualquier objeto que cubra la misma área en la hemiesfera tendrá el mismo factor de forma, ya que ocupará el mismo ángulo sólido. Debido a que todos los parches



(a) Analogía de Nusselt. (b) Áreas con el mismo factor de forma.

Figura 2.6: Factores de forma del diferencial de área de un elemento, calculado usando la proyección del hemisferio y luego otra en el disco.

visibles se proyectan en áreas disjuntas en la base de la hemiesfera, la suma de todas las áreas no puede ser superior al área de la base, porque los factores de forma no toman valores negativos. Por tanto, si sobre un parche i cualquiera se genera una hemiesfera de radio 1 y se proyectan todos los parches visibles desde i, la suma de todos los factores de forma obtenidos (área proyectada dividido área del círculo) no puede ser mayor a 1, esto es debido a que la radiación que deja una superficie se conserva, en consecuencia la siguiente ecuación:

$$\sum_{j=1}^{n} \mathbf{F}_{ij} \leq 1, \forall i \tag{2.7}$$

Según Cohen et al. [11] existen dos categorías que diferencian a los algoritmos que resuelven el cálculo de los factores de forma: aquellos que utilizan un enfoque basado en el cálculo de los factores de forma a partir de dos áreas, y aquellos que utilizan un diferencial de área y un área. En estos últimos el diferencial de área representa un punto infinitesimal, que es sencillo de utilizar, y por otro lado el área depende de la forma del parche, complejizando el cálculo de los factores de forma. Debido a la complejidad de realizar los cálculos sobre distintas formas, y a que el cálculo con diferenciales de área tiene una aplicación más general, existe mayor cantidad de algoritmos que pertenecen a la categoría de diferencial y área.

Uno de los métodos más aplicados para realizar el cálculo de los factores de forma, con el enfoque de áreas, está basado en la utilización de la integración de Monte Carlo. En dicho método, dados dos parches i y j entre los que se desea calcular el factor de forma $\mathbf{F}_{ij}$, se toman al azar y con distribución uniforme n pares de puntos $[x_i, x_j]$, donde x_i pertenece al parche i y x_j pertenece al parche j. Luego se evalúa si hay visibilidad entre los pares de puntos por medio del trazado de n rayos entre cada par $[x_i, x_j]$.

Por otro lado Cohen et al. [44] proponen la técnica del hemi-cubo para realizar el cálculo de los factores. Esta técnica toma un parche i cualquiera y calcula los factores de forma $\mathbf{F}_{ij}$ con todos los parches j de la escena. Esto se realiza utilizando el algoritmo de Z-buffer [42] para proyectar cada parche de la escena en las cinco caras de un medio cubo (hemi-cubo) que son subdivididas en píxeles. El algoritmo de Z-buffer determina la distancia entre el centro de proyección (baricentro del parche i) y cada parche j que es visible a través del píxel dado, tomando para dicho píxel como parche visible aquel que retorne menor distancia (ver Figura 2.7). Luego de aplicado el método se tiene un hemi-cubo con la información de qué parche es visible a través de cada uno de sus píxeles. La principal ventaja del uso del algoritmo de Z-buffer es que su implementación es por hardware y por lo tanto su cómputo es muy eficiente. Este algoritmo ha resultado muy exitoso para el cálculo eficiente de los factores de forma y es ampliamente utilizado en técnicas de radiosidad con este objetivo. La principal desventaja del método del hemi-cubo es que el uso de píxeles causa aliasing. Para contrarrestar este problema se deben utilizar hemi-cubos con mayor resolución lo que implica mayor costo de cómputo. Otro problema es el costo de realizar cinco proyecciones independientes (una por cara del hemi-cubo) por cada parche en la escena. Coombe et al. [45] estudian este problema y proponen cambiar la proyección en un hemi-cubo por una proyección estereográfica, que permite proyectar todo el hemisferio visible en una superficie acotada, de una forma similar a la analogía de Nusselt. Heidrich et al. [46] proponen otro acercamiento que también utiliza una proyección parabólica. En esta la escena se proyecta sobre un paraboloide (técnica pensada originalmente para su aplicación en la construcción de los *environment maps* [47]).

Otro método para el cálculo de los factores de forma es el propuesto por Sillion et al. [48] en donde se proyectan los parches a un único plano que está por encima del diferencial de área por medio de un algoritmo de superficie.

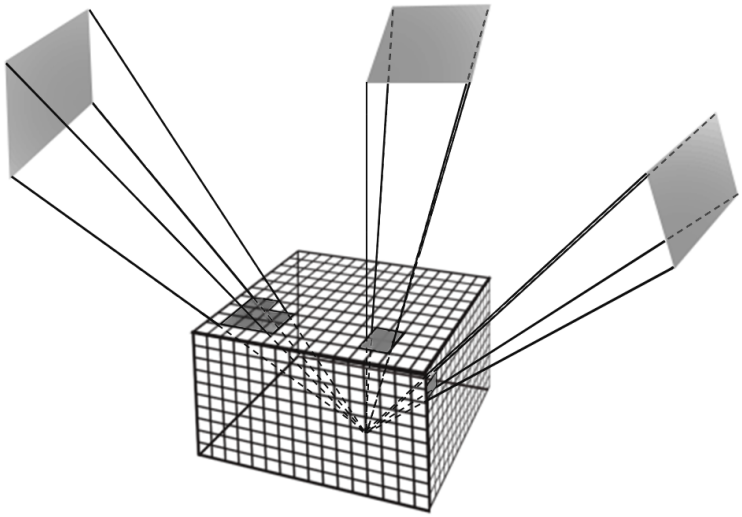


Figura 2.7: El hemi-cubo y una representación gráfica de la proyección de parches en la escena a los píxeles que se encuentran sobre las caras del mismo.

2.3.2. Radiosidad de rango bajo

Una propiedad interesante, que sucede en muchas escenas, es la existencia de coherencia espacial [49] entre los valores de la radiosidad de pares de parches que se encuentran cercanos entre si, i.e., dados dos pares de parches cercanos entre sí, estos poseen una vista muy similar de la escena. Debido a lo anterior, y siendo que cada fila $\mathbf{F}(i,:)$ es calculada basándonos en la vista de la escena desde el elemento i, y a que $\mathbf{R}$ es una matriz diagonal, tenemos que $\mathbf{RF}$ (ver Ecuación 2.5) tendrá un rango numérico bajo ya que tiene muchas filas similares. El rango bajo de la matriz $\mathbf{RF}$ permite su factorización en dos matrices $\mathbf{U}_k$ y $\mathbf{V}_k^T$ de tamaño $n \times k$ con $n \gg k$ y sin mayor pérdida de información. Existen varios trabajos que estudian las propiedades de rango bajo de la matriz de radiosidad entre ellos se encuentran: Baranoski et al. [50], Ashdown [51], Hašan et al. [27] y Fernández [52].

La radiosidad de rango bajo [52] (LRR por sus siglas en inglés) es una técnica que permite realizar esta factorización, tomando en consideración dicho rango bajo y aproximando la matriz $\mathbf{RF}$ por el producto $\mathbf{U}_k\mathbf{V}_k^T$. $\mathbf{U}_k$ y $\mathbf{V}_k^T$ son matrices de tamaño $n \times k$ con $n \gg k$ que pueden ser computadas en $O(n^2)$ operaciones, y donde se requiere $\mathrm{O}(nk)$ memoria para ser almacenadas (siendo $\mathbf{U}_k$ una matriz densa y $\mathbf{V}_k^T$ una matriz dispersa). Esta aproximación puede ser realizada sin perder información relevante de la escena y se basa en la técnica de dos niveles jerárquicos [31], donde se tienen dos mallas con distinta granularidad, una malla gruesa de k parches y una malla fina con n elementos. Por lo tanto, tomando la Ecuación 2.5 y sustituyendo $\mathbf{RF}$ tenemos la Ecuación 2.8, siendo $\tilde{B}$ una aproximación de B.

$$(\mathbf{I} - \mathbf{U}_k\mathbf{V}_k^T)\tilde{B} = E \tag{2.8}$$

Luego es posible utilizar la fórmula de Sherman-Morrison-Woodbury [53] para calcular

una aproximación de la matriz inversa de la radiosidad $\mathbf{B}$ como sigue:

$$\mathbf{M} = (\mathbf{I} - \mathbf{RF})^{-1} \approx (\mathbf{I} + \mathbf{Y}_k \mathbf{V}_k^T) = \tilde{\mathbf{M}} \quad \text{donde} \quad \mathbf{Y} = \mathbf{U}_k(\mathbf{I} - \mathbf{V}_k^T \mathbf{U}_k)^{-1} \tag{2.9}$$

Para luego llegar a la reducción final de la Ecuación 2.5 como el siguiente producto matriz-vector: $\tilde{B} = \tilde{\mathbf{M}}E$, que a su vez puede ser formulado como:

$$\tilde{B} = E + \mathbf{Y}_k(\mathbf{V}_k^T E) \tag{2.10}$$

En escenas donde la geometría no cambia y la iluminación es dinámica (E varía), parte del cómputo puede pre-computarse y ser reutilizado, porque las matrices de dimensión $n \times k$ llamadas $\mathbf{Y}_k$, $\mathbf{U}_k$ y $\mathbf{V}_k$ deben ser calculadas solo una vez. Por lo tanto, el resto de los cálculos ahora poseen complejidad $O(nk)$. Esto permite calcular la radiosidad en tiempo real para los problemas inversos de iluminación [54]. Una de las principales desventajas de la técnica de LRR es que requiere el cálculo de todos los factores de forma de la escena, lo que impacta aumentando considerablemente el tiempo necesario para pre-computar la factorización. Además depende de la existencia de dos mallas de diferente granularidad, las cuales inciden en el tamaño de las matrices y en la precisión del cálculo de la aproximación de la radiosidad.

Existen también otros métodos para calcular las matrices $\mathbf{U}_k$ y $\mathbf{V}_k^T$, como por ejemplo SVD [53] y la factorización CUR [55]. A su vez otros métodos han sido propuestos para acelerar la velocidad de cómputo de las matrices $\mathbf{Y}_k$ y $\mathbf{V}_k$, siendo ejemplos de esto los trabajos de Fernández y Besuievsky [54], y Aguerre y Fernández [56].

La técnica de LRR propuesta por Fernández [52] es utilizada para mejorar el desempeño de los algoritmos implementados en esta tesis.

2.3.3. Matriz inversa de la radiosidad

Una de las formas de calcular la radiosidad es por medio del cálculo iterativo utilizando series de Neuman [57], donde en la iteración i se calcula el aporte del i-esimo rebote de la luz en la escena (y la serie en dicho paso los i rebotes) [11]. El resultado de esta técnica es equivalente al obtenido mediante el cálculo de la matriz inversa de radiosidad $\mathbf{M} = (\mathbf{I} - \mathbf{RF})^{-1}$ y permiten el cálculo de infinitos rebotes de luz como se describe en la Ecuación 2.11.

$$B = \mathbf{M}E = (\mathbf{I} - \mathbf{RF})^{-1}E = (\underbrace{\mathbf{I} + \mathbf{RF}}_{\text{Iluminación directa}} + \underbrace{(\mathbf{RF})^2 + (\mathbf{RF})^3 + \ldots}_{\text{Iluminación indirecta}})E \tag{2.11}$$

La principal desventaja de resolver la serie de Neuman es que contiene una gran cantidad de multiplicaciones matriciales las cuales pueden resultar costosas, además de que el cálculo de infinitos rebotes requeriría del cálculo de infinitas iteraciones (en la práctica se detiene el cálculo en alguna iteración que resulte conveniente). Por lo tanto el cálculo de la matriz inversa de la radiosidad permite computar la radiosidad en un tiempo acotado de $O(n^3)$.

Esta tesis se basa en el cálculo de la matriz inversa de la radiosidad que es utilizada en cada paso de la búsqueda del óptimo para computar la radiosidad y así obtener la iluminación sobre la escena.

2.4. Problemas inversos de renderizado

La rama de estudio conocida como física teórica tiene como objetivo principal la comprensión del universo y sus fenómenos, por medio de modelos matemáticos y conceptuales que intentan representar la realidad. Estos fenómenos son modelados con la intención de encontrar lógica en los mismos y con el objetivo de predecirlos, es decir dada una descripción completa de un sistema físico (modelo utilizado como entrada), las teorías físicas nos permiten predecir qué resultados (salida) se obtendrá [58]. En este sentido, las teorías físicas describen un problema de forma directa. Se dice que un problema es directo cuando se intenta predecir la salida de un fenómeno físico a partir de las propiedades del modelo (utilizado para representar el entorno de dicho fenómeno) y su entrada. Los problemas inversos, en contraste a los problemas directos, son aquellos que intentan obtener una descripción de las propiedades de un entorno físico o ambiente (salida) a partir de los efectos medidos en el mismo (entrada).

Existen teorías físicas que intentan describir todos los fenómenos conocidos por el hombre, estos problemas pueden ser de una amplia gama, yendo desde los más simples y ampliamente conocidos hasta los más complejos (como la expansión acelerada del universo) hasta el día de hoy no muy comprendidos. Si bien las teorías físicas son en muchos casos complejas, los problemas inversos agregan un grado mayor de dificultad frente a los problemas directos, porque generalmente se posee una cantidad inmensa de soluciones (incluso es posible que existan infinitas soluciones). Además, estos problemas suelen ser complejos del punto de vista númerico.

Uno de los fenómenos físicos más estudiados por la computación gráfica es la creación de imágenes realistas, en especial las imágenes foto realistas [15], y es un problema directo. Las imágenes foto realistas son aquellas que intentan simular imágenes tomadas por medio de cámaras fotográficas [20]. Para la generación de las mismas se deben tomar en cuenta los efectos y características físico-matemáticas de la luz por medio de algoritmos y cálculos complejos que intentan simular efectos como las texturas y sombras. Todo renderizado realista está basado, de una forma u otra, en la ecuación de rendering (ver Ecuación 2.2), ecuación estudiada por Marschner [15] para la resolución de problemas inversos. En dicho estudio Marschner identifica tres clases de problemas inversos de renderizado:

- problemas inversos de iluminación (ILP), son problemas relacionados con $\epsilon(x, x')$ (ver Ecuación 2.2) donde se debe encontrar la emisión en una escena;
- problemas inversos de reflectometría, son problemas relacionados con $\rho(x, x', x'')$ donde se estudia la reflexión de las superficies;
- problemas inversos de geometría, son problemas relacionados con $g(x, x')$ donde se estudia la geometría de los objetos.

Si bien la teoría detrás de la luz y su interacción son conocidas, el cálculo eficiente de estos problemas inversos es un desafío importante.

En esta tesis se plantean técnicas para la resolución eficiente de algunos tipos de problemas inversos de iluminación.

2.5. Problemas inversos de iluminación

Según Patow y Pueyo [5], se pueden clasificar los problemas inversos de iluminación de la siguiente forma:

- problemas de emitancia inversa, donde se desea encontrar las emisiones de un subconjunto de superficies;
- problemas de posicionamiento de la luz, donde se desea encontrar la posición de los emisores para lograr cierto conjunto de LI.

También es posible realizar una clasificación de los problemas inversos de iluminación en base a la ecuación de rendering. Con este planteamiento es posible identificar las siguientes clasificaciones:

- ajuste de la radiosidad clásica de elementos finitos, donde se considera un conjunto finito de fuentes luminosas con su iluminación descrita cierta función y describiendo la iluminación en la escena como la combinación lineal de estas funciones;
- esquema de Monte Carlo inverso, donde se disparan rayos desde superficies cuyas propiedades son conocidas a aquellas superficies con propiedades desconocidas obteniendo así información de las segundas a partir de las primeras;
- como un problema de iluminación local, donde se utiliza un método simple de iluminación local en lugar de utilizar uno global, y por lo tanto la ecuación de rendering no es utilizada.

Dadas las clasificaciones anteriores, esta tesis se centra en la resolución de problemas inversos de emisión y posicionamiento de la luz especialmente enfocado a las emisiones de luminarias. En la Figura 2.8 se puede ver una representación del ILP, donde se desea optimizar la configuración de luminarias, en concreto sus parámetros, que son inferidos a partir del conjunto de LI establecidas por el diseñador para las superficies seleccionadas.

En los problemas inversos de iluminación, el problema de resolver la emisión de una superficie dada la reflexión de esta es dado por la Ecuación 2.12 (siempre y cuando las superficies sean lambertianas). Dicha ecuación puede ser obtenida directamente a partir de la Ecuación 2.5.

$$\mathbf{RF}E = (\mathbf{I} - \mathbf{RF})C \tag{2.12}$$

En esta ecuación, E es la emisión que es una incógnita, C es la radiosidad reflejada conocida de todas las superficies ($C = B - E$). Por lo antes dicho tenemos que $(\mathbf{I} - \mathbf{RF})C$ es un vector definido.

El cálculo de E a través de la inversión de la matriz $\mathbf{RF}$, ya sea por medio de la descomposición truncada en valores singulares (o TSVD por sus siglas en inglés) o la MTSVD (TSVD modificada) [59], ofrece resultados limitados debido a dos problemas principales:

- los valores hallados de E pueden ser negativos[4];
- es un problema "mal planteado" (de la palabra "$ill - posed$" en inglés).

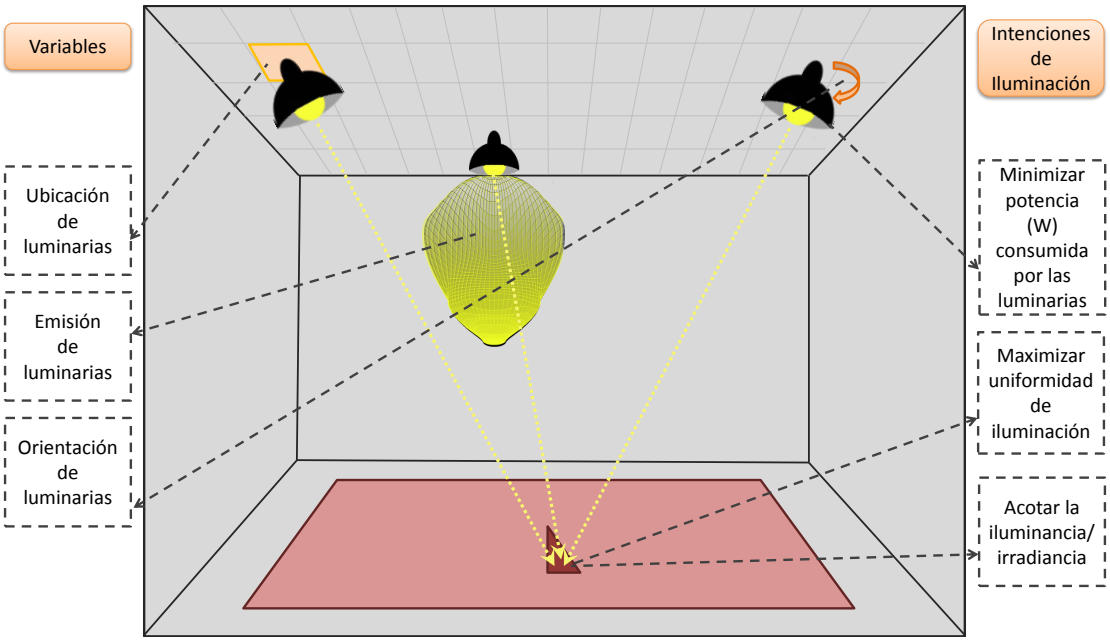


Figura 2.8: Figura que muestra un ejemplo de ILP. Las variables pueden ser la posición, emisión (la distribución de la intensidad luminosa) y orientación de cada luminaria en la configuración. Las LI son establecidas por el diseñador e incluyen objetivos y restricciones de optimización.

Resultados negativos en emisiones no son soluciones posibles debido a que se estaría indicando que es posible emitir obscuridad, quitándole luz a otros parches. Por otro lado según Hadamard [60] los problemas "bien planteados" son aquellos que cumplen:

- existe solución numérica;
- la solución numérica es única;
- la solución numérica depende continuamente de los datos (condiciones iniciales).

Los problemas mal planteados son aquellos que no cumplen dichas condiciones. El hecho de que los ILP sean problemas mal planteados es consecuencia del bajo rango numérico de la matriz **RF**, lo que resulta en que cambios pequeños en la entrada C implican cambios grandes en la solución E [61, 62]. Sumado a lo anterior, los valores de C son estimados, no se conocen sus valores exactos.

Dados todos los problemas antes mencionados, los ILP son en general resueltos como problemas de optimización [4, 5, 6], donde el objetivo es encontrar la mejor solución, la más cercana al objetivo planteado, o la que mejor cumple los objetivos y restricciones.

2.6. Optimización

La optimización matemática es el área que trata sobre la selección del mejor elemento (utilizando algún conjunto de criterios) dentro de un conjunto de elementos dado [63]. El caso general consiste en maximizar o minimizar una función real, tomando sistemáticamente valores de entrada dentro del domino permitido y calculando su valor según la función dada. Los algoritmos de optimización son objeto de mucho estudio en las áreas de

matemática y computación. La matemática provee los conceptos teóricos, mientras que la computación provee los medios para realizar una cantidad enorme de evaluaciones en poco tiempo. El modelado computacional se está convirtiendo en el tercer paradigma de las ciencias modernas, como predijo Ken Wilson [1]. La demanda creciente de precisión y de complejidad en las estructuras y sistemas, hacen que el modelado y la simulación consuma cada vez más recursos computacionales (tiempo y memoria). Aun así, el diseño impulsado por la simulación se convierte en una obligación para un número creciente de áreas, generando la necesidad de metodologías de optimización sólidas y eficientes que produzcan diseños satisfactorios, incluso en presencia de objetivos analíticamente intratables y de recursos de cómputo limitados [1].

Los problemas de optimización combinatoria son aquellos que consisten en la búsqueda de soluciones óptimas dado un espacio de búsqueda finito [64]. Dentro de esta existen los problemas de optimización, donde los candidatos son evaluados por una función objetivo (usualmente denominada función de aptitud o $fitness$) que indica qué tan cercana se encuentra la solución de alcanzar el conjunto de objetivos (y restricciones) que se imponen en la búsqueda, siendo la finalidad del problema encontrar la solución con el mejor valor retornado para la función objetivo [7]. Estos problemas se caracterizan por poseer un espacio de búsqueda (que contiene potenciales soluciones al problema particular). Para la mayoría de este tipo de problemas, el espacio de búsqueda crece de manera exponencial en relación a las variables que este considere. La complejidad computacional de estos problemas se puede dividir en dos subclases, aquellos que pueden ser resueltas por una máquina de Turing determinística en tiempo polinómico (clase P) y por otro lado se tiene la clase que comprende aquellos problemas que pueden ser resueltos por una máquina de Turing no determinística en tiempo polinomial (clase NP) [65].

La pregunta de si $NP \subseteq P$ o si $P = NP$ es uno de los problemas abiertos más prominentes en las ciencias de la computación (ver Figura 2.9). Dado que el mejor algoritmo conocido para resolver problemas NP-Difícil es de complejidad exponencial, si se aumenta el tamaño del problema este rápidamente se convierte en inviable para su resolución en tiempos razonables. Siempre es posible resolver una versión lo suficientemente pequeña (con dominio acotado o suficientes restricciones) por medio de algoritmos que utilicen fuerza bruta. También es posible agrandar el dominio de problemas NP de forma que resulte intratable para cualquier capacidad de cómputo dada.

Si bien existen estrategias que intentan paliar la capacidad de cómputo contemporánea, como por ejemplo la computación distribuida [66, 67, 68], siempre existe un límite impuesto por los recursos disponibles en el momento dado. Sin embargo, en muchos problemas no es necesariamente un requerimiento encontrar el óptimo global o la mejor solución al problema, sino el encontrar una solución lo suficientemente buena y en un tiempo razonable. Esto sucede en la mayoría de los problemas de interés práctico. Es aquí donde surge la necesidad del uso de heurísticas. Una heurística es una técnica diseñada para resolver un problema de forma rápida, encontrando una solución aproximada [69]. Es importante destacar que por su naturaleza, cada heurística no siempre retorna la misma solución dado el mismo problema y entrada, ni garantiza encontrar la mejor solución a un problema, en contraposición a los métodos exactos que sí lo hacen. Por otro lado, una metaheurística es un procedimiento o heurística de alto nivel diseñado con el objetivo de encontrar, generar, seleccionar o guiar una heurística subordinada de forma inteligente [70].

Según Blum y Roli [71], las metaheurísticas se caracterizan por:

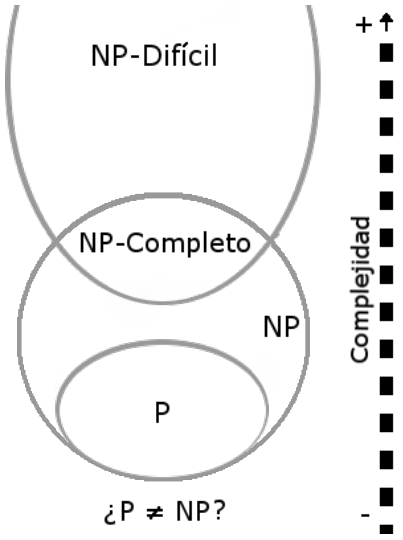


Figura 2.9: Diagrama con las distintas clasificaciones de complejidad de los problemas de optimización. La relación entre las clases P y NP es aún objeto de estudio.

- ser algoritmos aproximados y usualmente no deterministas;
- aplicar técnicas para evitar quedar atrapados en ciertas áreas del espacio de búsqueda (alrededor de óptimos locales);
- ser de aplicación general, i.e., no son especificas a un problema;
- usar conocimiento del dominio para optimizar la búsqueda.

Existen distintos tipos de familias de modelos de optimización que son utilizadas en la literatura para formular y resolver problemas de toma de decisiones. Los modelos de optimización clásicos pueden ser divididos en las siguientes clases:

- modelos de optimización combinatoria [72];
- modelos no-analíticos [73];
- modelos de programación matemática [74] (que son divididos en contiguos, enteros, mixtos, y también en lineales y no-lineales);
- modelos de satisfacción de restricciones [75].

Los modelos más exitosos están basados en programación matemática y restrictiva.

Esta sección no pretende ser más que una introducción general a la optimización. Existen varios trabajos que presentan una introducción más profunda a los conceptos relacionados al estudio de la optimización, entre ellos se encuentran: Garey y Johnson [65], Kalyanmoy [76] Coello et al. [77], Korte y Vygen [7], Talbi [78] y Luenberger y Ye [79].

2.6.1. Optimización en problemas inversos de iluminación

La optimización es también objeto de estudio en lo referente a la luz y es un proceso de suma importancia en los ILPs. Resolver un ILP significa colocar los emisores en

una escena de forma de satisfacer un conjunto de LI. Este proceso fue propuesto como alternativa a resolver una ecuación lineal, que resulta mal planteada (*ill-posed*) debido a las características de bajo rango de la matriz **RF** [54]. En la literatura es posible encontrar muchos modelos y algoritmos que buscan resolver este problema de forma eficiente. Sin embargo no existe ningún método que se haya impuesto o destaque sobre los demás. En el trabajo de Patow y Pueyo [5] se pueden encontrar los siguientes métodos como los más utilizados:

- recocido simulado (*simulated annealing*) [80];
- descomposición en valores singulares truncada (MTSVD por su sigla en inglés) [59];
- mínimos cuadrados y mínimos cuadrados restrictivos [81].

También existen trabajos que exploran la utilización de otras técnicas como *hill climbing* [82] o *beam search* [83]. Castro et al. [84, 85] realizó experimentos con una gran variedad de algoritmos.

Al modelar el problema se construye un modelo matemático abstracto que usualmente implica realizar aproximaciones y, en algunos casos, quitar elementos debido a su complejidad matemática. Una vez que se tiene un modelo para el problema, el proceso de optimización es ejecutado. La solución hallada quizá no sirva en la realidad, por eso debe ser analizada en la práctica por el diseñador quien determinará si es una solución acaptable.

Existen varios trabajos que proponen técnicas focalizadas en resolver el ILP. Schoeneman et al. [86] proponen una técnica que permite determinar la configuración de la iluminación en un ambiente, donde las fuentes luminosas tienen posiciones fijas, y se determinan los colores e intensidades de estas intentando acercarse lo máximo posible a lo que el diseñador pinta (de forma interactiva) en la escena. Otro interesante trabajo es el propuesto por Kawai et al. [87]. En este trabajo un conjunto de objetivos y restricciones es establecido por el diseñador, y utilizado como entrada del proceso de optimización. Los autores consideran tanto luces difusas (descritas únicamente por su emisión) como direccionales (descritas por su posición, dirección y distribución luminosa). Las posiciones de las fuentes luminosas son fijas y no forman parte de la optimización, siendo variables del proceso de optimización la emisión (representadas como funciones cosenoidales, no con emisiones calculadas a partir de los datos de archivos fotométricos), las direcciones, la radiosidad y la reflectividad. La principal desventaja de los métodos anteriores es que resuelven el ILP considerando materiales difusos únicamente, ya que los algoritmos están basados en radiosidad. Por otro lado en el trabajo propuesto por Costa et al. [88] también se intenta brindar herramientas de asistencia a los arquitectos y diseñadores de iluminación, al proveer otro método que resuelve el ILP. Los autores proponen una técnica que permite optimizar la geometría de la escena y los materiales utilizados, y optimizar los objetivos de iluminación establecidos por el diseñador, utilizando el programa *Radiance* (Ward [89]) como método de iluminación global. Un trabajo inspirado en el trabajo de Schoeneman et al. [86] es el propuesto por Pellacini et al. [90], donde es posible considerar más efectos, como sombras y reflexiones, además de optimizar las posiciones de las fuentes luminosas (las posiciones no son fijas o preestablecidas).

El objetivo de esta tesis es el estudio e implementación de técnicas que permitan optimizar la ubicación, intensidad luminosa y orientación de luminarias de manera que

satisfagan el conjunto de LIs propuestas por el diseñador dándole la posibilidad de utilizarlo como punto de partida en su diseño. Este es un problema de optimización combinatoria, donde existe una configuración por cada combinación de los posibles valores de las variables consideradas. A su vez, la complejidad de la función objetivo que se analiza (radiosidad), y las restricciones aplicadas en la búsqueda, hacen que no sea posible encontrar un método sencillo de dirigir la búsqueda de manera inteligente. Es por esto que se evalúan dos técnicas para la búsqueda de soluciones óptimas: VNS y Algoritmos Evolutivos.

2.6.2. Metaheurística Variable Neighbourhood Search

La metaheurística Variable Neighbourhood Search (VNS) es propuesta por Mladenovic y Hansen [91]. Esta es una metaheurística que tiene como objetivo resolver problemas de optimización combinatoria y optimización global. La principal idea detrás del VNS es la sucesiva exploración de vecindarios (usualmente anidados), donde un conjunto finito y aleatorio de representantes es seleccionado con el fin de encontrar una solución que sea mejor a la mejor obtenida hasta el momento (en iteraciones previas). El VNS se basa en tres puntos [92]:

- un óptimo local perteneciente a un vecindario no es necesariamente un mínimo de otro vecindario;
- un óptimo global es un óptimo local de todos los vecindarios posibles;
- en muchos problemas el óptimo local de cada vecindario es relativamente cercano al del resto de los vecindarios.

En cada vecindario se utiliza una búsqueda local para encontrar el óptimo local, generando distintos óptimos locales para cada vecindario y donde se tiene que el óptimo global es el óptimo local de alguno de los vecindarios [78]. Lo interesante de este método es que realiza una exploración sistemática pasando de vecindario en vecindario de forma de escapar de los óptimos locales, y por lo tanto aumentando las probabilidades de encontrar el óptimo global [78] (ver Figura 2.10 [78] y Algoritmo 1).

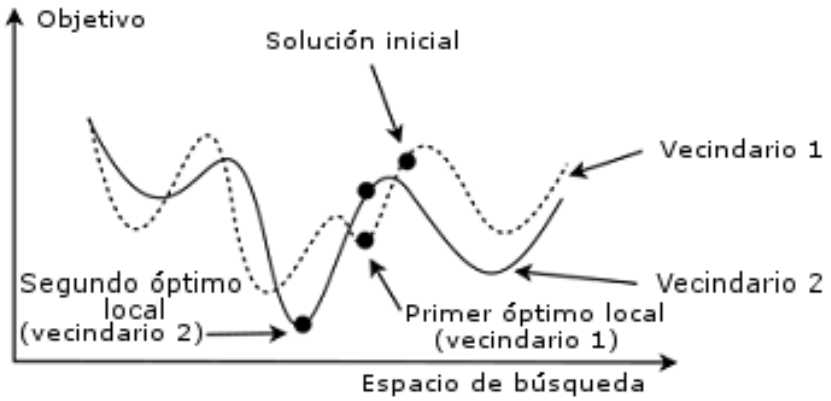


Figura 2.10: VNS metaheurística aplicada a dos vecindarios. El primer óptimo local es obtenido del primer vecindario, luego en el segundo vecindario se encuentra un nuevo óptimo local.

Algoritmo 1 Algoritmo `VNS` (basado en pseudocódigo extraído de [93]).

Input: Vecindarios
Output: Sol_{mejor}
Sol_{mejor} = solucionAleatoria()
while ¬condicionDeParada() **do**
  for $Vecindario_i$∈Vecindarios **do**
    $Vecindario_{sol}$ = calcularVecindario(Sol_{mejor}, $Vecindario_i$)
    $Sol_{candidata}$ = obtenerCandidatoAleatorio($Vecindario_{sol}$)
    $Sol_{candidata}$ = localSearch($Sol_{candidata}$)
    if (fitness($Sol_{candidata}$) ≤ fitness(Sol_{mejor})) **then**
      Sol_{mejor} =$Sol_{candidata}$
      break
    end if
  end for
end while
Return Sol_{mejor}

Otras metaheurísticas han sido creadas basándose en el VNS, como son las técnicas de VNDS [94] (Variable Neighborhood Decomposition Search), SVNS [91] (Skewed Variable Neighborhood Search), RVNS [94] (Reduced Variable Neighborhood Search). Estas variantes no están relacionadas directamente con el cambio de vecindarios, concepto principal del VNS. En la técnica VNDS el problema de optimización se descompone en subproblemas, principalemente enfocado a problemas con cantidades grandes de instancias. Mientras que la técnica de SVNS se enfoca en cómo mejorar la salida de grandes valles [78]. El RVNS elimina del algoritmo VNS original el paso de la búsqueda local, beneficiosa cuando se tiene una cantidad de instancias grandes donde el costo del cálculo de cada una es alto [94].

Lo simple e intuitivo de este método hace que sea un buen candidato para su modificación creando nuevas técnicas incluso híbridas, por ejemplo combinándola con Tabu Search. Esto hace que la técnica sea objeto de estudio constante, donde continuamente surgen nuevas variantes o combinaciones con otras metaheurísticas [95].

Un ejemplo de propuesta de utilización del método VNS para la búsqueda de soluciones óptimas en el contexto de problemas ILP es el presentado por Fernández y Besuievsky [96]. En dicho estudio se introduce una técnica que reduce de forma significativa el tiempo de ejecución mediante la aplicación de distintos métodos. La propuesta se basa en que un problema de optimización consiste en encontrar la mejor solución de todas las soluciones factibles, que se definen a través de un conjunto de restricciones. Para fines de iluminación, cada restricción está relacionada con una intención de iluminación para la totalidad o parte de la escena.

En esta tesis se estudia el uso de la metaheurística VNS para realizar la búsqueda de soluciones en ILPs.

2.6.3. Algoritmos evolutivos

Las técnicas de computación evolutiva conforman una familia de metaheurísticas estocásticas utilizadas para resolver variados problemas en las áreas de búsqueda y optimización combinatoria (entre otras). Estas basan su funcionamiento en mecanismos

análogos a los establecidos por la teoría construida alrededor del estudio de la evolución natural de las especies biológicas. La evolución natural de las especies se puede ver como un proceso de optimización en sí mismo, donde el objetivo es la supervivencia de los individuos que interaccionan entre sí y con un entorno de recursos limitados (restricciones). Los algoritmos evolutivos intentan establecer una analogía con el proceso evolutivo biológico, la selección natural y la supervivencia del más apto.

Los primeros trabajos en esta área fueron propuestos por Alan Turing en 1948 proponiendo implementar programas automodificables capaces de jugar ajedrez y simular otras actividades inteligentes por medio de técnicas evolutivas. En las siguientes décadas se desarrollaron numerosas variantes de algoritmos evolutivos, que se diferencian entre sí por los modelos utilizados para realizar la evolución de sus poblaciones y por el tipo de operadores de evolución utilizados. Los ejemplos más significativos de estudios y avances en este área son los realizados por Goldberg [97], Back et al. [98] y Mitchel [99].

Rechenberg [100] introduce por primera vez las estrategias de evolución. En su propuesta, el método de optimización consta de individuos compuestos por números reales y optimiza los parámetros en problemas de diseño en ingeniería. Esta propuesta sería desarrollada luego por Schwefel en los años 70. En su versión más simple se genera un descendiente por parte de un individuo padre. Esto es realizado por medio del operador de mutación. Este nuevo individuo reemplaza a su progenitor en la población.

Dos modelos avanzados de estrategias de evolución fueron formulados por Schwefel [101]; que se denominan (μ, λ) y $(\mu + \lambda)$. En ambas estrategias de evolución un conjunto de padres genera un conjunto de descendientes. Estas estrategias se diferencian entre si por la forma en que estas reemplazan a sus progenitores, en modelos donde m padres generan l descendientes. En el modelo (μ, λ) la selección se realiza solamente entre los descendientes, mientras que en el modelo $(\mu + \lambda)$ los padres y los descendientes compiten entre sí.

En las siguientes décadas se realizaron propuestas de numerosas estrategias de evolución. El mecanismo evolutivo caracteriza a esta familia de métodos siendo el operador de mutación su base, sin embargo en propuestas más recientes se ha incorporado la recombinación como operador secundario [98].

Los algoritmos genéticos, introducidos por Holland [102], incluyen el uso de genomas y operadores de mutación, inversión y cruzamiento. Estas son las técnicas de computación evolutiva más difundidas en la actualidad y se basan fuertemente en una simulación de la evolución natural de los seres vivos, utilizando una población de soluciones potenciales que evoluciona en el correr de la ejecución por medio de interacciones y transformaciones. Los individuos de cada población se esfuerzan por sobrevivir emulando la selección del proceso evolutivo, proclive hacia los individuos más aptos de la población. Los sobrevivientes serán aquellos que formen parte de la siguiente generación. La aptitud de cada individuo es evaluada por medio de una función de fitness. En el proceso y luego de la generación de determinado número de generaciones, este mecanismo de seleccion lleva a la población a converger a una solución cercana al óptimo del problema.

Lo anterior no es más que una introducción a los conceptos básicos y principales de los algoritmos evolutivos, pero existen trabajos que explican en mayor profundidad los conceptos detrás de la teoría, tales como Goldberg [97] y Spears [103].

En lo que respecta al uso de algoritmos genéticos en la resolución de ILPs, Elorza et al. [104] proponen un método que utiliza un pincel de luz. Este trabajo está basado en el trabajo de Schoeneman et al. [86]. Para el cálculo de la iluminación se utiliza un motor

gráfico basado en radiosidad. Además de este trabajo, Ferentinos et al. [105] utiliza algoritmos genéticos para optimizar la iluminación en invernaderos, tomando en cuenta solamente la luz directa y optimizando el consumo de las luces utilizadas y la cantidad de luces utilizadas. Por otro lado, Delepoulle et al. [106] utiliza algoritmos genéticos para posicionar luces de forma óptima pero considerando la iluminación directa e indirecta y la posición de las mismas.

En esta tesis se estudia el uso de algoritmos evolutivos basados en el modelo de evolución (μ, λ) como método para realizar la búsqueda de posibles soluciones.

2.6.4. Ejecución paralela

La utilización de metaheurísticas que permitan su evaluación de forma paralela ayuda a reducir el tiempo total que toma realizar la búsqueda de soluciones [78]. Los objetivos que se buscan alcanzar cuando se desea aplicar métodos de paralelismo son:

- Reducir los tiempos totales de ejecución
- Mejorar la robustez
- Mejorar el resultado obtenido en promedio por el algoritmo (calidad del resultado)
- Resolver problemas a mayor escala

Existen muchas formas de realizar la evaluación paralela. Las arquitecturas de computadores propuestas en la taxonomia de Flynn [107] plantean cuatro clasificaciones distintas que se basan en el número de flujos de instrucciones y datos concurrentes, que pueden ser utilizadas para reducir el tiempo necesario para realizar el cómputo si este es paralelizable.

El VNS es una técnica que es fácilmente paralelizable, ya que la evaluación de cada individuo seleccionado en el vecindario es totalmente independiente de la evaluación del resto. La única excepción es que el algoritmo no sigue seleccionando individuos al encontrar una candidato que evalúe mejor que la mejor solución hasta el momento (ver Algoritmo 1 línea 11). Además utilizando la matriz inversa de la radiosidad, para resolver el ILP por medio de la Ecuación 2.11, tenemos que es posible transformar el producto matriz-vector $\mathbf{M}E$ en un producto matriz-matriz de la forma $\mathbf{ME}$ (donde $\mathbf{E}$ contiene cada uno de los E de los individuos del vecindario a ser evaluados) la cual es optimizable por medio de optimizaciones existentes en las librerías de algebra lineal numérica.

Un ejemplo de aplicación de estas técnicas para la mejora de los resultados obtenidos por el algoritmo VNS es el realizado por T. Davidović y T. Crainic [108] donde proponen varias estrategias de paralelización, utilizando una arquitectura multiprocesador, logrando mejorar tanto la calidad de los resultados como los tiempos totales de ejecución de los algoritmos en comparación a un VNS sin técnicas de paralelización aplicadas. Otro ejemplo es el trabajo de Decia et al. [109] cuyo principal el objetivo es el de mejorar los tiempos de ejecución de la metaheurísticas VNS. En dicho trabajo se estudian optimizaciones, tanto en CPU como en GPU, que aprovechan los algoritmos de multiplicación de bloques en matrices. Estas son aplicadas a la metaheurísticas VNS con el objetivo de acelerar la búsqueda de soluciones en ILP. En dicho trabajo se reduce el tiempo necesario para realizar la evaluación del algoritmo VNS cuando el producto matriz-matriz es usado en lugar de realizar la multiplicación matriz-vector múltiples

veces implementando tres tipos de variantes que juntan varias opeaciones en una única operación de multiplicación matriz-matriz. El proceso de optimización toma ventaja del hecho de que realizar un producto matriz-matriz mejora los tiempos de ejecución debido a un mejor uso de la memoria cache del sistema pese a que realizar múltiples productos matriz-vector resulte en el mismo orden de operaciones [110].

La optimización multimodal es el conjunto de técnicas enfocadas en la búsqueda de múltiples óptimos globales y locales en oposición a la búsqueda de un único óptimo global. Su motivación se basa en que el conocimiento de varios óptimos permite el cambio entre los mismos, en caso de ser necesario, logrando mantener la performance general del sistema. Esto resulta de utilidad cuando el costo o restricciones de obtener el óptimo global lo hacen inviable. La aplicación de técnicas de algoritmos evolutivos es capaz de obtener múltiples soluciones mediante la simulación de una población en oposición a los métodos convencionales que requieren de reiniciar la búsqueda o realizar múltiples ejecuciones. Múltiples técnicas de optimización evolutiva, conocidas como técnicas de "*niching*"(técnicas de nicho), han sido desarrolladas en las últimas décadas para la localización de óptimos locales y globales. Dichas técnicas pueden ser incorporadas a algoritmos evolutivos para lograr mantener múltiples subpoblaciones dentro de una población objetivo, logrando así localizar múltiples óptimos globales o locales de manera simultánea. Para lograr realizar la búsqueda multimodal de soluciones las técnicas de *niching* conducen la búsqueda a distintas áreas en el espacio de búsqueda con el objetivo de que estas converjan a distintos picos simultáneamente, logrando de esta manera mantener diversas subpoblaciones dentro de la población general [111]. Las técnicas de *niching* pueden ser identificadas según las siguientes cuatro categorías:

1. Secuencial: aquellos que localizan los nichos de manera iterativa
2. Paralelo: aquellos métodos que localizan todos los nichos de forma paralela
3. Cuasi secuencial: aquellos que localizan los nichos de manera secuencial y realizan la búsqueda de nuevos nichos de forma paralela mientras que los actuales son mantenidos
4. Jerárquico: es una versión hibrida entre las técnicas secuencial y paralela diseñada para sobreponer las limitantes de la primera

La técnica de *clearing*, introducida por Petrowski [112], se basa en el concepto de la competencia por la obtención de recursos limitados de un entorno por una subpoblación dada, eliminando así dentro de cada nicho los individuos menos aptos y manteniendo la diversidad.

En esta tesis se estudian técnicas de ejecución paralela con el objetivo de mejorar las soluciones obtenidas y de obtener mejores tiempos de ejecución, enfocadas principalmente a aprovechar la multiplicación matriz-matriz resultante de agrupar varios vectores de emisión E en una matriz de emisiones $\mathbf{E}$.

2.7. Fotometría

El objetivo de la fotometría es medir la luz tomando en cuenta el sistema visual humano. Mientras que la radiometría se encarga de medir la luz en todas las regiones espectrales, incluyendo los espectros infrarrojo y ultravioleta, la fotometría solo mide en

la región espectral visible por el ojo humano (que está entorno de 360nm a 830nm). Por esto, la fotometría es la ciencia esencial para la evaluación de fuentes de luz y objetos utilizados para la iluminación, señalización, pantallas, y otras aplicaciones donde la luz está destinada a ser vista por los seres humanos. En esta tesis es de importancia el estudio de la forma en que las luminarias emiten luz en su entorno, siendo los archivos fotométricos los contenedores de la información de la emisión de la luz de cada luminaria.

2.7.1. Archivos fotométricos

Los archivos fotométricos son archivos, definidos en texto plano, que contienen las características fotométricas de una luminaria [113]. Los diagramas fotométricos polares o curvas polares, son la forma estándar de visualizar la distribución de luz de un archivo fotométrico. Estas características son el flujo luminoso, cuánta luz genera, la distribución de intensidad luminosa, en qué dirección emite luz y con qué intensidad y la potencia consumida. El flujo luminoso total es la cantidad de luz que es emitida por una fuente de luz en el espectro perceptible por el ojo humano y es medida en lúmenes (lm). La intensidad luminosa es la cantidad de lúmenes que es emitida en una dirección dada, por la luminaria, por ángulo sólido, y es medida en candelas (cd), equivalente a lúmenes por estereorradián. Un archivo fotométrico contiene la intensidad luminosa para un conjunto discreto de direcciones. Una de las principales instituciones encargadas de establecer el estándar para el formato de los archivos fotométricos es Illuminating Engineering Society of North America la cual define el formato LM-63-02 IESNA [113] que es el estándar utilizado en Norte América. Existe también el formato EULUMDAT estándar en Europa. Esta tesis utiliza el formato LM-63-02 IESNA ya que existe software libre que permite la transformación del formato EULUMDAT a este [114]. El objetivo principal de estos archivos es proveer la información de las luminarias a profesionales dedicados al diseño de la iluminación, existiendo herramientas como Dialux [13] que leen dichos archivos y proveen, en su software, herramientas útiles para la medición y diseño de la iluminación.

En la actualidad cada fabricante de luminarias provee una base de datos de archivos fotométricos que contiene la información fotométrica de las luminarias que fabrica. Las bases de datos utilizadas en los experimentos teóricos en esta tesis fueron descargadas de los fabricantes: Philips [115] y Cree [116] de los cuales se utilizan las bases de datos enteras sin pre-procesar. En la Figura 2.11 se pueden ver tres ejemplos de curvas polares y en la Figura 2.12 una representación en tres dimensiones de una curva polar. Las curvas polares son la representación gráfica de la intensidad luminosa de la luminaria.

En esta tesis los archivos fotométricos son utilizados para extraer la información (la intensidad luminosa entre otros datos) que es necesaria para realizar el cálculo de la emisión de luz de cada luminaria en la escena, y que es luego utilizada como la emisión en el método de radiosidad. Los datos de los archivos fotométricos son previamente transformados para representar el flujo luminoso que es recibido en cada parche de la escena.

2.7.2. Optimización de configuraciones de luminarias

Varias técnicas han sido desarrolladas con el objetivo de optimizar la posición de las luminarias en una escena, con el objetivo de alcanzar cierto conjunto de LIs. Shariful et al. [16] formulan un método para posicionar dos luminarias fijas en dos conjuntos de posiciones disjuntos. Los archivos fotométricos que utilizan para cada luminaria son

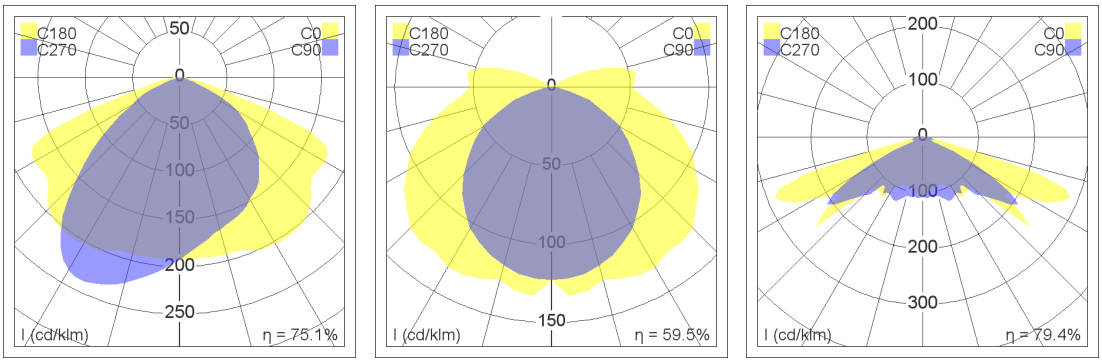


Figura 2.11: Ejemplos de curvas polares que representan gráficamente la intensidad luminosa de tres luminarias distintas. Las curvas polares muestran la intensidad luminosa en los planos construidos por los planos: C0 más C180 y C90 más C270.

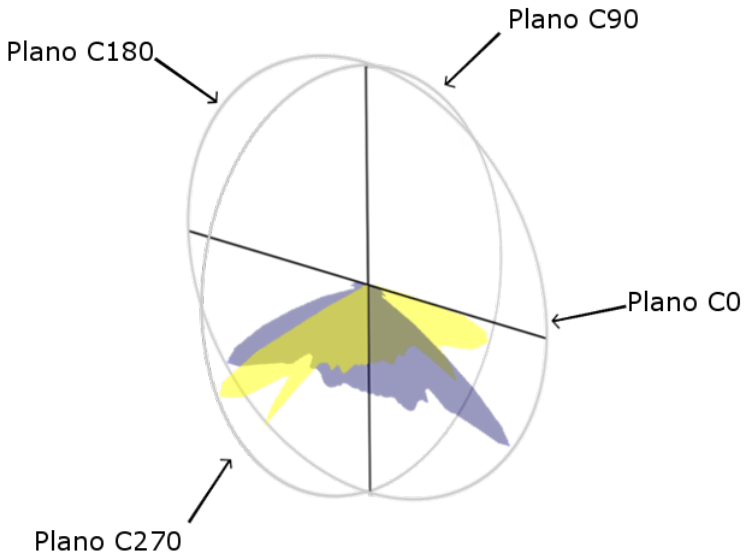


Figura 2.12: Representación en tres dimensiones de una curva polar. Como se puede notar la información de los planos C0 y C180 se junta para formar un único plano que contiene la distribución de la intensidad luminosa en cada dirección del nuevo plano. Lo mismo sucede para los planos C90 y C270. Los archivos fotométricos contienen la información necesaria para construir un conjunto discreto de planos que en conjunto tienen la información, discretizada, de la emisión de la luminaria.

previamente seleccionados, y fijos, y utilizados como entrada para el cálculo de la iluminación resultante. Otro método, que utiliza un acercamiento diferente, es propuesto por Uygun et al. [17], donde optimizan las posiciones de las luminarias también considerando la información fotométrica de las mismas. En el trabajo de Schwarz y Wonka [18] un interesante método es presentado para el diseño de la iluminación de exteriores de edificios, donde optimizan la posición y la rotación tomando en cuenta también la información fotométrica de las luminarias utilizadas, pero consideran únicamente la iluminación directa (principal componente de iluminación) y solo un número reducido de tipos de luminarias es considerado.

En esta tesis las técnicas propuestas resuelven el ILP calculando la propagación de la

luz en la escena, que es emitida por las luminarias posicionadas en esta, usando el método de radiosidad. La emisión de las luminarias es simulada utilizando la información de archivos fotométricos. Las técnicas propuestas están parcialmente basadas en el trabajo realizado por Fernández y Besuievsky [96], donde el método LRR es utilizado para resolver el ILP.

Capítulo 3

Optimización de configuraciones de luminarias

3.1. Introducción

Los métodos presentados en esta tesis se basan en la ecuación de la radiosidad, utilizándola para realizar el cálculo de la iluminación en la escena. Para poder utilizar la ecuación de radiosidad es necesario conocer los parches emisores de luz, aquellos que generan la iluminación en la escena. Los métodos propuestos utilizan como emisión, la primera reflexión de la luz emitida por una luminaria en los parches que su luz alcance. Es importante notar que el uso de varias luminarias es tan simple como acumular la cantidad de luz emitida por cada una de estas, y utilizar el resultado acumulado como entrada en la ecuación de radiosidad. El cálculo de la primera reflexión de una luminaria es realizado utilizando la información de la distribución de la intensidad luminosa de la luminaria. Esta información es descrita en los archivos fotométricos, que cada fabricante de luminarias provee para cada luminaria. Una vez computada la radiosidad, se utilizan los valores de iluminación obtenidos sobre la escena para determinar si se cumplen las LIs (Lighting Intentions) [12] y para evaluar qué tan buena es la solución (los valores a optimizar). La búsqueda del óptimo es implementada por medio de dos heurísticas, VNS y Algoritmos Evolutivos. La eficiencia de estas técnicas en la búsqueda de soluciones es evaluada en futuros capítulos.

3.2. Arquitectura de la solución

La arquitectura de la solución, mostrando las etapas del proceso, se puede ver en la Figura 3.1. El método tiene como entrada un modelo arquitectónico de un diseño interior preparado para el modelado de iluminación. También, recibe como entrada una base de datos con archivos fotométricos que contiene la información de cada una de las luminarias que se desean considerar en el proceso de optimización. Dichos archivos fotométricos pueden ser obtenidos de los fabricantes de luminarias.

El siguiente paso es la definición, por parte del diseñador, de los parámetros concernientes a las LIs, los objetivos y restricciones de iluminación, e.g., consumo de energía o cantidad de luz recibida en cierta superficie, además de los lugares donde las luminarias pueden ser ubicadas en la escena y otras variables a considerar en la optimización.

Luego, el proceso de pre-cómputo es ejecutado, donde se procesan tanto los archivos

fotométricos de las luminarias (para ordenarlas) como también las posibles ubicaciones en que estas pueden ser localizadas. El objetivo del pre-procesamiento de las ubicaciones es generar hemi-cubos que contienen los parches visibles desde cada ubicación (estos hemi-cubos serán descritos más adelante). Además, el proceso de pre-cómputo retorna las matrices de LRR, a partir de la geometría de la escena, que son utilizadas para realizar el cálculo, veloz, de la radiosidad.

Por último, se tiene la etapa de optimización. En esta se realiza la búsqueda de la configuración óptima. Una vez la configuración óptima es retornada por el proceso de optimización, es evaluada por el diseñador quien decide si cumple con lo deseado. De no acercarse a lo deseado, el diseñador puede opcionalmente modificar las variables de optimización o las LIs y volver a realizar los pasos a partir de la definición del problema. Otra posibilidad consiste en modificar las soluciones manualmente, para alcanzar una solución de compromiso entre el resultado del método y su experiencia, o de alguna otra herramienta de asistencia que utilice. Dependiendo de los cambios es posible que se deba realizar el pre-cómputo nuevamente, por ejemplo si se modifica la base de datos de archivos fotométricos o si se modifican las posiciones donde se pueden ubicar las luminarias.

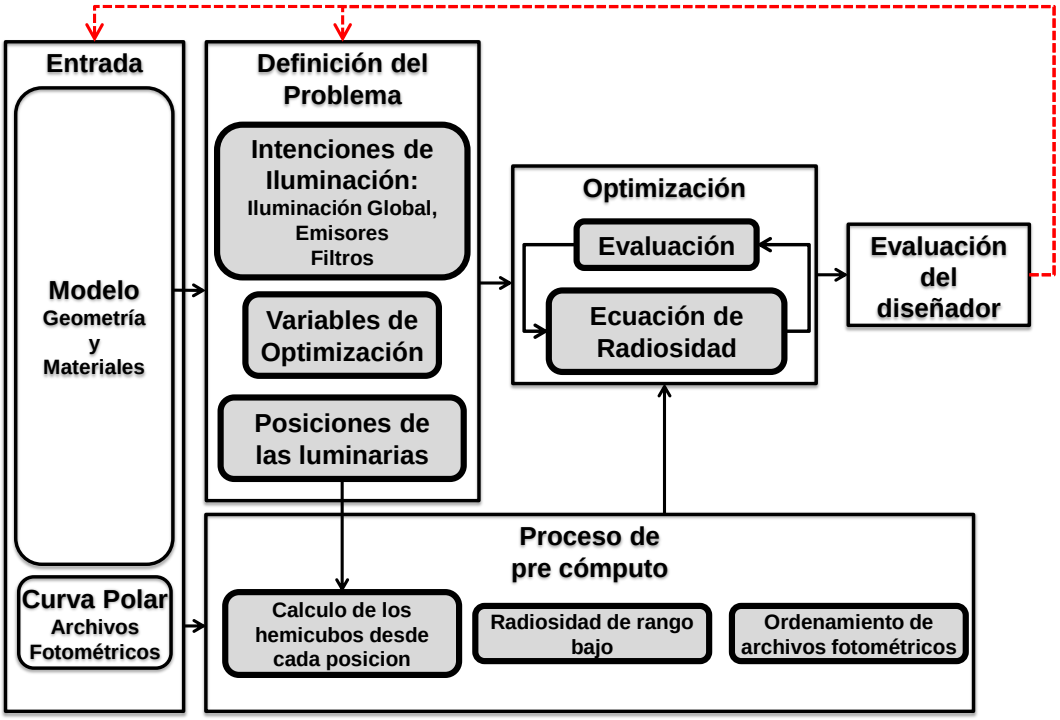


Figura 3.1: Tubería del sistema.

3.3. Pre-cómputo

El proceso de pre-cómputo se describe en la Figura 3.2. Se reciben como entrada la base de datos de archivos fotométricos y la geometría de la escena con la definición de las posiciones donde es posible ubicar las luminarias.

Las distribuciones de intensidad luminosa de cada luminaria en la base de datos son utilizadas para ordenarlas por similitud. Los pares de hemi-cubos de vista **H**, son creados para cada posible ubicación de una luminaria, utilizando la información de la geometría de la escena y la definición de las posiciones de las luminarias. Estos hemi-cubos son calculados utilizando el algoritmo de Z-buffer, donde cada píxel contiene el índice del parche, de la escena, que es visto a partir de este (ver Figura 3.3).

Como salida se tendrán, un vector con el índice de cada luminaria en la lista ordenada (I), un par de hemi-cubos de vista (**H**) para cada ubicación y una representación compacta de la escena es calculada por medio del método LRR (Sección 2.3.2).

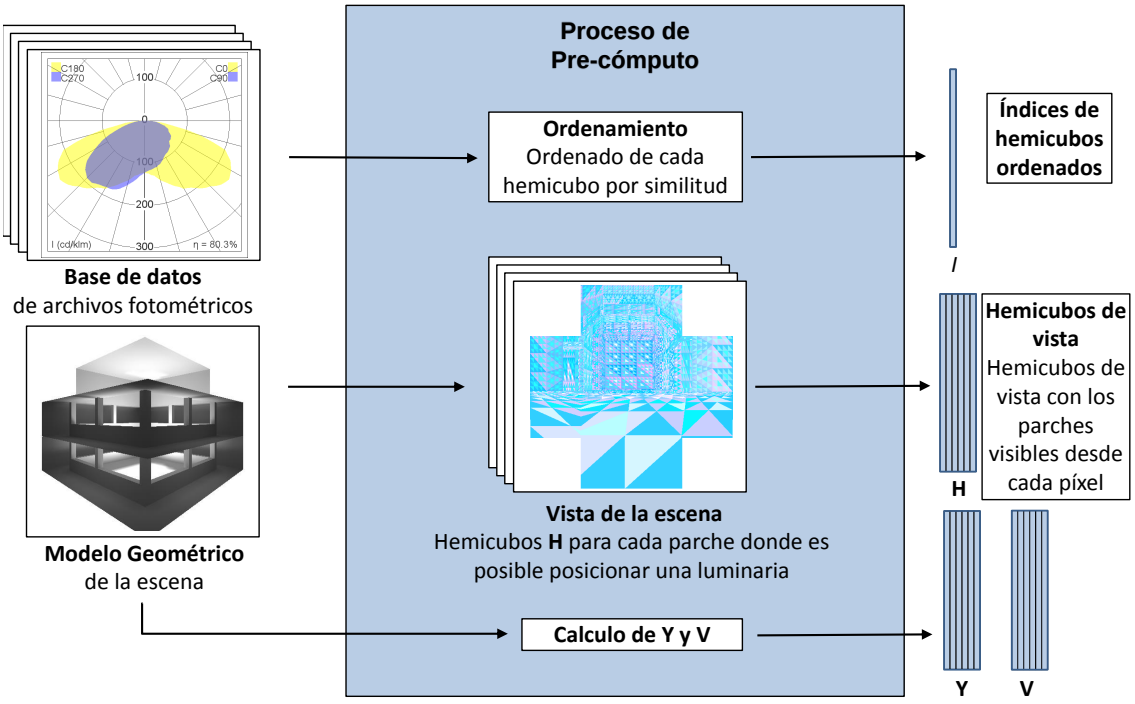


Figura 3.2: Proceso de pre-cómputo. Los hemi-cubos **H** son generados, además de las matrices **Y** y **V** del método LRR. A su vez las luminarias son ordenadas según sus distribuciones de intensidad luminosa (I) de manera de tener cerca aquellas luminarias con distribuciones similares.

3.3.1. Generación de hemi-cubos de emisión

Dependiendo de si se utilizan rotaciones, el proceso de pre-cómputo puede además incluir la generación de los hemi-cubos que contienen la emisión de cada luminaria, **I** (calculados utilizando la información fotométrica de las luminarias). Es decir cuando no es necesario aplicar rotaciones, estos hemi-cubos son constantes y por lo tanto pueden ser calculados en el pre-cómputo. En cambio, cuando se aplican rotaciones, dichos hemi-cubos dependen directamente de las rotaciones aplicadas y, debido a que sería de gran costo mantener cada uno de los hemi-cubos correspondientes a cada rotación en memoria, este cálculo es realizado en la etapa de optimización (ver Figuras 3.7(e) a 3.7(h)).

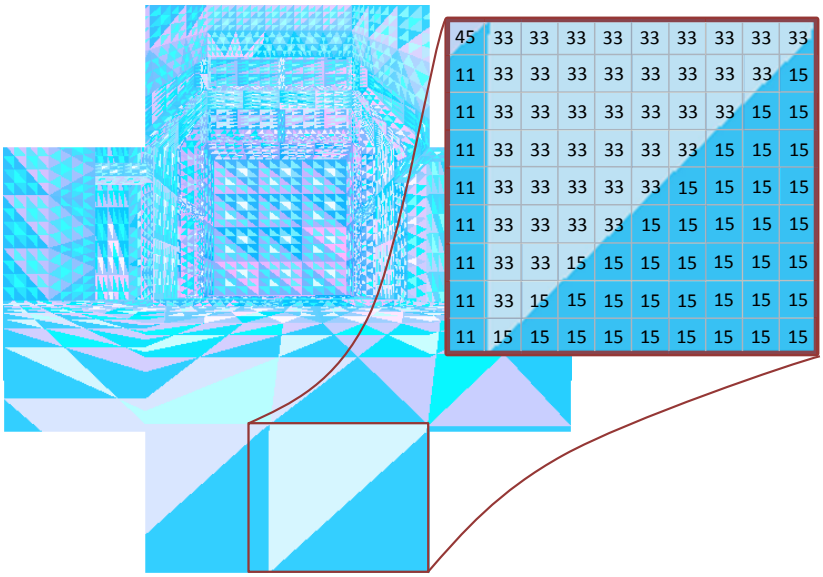


Figura 3.3: Un hemi-cubo **H** que contiene los parches de la escena que pueden ser vistos desde una luminaria. Cada píxel posee un número que indica el índice del parche visto a través de él. En la esquina superior derecha se muestra el índice asociado a cada píxel. El parche que es visto desde cada píxel del hemi-cubo **H** es determinado por medio del algoritmo Z-buffer.

3.3.2. Ordenamiento de luminarias

Para poder utilizar la metaheurística VNS, es necesario tener una forma de agrupar por similitud las variables involucradas en la optimización (crear los vecindarios). Es decir, dado un valor para una variable es necesario saber cuáles son sus valores vecinos o cercanos.

Cada luminaria en la configuración es representada por medio de un conjunto de variables. Dicho conjunto está compuesto por tres variables que determinan la rotación (cada variable representando un ángulo), dos variables que determinan su posición en el espacio y una variable con el identificador de la luminaria utilizada.

En el caso de las variables que definen las rotaciones, la agrupación en vecindarios es directa ya que las rotaciones son números, y por lo tanto rotaciones con valores cercanos (ángulos cercanos) son agrupadas en el mismo vecindario. En el caso de los hemi-cubos **H**, es posible construir los vecindarios usando sus posiciones en la escena, es decir, su ubicación en la escena 3D determina qué tan cercana es una ubicación de la otra, resultando trivial la creación de vecindarios. Sin embargo para el caso las distribuciones de intensidad luminosa representados por los hemi-cubos **I**, no existe una forma simple de agruparlos por similitud, o sea de construir los vecindarios necesarios para el VNS. Esto es debido a que los archivos fotométricos contienen la información de la emisión de la luminaria en cada ángulo, lo cual es representable como una matriz, y no todos los archivos fotométricos coinciden en los ángulos utilizados para su medición. En este caso utilizamos un algoritmo (Algoritmo 2) que ordena las luminarias utilizando la información de sus distribuciones de intensidad luminosa (obtenidas a partir de los archivos fotométricos y previamente interpolada calculando para cada luminaria su emisión en un conjunto idéntico de ángulos) y la distancia euclídea (o la norma de Frobenius debido a que son matrices).

Algoritmo 2 Ordenamiento de hemi-cubos de luminarias

Require: **D** {**D** es una matriz de distancias de dimensión n×n}
1: $I \leftarrow$ [1:n]
2: **for** k=1:n-1 **do**
3: $\mathbf{subD}_k \leftarrow \mathbf{D}(1{:}k, k{+}1{:}n)$
4: $(min, i, j) \leftarrow minimum(\mathbf{subD_k})$
5: **if** $min < \delta$ **then**
6: $\mathbf{D} \leftarrow moveCol(\mathbf{D}, j, i)$ % mueve columna j a la posición siguiente a i
7: $\mathbf{D} \leftarrow moveRow(\mathbf{D}, j, i)$ % mueve fila j a la posición siguiente a i
8: $I \leftarrow move(I, j, i)$ % mueve el índice j a la posición siguiente a i
9: **else**
10: $\mathbf{D} \leftarrow moveCol(\mathbf{D}, j, k)$ % mueve columna j al final de la lista ordenada
11: $\mathbf{D} \leftarrow moveRow(\mathbf{D}, j, k)$ % mueve fila j al final de la lista ordenada
12: $I \leftarrow move(I, j, k)$ % mueve índice j a la posición final
13: **end if**
14: **end for**
15: **return** I

El Algoritmo 2 recibe como entrada una matriz **D**, donde $\mathbf{D}(i,j)=||\mathbf{I}_i-\mathbf{I}_j||_{Fr}$ es la distancia euclídea entre los hemi-cubos **I** de las i-ésima y j-ésima luminarias. En el paso k-ésimo, $I(1{:}k)$ es una lista que contiene las luminarias que ya han sido ordenadas, y $I(k{+}1{:}n)$ contiene el resto de luminarias que falta ordenar. En el k-ésimo paso se selecciona una luminaria j de forma que minimiza la distancia a alguna de las luminarias de la lista ordenada. O sea la luminaria j es seleccionada si esta posee el menor valor en la matriz $\mathbf{subD}_k$. En la línea 3 se construye la matriz $\mathbf{subD}_k$ (ver Figura 3.4). Esta contiene las distancias entre las luminarias de la lista ordenada (1 a k) y las luminarias aún no ordenadas ($k+1$ a n). Se selecciona la mínima distancia de $\mathbf{subD}_k$ (línea 4 y Figura 3.4), que se encuentra en la fila i columna j. Esto indica que la luminaria no ordenada j, es la más cercana a las luminarias ya ordenadas, en particular a i. Si la distancia es menor a cierto umbral δ (predefinido como variable de entrada en el algoritmo de ordenamiento), entonces la luminaria j es ubicada en la lista ordenada entre las luminarias i e $i{+}1$ (líneas 6 a 8 y Figura 3.5(a)). Si la distancia es mayor o igual a δ, j es ubicado al final de la lista ordenada (líneas 10 a 12 y Figura 3.5(b)). El algoritmo comienza incluyendo la primera luminaria en la lista ordenada, y en cada paso es agregada a la lista una nueva luminaria, tomada del conjunto desordenado. Como resultado se tiene que todas las luminarias están en la lista ordenada y se retorna un vector que contiene los índices, ordenados, de las luminarias en la lista ordenada (línea 15). En dicho vector, la posición i contiene el índice de la i-ésima luminaria en la lista ordenada.

Un ejemplo con el resultado del ordenamiento de la base de datos de luminarias se puede ver en la Figura 3.6. En esta se observa que existe una mayor semejanza entre las columnas consecutivas en la base de datos ordenada. Cada columna contiene un hemi-cubo **I** de dimensión $n \times n$, pero ordenado en un vector de dimensión $n^2 \times 1$.

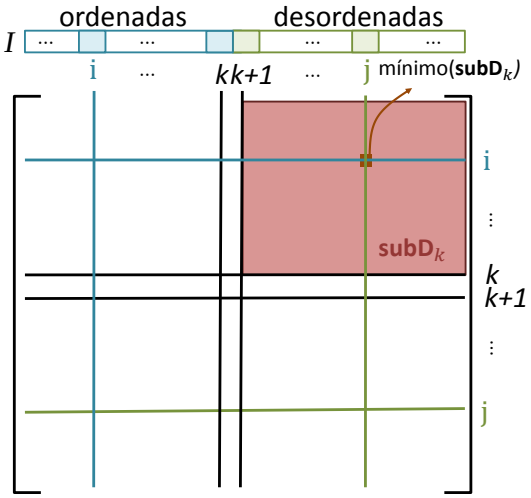
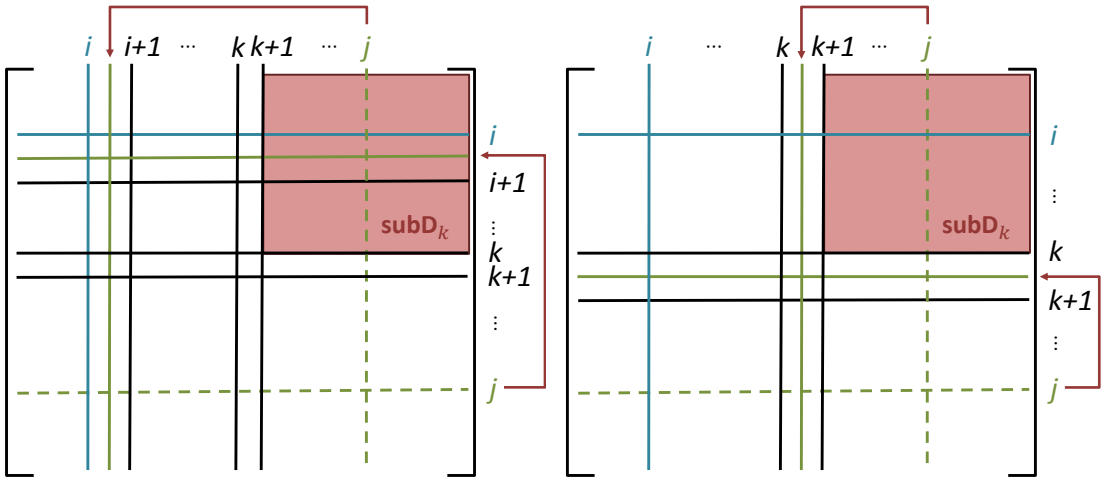
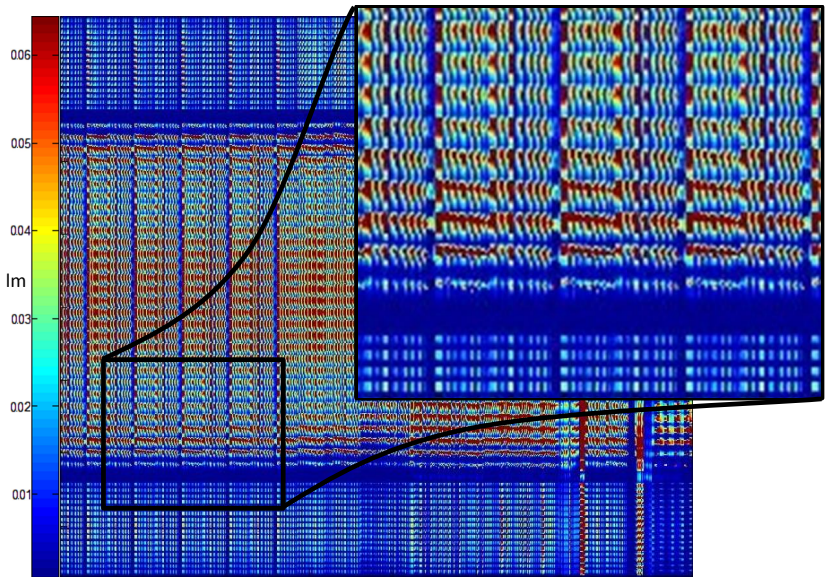


Figura 3.4: La matriz $\mathbf{subD}_k$ contiene las distancias euclidianas entre las luminarias que están en el conjunto ordenado a aquellas en el conjunto desordenado. En $\mathbf{subD}_k(i, j)$ se encuentra la mínima distancia. En cada paso se selecciona aquella distancia que es mínima.

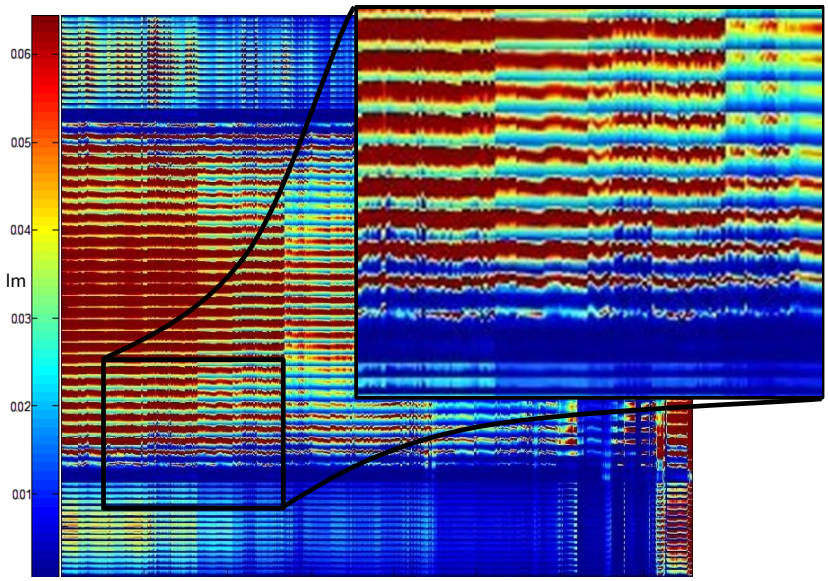


(a) $min(\mathbf{subD}_k) < \delta$ (b) $min(\mathbf{subD}_k) \geq \delta$

Figura 3.5: Etapas del ordenamiento. Se reubican la fila y columna j.



(a) Base de datos desordenada



(b) Base de datos ordenada

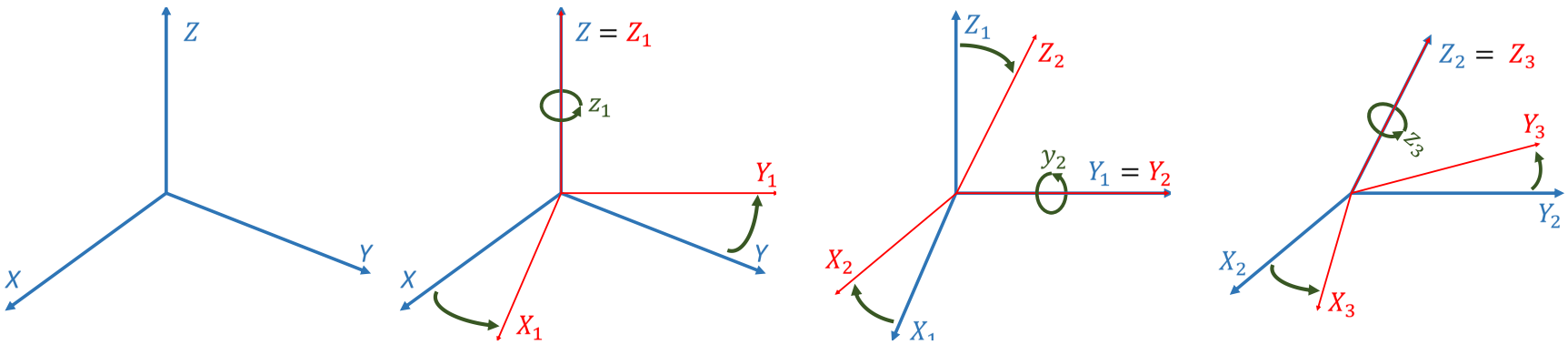
Figura 3.6: Gráfico de las base de datos que contiene los hemi-cubos de las luminarias. En la imagen cada columna representa un hemi-cubo (reordenado como un vector).

3.4. Optimización

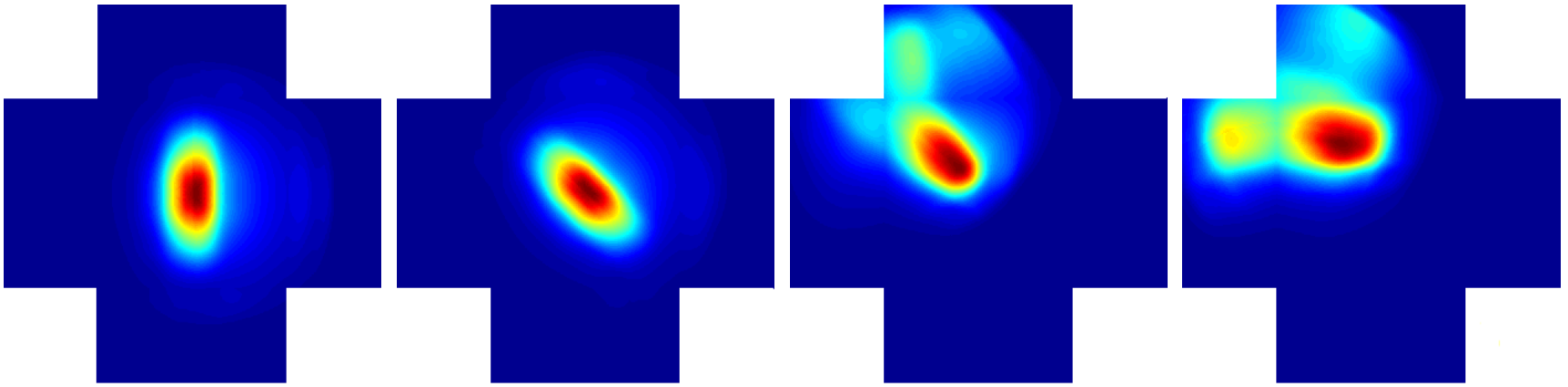
Uno de los algoritmos con que se realiza la optimización está basado en la metaheurística VNS. En este se realizan exploraciones sucesivas de los vecindarios seleccionando representantes aleatorios y evaluándolos con el objetivo de encontrar una solución mejor a las obtenidas en iteraciones previas. Las variables que pueden ser consideradas por VNS son: la posición, la rotación y la distribución de intensidades luminosas de cada luminaria. Las distribuciones de intensidades luminosas de las luminarias son agrupadas por cercanía utilizando el ordenamiento obtenido por medio del Algoritmo 2. Los vecindarios para las demás variables pueden ser construidos de forma directa. En la Sección 3.5 se describe el método utlizado para aplicar rotaciónes a los hemi-cubos **I**. Luego en la Sección 3.6 se detalla cómo se realiza el cálculo de la emitancia luminosa directa. En la Sección 3.7 se plantean las ecuaciones de radiosidad, mostrando cómo se utiliza la emitancia luminosa directa acumulada en su cálculo. A continuación, en la Sección 3.8, se detallan las heurísticas utilizadas en el proceso de optimización para realizar la búsqueda de óptimos. Luego en la Sección 3.9 se describe el proceso de evaluación paralela. Por último, en la Sección 3.10 se brindan detalles de la implementación como los lenguajes utilizados y el uso de la luminaria vacía.

3.5. Rotaciones

El método presentado utiliza fuentes de luces direccionales, fuentes de luz para las cuales su emisión varía según la dirección que sea considerada, y es por esto que es necesario optimizar cómo estas son orientadas en la configuración. Cuando se aplican rotaciones se utilizan ángulos de Euler con rotaciones intrínsecas. La elección de este tipo de rotaciones es debido a su aplicación intuitiva en este problema, pero podría aplicarse otro tipo de rotación, siempre y cuando no resulte demasiado ineficiente de computar. La elección de los ejes utilizados para aplicar las rotaciones también fue elegido porque resulta intuitivo su uso en este problema, pero es también posible utilizar cualquier otro tipo de variante. Las rotaciones intrínsecas seleccionadas se determinan por tres ángulos z_1, y_2 and z_3, como se puede ver en la Figura 3.7, y son utilizadas para orientar las distribuciones de intensidades luminosas de las luminarias. Como se ve en la Figura 3.7, el proceso seguido para generar las distribuciones rotadas comienza tomando un sistema de ejes inicial. Un ejemplo de hemi-cubo de emisión sin aplicar ninguna rotación se muestra en la Figura 3.7(e). Luego la primera rotación por z_1 grados es realizada sobre el eje original Z produciendo los nuevos ejes $X_1Y_1Z_1$ y obteniendo a su vez la luminaria rotada $\mathbf{I}_l^z$ (ver Figuras 3.7(b) y 3.7(f)). Luego se aplica nuevamente otra rotación, pero esta vez utilizando el ángulo y_2 y sobre el eje Y_1 (ver Figuras 3.7(c) y 3.7(g)) obteniéndose $\mathbf{I}_l^{zy}$ con un nuevo conjunto de ejes $X_2Y_2Z_2$. Por último se aplica la última rotación, esta vez z_3 grados sobre el eje Z_2 y obteniéndose de esta manera $\mathbf{I}_l^{zyz}$ (ver Figuras 3.7(d) y 3.7(h)). Estas rotaciones son realizadas durante el proceso de optimización. Hacerlas en la etapa de pre-cómputo implica costos computacionales excesivos. La metaheurística VNS es utilizada para realizar el control de cómo los ángulos son seleccionados.



(a) Ejes originales. (b) Primera rotación, eje Z. (c) Segunda rotación, eje Y. (d) Ultima rotación, eje Z.



(e) $\mathbf{I}_l$, z_1=0, y_2=0, z_3=0. (f) $\mathbf{I}_l^{\tilde{z}}$, z_1=45, y_2=0, z_3=0. (g) $\mathbf{I}_l^{zy}$, z_1=45, y_2=45, z_3=0. (h) $\mathbf{I}_l^{zyz}$, z_1=45, y_2=45, z_3=45.

Figura 3.7: La primer fila presenta las rotaciones intrínsecas utilizadas para realizar la orientación de las distribuciones de intensidades luminosas. De izquierda hacia la derecha, se comienza con los ejes originales de un sistema de coordenadas y aplicando las rotaciones según los ángulos z_1, y_2 and z_3 respectivamente. De esta forma se obtienen los ejes de coordenadas $X_3Y_3Z_3$ y la orientación final para la distribución de intensidades luminosas. La segunda fila muestra cómo se ve afectado el hemi-cubo de emisión $\mathbf{I}$ para cada rotación y para la misma luminaria (notar que cada imagen muestra una rotación diferente).

3.6. Cálculo de emitancia luminosa directa

En esta sección se detallan los conceptos involucrados en los cálculos realizados para obtener la primera reflexión de la luz que es emitida por una luminaria en la escena. La técnica utilizada está basada en la técnica del hemi-cubo y utiliza la información contenida en los archivos fotométricos (previamente procesados). El resultado será la primera reflexión de la luz en la escena que será usada como entrada en la ecuación de radiosidad para calcular la iluminación final en la escena. Cabe destacar que el uso de múltiples luminarias es sencillo, debiéndose utilizar como entrada en la ecuación de radiosidad, la suma de todas las primeras reflexiones de todas las luminarias consideradas. La Ecuación (3.1) muestra cómo se realiza el cálculo de la emitancia luminosa directa $E_{i,l}^{zyz}(p)$ para cada parche p en la escena iluminada por la luminaria l que está posicionada en el parche i.

$$E_{i,l}^{zyz}(p) = \frac{R(p)}{A(p)} \sum_{\substack{\{(u,v):\\ \mathbf{H}_i(u,v)=p\}}} \mathbf{\Delta\Omega}(u,v)\mathbf{I}_l^{zyz}(u,v) \tag{3.1}$$

donde p es un parche dado en la escena, $R(p)$ su reflectividad difusa, $A(p)$ su área, $\mathbf{H}_i$ es un hemi-cubo de vista de la escena centrado en el parche i (ver Figura 3.3), cada uno de los elementos (u,v) de la matriz $\mathbf{\Delta\Omega}$ contiene el ángulo sólido (en sr) del píxel (u,v) (ver Figura (3.8)). $\mathbf{I}_l^{zyz}$ es una matriz donde cada elemento (u,v) contiene la intensidad luminosa medida en candelas que pasa a través del píxel (u,v) por la luminaria l. La matriz $\mathbf{I}_l^{zyz}(u,v)$ es construida utilizando la información contenida en el archivo fotométrico correspondiente a la luminaria l, i.e., usando su distribución de intensidades luminosas, asignando el valor que corresponde a la emisión en candelas (lm/sr) de la luminaria a cada píxel en el hemi-cubo, y considerando la rotación que se aplicó a la luminaria.

Cada sumando $\mathbf{\Delta\Omega}(u,v)\mathbf{I}_l^{zyz}(u,v)$ en la Ecuación (3.1) determina el flujo luminoso, i.e., la cantidad de lúmenes, que pasan a través del píxel (u,v), y la sumatoria determina el flujo luminoso incidente (lm) en el parche p.

Por otro lado $\mathbf{\Delta\Omega}(u,v)$ es determinado por la Ecuación (3.2):

$$\mathbf{\Delta\Omega}(u,v) = \frac{4\pi\Delta A}{A_s}\ \hat{t}\cdot\hat{n} \tag{3.2}$$

En dicha ecuación, ΔA es el área del píxel (u,v), A_s es el área de la esfera de radio r $(4\pi r^2)$ que está centrada en el punto o (ver Figura 3.8), $\hat{n}$ es el vector unitario normal al píxel (u,v) y $\hat{t}$ es el vector unitario centrado en el píxel (u,v) que tiene dirección radial (su dirección es desde el centro del píxel al punto o). El producto escalar $\hat{t}\cdot\hat{n}$ determina el coseno del ángulo α. Por lo tanto la Ecuación (3.2) puede ser expresada como sigue:

$$\mathbf{\Delta\Omega}(u,v) = \frac{4\pi\Delta A}{4\pi(x^2+y^2+z^2)}\frac{1}{\sqrt{x^2+y^2+z^2}} \tag{3.3}$$

debido a que $r^2 = x^2+y^2+z^2$, y $\hat{t}\cdot\hat{n} = \dfrac{1}{\sqrt{x^2+y^2+z^2}}$

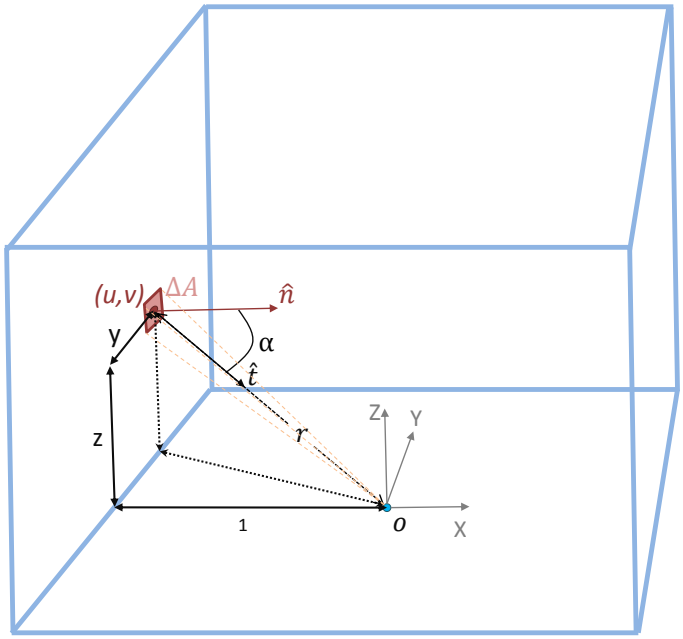


Figura 3.8: Bosquejo de ángulos y relaciones necesarias para realizar el cálculo del ángulo solido ($\Delta\Omega$) de un píxel (u,v) con respecto al centro del hemi-cubo (o).

lo que resulta en la Ecuación (3.4).

$$\mathbf{\Delta\Omega}(u,v) = \frac{\Delta A}{(x^2+y^2+z^2)^{\frac{3}{2}}} \tag{3.4}$$

La suma de todos los $\mathbf{\Delta\Omega}(u,v)$ es igual a 2π, o sea que cubre la hemi-esfera en su totalidad. Finalmente, dado que $E_{i,l}^{zyz}$ es la emitancia luminosa directa (medida en lux), entonces el flujo luminoso incidente en el parche p es dividido por $A(p)$ (determinando así la iluminancia directa) y multiplicado por la reflectividad $R(p)$.

3.7. Cálculo de la radiosidad

El vector $E_{i,l}^{zyz}$ calculado en la Ecuación 3.1 es usado como entrada en la ecuación de radiosidad (Ecuación 2.5) como sigue:

$$\tilde{B} = E_{i,l}^{zyz} + \mathbf{Y}(\mathbf{V}^T E_{i,l}^{zyz}) \tag{3.5}$$

Esta fórmula nos permite realizar miles de evaluaciones de configuraciones en cuestión de minutos. Cuando una configuración contiene muchas luminarias, $E_{i,l}^{zyz}$ es reemplazado por la sumatoria de las emisiones de cada luminaria contenida en la configuración, dando como resultado la siguiente ecuación:

$$\tilde{B} = \hat{E} + \mathbf{Y}(\mathbf{V}^T \hat{E}), \quad donde \quad \hat{E} = \sum_i E_{i,l}^{zyz} \quad i \in \mathbf{U} \tag{3.6}$$

U representa el subconjunto de posiciones donde se está poniendo una luminaria (en la configuración dada) y en la sumatoria l es la luminaria que se ubica en la posición i.

3.8. Heurísticas de búsqueda

Existen muchas heurísticas que permiten realizar la búsqueda de candidatos dentro del espacio de búsqueda. En esta tesis analizamos dos tipos: el VNS y técnicas de algoritmos evolutivos.

3.8.1. VNS

En esta tesis planteamos y estudiamos el uso de la metaheurística VNS como algoritmo de búsqueda de soluciones al problema de optimización de configuraciones de luminarias. A continuación se detallan las características y mecanismos utilizados en el algoritmo implementado para controlar la metaheurística, mejorando su eficiencia y eficacia en la búsqueda del óptimo.

Tabu Search

La técnica de Tabu Search [117] fue utilizada para evitar que el método recalcule configuraciones que fueron previamente visitadas.

Probabilidad de cambio en una variable en la configuración de una luminaria

Para configurar una única luminaria en la escena el algoritmo de optimización utiliza seis variables: una para seleccionar la luminaria (contiene el índice de la luminaria en la base de datos ordenada), dos para seleccionar la posición y las restantes tres para seleccionar la rotación (tres ángulos z_1, y_2 y z_3) a ser aplicada. El algoritmo VNS establece la cantidad de variables que pueden ser cambiadas simultáneamente con el objetivo de buscar candidatos en el vecindario explorado. Debido a que la cardinalidad de cada una de estas variables es diferente, el algoritmo de optimización da mayor probabilidad a aquellas variables que tienen el dominio con cardinalidad más grande. La probabilidad de cambio de una variable es calculada comparando el tamaño de su dominio con respecto al tamaño del dominio de las demás variables.

Vecindarios

Los vecindarios son determinados por dos condiciones: el entorno en que se permite realizar la búsqueda alrededor de una cierta variable, y la cantidad de variables simultáneas en las que se permite realizar las búsquedas en cierto momento. Las búsquedas dentro de cada variable son aleatorias pero están controladas por dichas condiciones. Los valores que pueden tomar dichas condiciones son ajustables y fueron configuradas de la siguiente forma:

$$\begin{aligned} \textit{entorno de búsqueda} &= \{\textit{0.1 0.3 0.5 0.7 1.0}\} \\ \textit{cantidad de variables} &= \{\textit{1 3 5 .. v}\} \end{aligned} \tag{3.7}$$

donde v es la cantidad máxima de variables simultáneas en la configuración

Por lo tanto existen tantos vecindarios como el número de parejas que podemos armar con los valores de estas dos condiciones de búsqueda (su producto cartesiano).

Por ejemplo si se tienen dos variables en la configuración (luminaria l y posición i) donde existen 100 luminarias y 50 posiciones tendremos que los vecindarios se arman como sigue:

$$\begin{aligned} \textit{cantidad de variables} &= \{1\ 2\ 3\} \\ \textit{donde tenemos los vecindarios} & \\ \{\{0.1,1\}\ \{0.1,2\}\ \{0.1,3\}\ \{0.3,1\}\ \{0.3,2\}\ \{0.3,3\}\ \{0.5,1\}\ \{0.5,2\} & \\ \{0.5,3\}\ \{0.7,1\}\ \{0.7,2\}\ \{0.7,3\}\ \{1.0,1\}\ \{1.0,2\}\ \{1.0,3\}\} & \end{aligned} \tag{3.8}$$

Por lo tanto si se está realizando la búsqueda en el vecindario $\{0.5,2\}$ se buscarán individuos aleatoriamente cambiando tanto su posición como su luminaria y donde la luminaria esté en un entorno de $[l-50,l+50]$ y posición $[i-25,i+25]$, o sea buscando alrededor de los valores que posee la mejor solución encontrada hasta el momento en ambas variables.

Selección de variables que determinan un vecindario

El orden en que se recorren dichos vecindarios no es aleatorio. Para hacer la recorrida se cambia la combinación de parejas de forma que en cada transición del vecindario se cambia el miembro de la pareja que no cambió en la transición anterior. Esto se hace recorriendo la matriz por sus diagonales, formada por el producto cartesiano de los valores de las condiciones, como se muestra en la Figura 3.9.

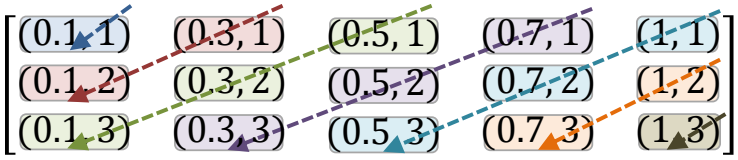


Figura 3.9: Navegación para el caso donde se tienen los tamaños de vecindarios $\{0.1\ 0.3\ 0.5\ 0.7\ 1\}$ y 3 variables en la solución. La navegación determina cómo se seleccionan las variables que determinan un vecindario.

En el ejemplo presentado en la Figura 3.9 los vecindarios se recorren en el siguiente orden: $\{0.1,1\}$, $\{0.3,1\}$, $\{0.1,2\}$, $\{0.5,1\}$, $\{0.3,2\}$, $\{0.1,3\}$, $\{0.7,1\}$, $\{0.5,2\}$, $\{0.3,3\}$, $\{1,1\}$, $\{0.7,2\}$, $\{0.5,3\}$, $\{1,2\}$, $\{0.7,3\}$, $\{1,3\}$.

Cambio de vecindario al encontrar una mejor solución

Al realizar la búsqueda dentro de un vecindario si se encuentra una solución mejor al óptimo actual, se deben recentrar los vecindarios a partir del nuevo óptimo. Sin embargo, en la variante del algoritmo implementado en esta tesis, al encontrar dicha condición se permite continuar la exploración del vecindario actual con el objetivo de no parar ante la primer mejora del valor a optimizar encontrada.

3.8.2. Algoritmos Evolutivos

En esta tesis planteamos el uso de un algoritmo evolutivo que resuelve el ILP presentado en la tesis por medio del solver GA [118]. Este algoritmo es utilizado en problemas

de optimización, y es basado en el proceso de selección natural simulando la evolución biológica. El algoritmo consiste en modificar repetidamente las poblaciones de individuos cruzándolos para generar nuevas generaciones intentando obtener mejores características en los individuos.

Los pasos para utilizar este tipo de algoritmos requiere decidir e implementar la forma en que las soluciones son representadas, el algoritmo de generación de la población inicial, la función de fitness que evaluará la aptitud de un individuo en la población y los operadores evolutivos (tipo de selección, recombinación y mutación).

Representación de las soluciones

En el problema planteado es necesario encontrar la combinación de ubicación y tipo de luminaria (emisión) para cada una de las luminarias que deben ser utilizadas, por lo tanto en la representación de una luminaria se debe contener la ubicación de la luminaria y su tipo. Las superficies donde se pueden ubicar las luminarias pueden ser solamente superficies planas y de forma rectangular. De esta manera, para ubicar la luminaria, se toman las coordenadas x e y que esta tendrá en la superficie objetivo (dominio de parches donde es posible ubicar las luminarias). Por otro lado para representar la luminaria se utiliza su índice en la base de datos. Por lo tanto la configuración de una luminaria es representada como sigue:

$$\begin{cases} lum = (x, y, l) \\ x \in 1..X, X \in \mathbb{N} \\ y \in 1..Y, Y \in \mathbb{N} \\ l \in 1..L, L \in \mathbb{N} \end{cases} \tag{3.9}$$

Donde X e Y representan la cantidad máxima de posiciones distintas según el eje x de la superficie objetivo e y respectivamente, y donde L representa la cantidad de luminarias existentes en la base de datos. En este problema es necesario poder calcular soluciones con múltiples luminarias y por lo tanto la solución es representada de la siguiente forma:

$$\begin{cases} sol = (x_1, y_1, l_1, ..., x_i, y_i, l_i, ..., x_n, y_n, l_n) \\ x_i \in 1..X, X \in \mathbb{N} \\ y_i \in 1..Y, Y \in \mathbb{N} \\ l_i \in 1..L, L \in \mathbb{N} \\ n \in \mathbb{N}, n>0 \end{cases} \tag{3.10}$$

Población inicial

La población inicial será generada tomando individuos de forma arbitraria.

Operadores evolutivos

En cuanto a los operadores evolutivos, se utilizará la selección por torneo con 2 individuos. Para la combinación se utiliza Cruzamiento de dos Puntos (2PX) y por último

para la mutación se realiza el cambio de un alelo por otro de forma aleatoria seleccionando dentro de los rangos de valores posibles para la variable (mínimos y máximos valores posibles) con una distribución uniforme.

Es importante destacar que cuando se utilizan métodos de algoritmos evolutivos para realizar la búsqueda de óptimos, no es necesario que la base de datos de luminarias esté ordenada, debido a que el algoritmo no hará uso de dicha información.

3.9. Cálculo de la radiosidad para múltiples candidatos

En esta tesis también incorporamos el uso del VNS con búsquedas paralelas. Esta técnica mejora el desempeño computacional del algoritmo dando mejores resultados en los tiempos totales que toma realizar las búsquedas de configuraciones óptimas. En esta mejora de performance, una matriz $\mathbf{E}$ que contiene la emisión de cada candidato ($\mathbf{E} = [E_1, ..., E_p]$, donde p es la cantidad de individuos evaluados en paralelo) es construida con el objetivo de realizar una evaluación paralela de todos los candidatos dentro de un mismo vecindario. Los resultados que se obtienen en la matriz $\tilde{\mathbf{B}}$ contienen cada vector $\tilde{B}$ de la ecuación descrita en la Sección 3.7, ver Ecuación (3.11).

$$\tilde{\mathbf{B}} = \mathbf{E} + \mathbf{Y}(\mathbf{V}^T\mathbf{E}) \tag{3.11}$$

En el Anexo se desarrollan algunos de los conceptos y algoritmos de paralelismo que pueden ser aplicados a la metaheurística VNS. En dicho Anexo se estudia el uso del paralelismo aplicado al VNS utilizando emisores lambertianos en lugar de la información de los archivos fotométricos de las luminarias.

3.10. Detalles sobre la implementación

La mayoría del código y herramientas implementadas fueron escritos en el lenguaje MATLAB. También se usó C++ y OpenGL para la visualización y cálculo de los hemicubos de vista o los renderizados de las emisiones.

Luminaria vacía Para no obligar al algoritmo a utilizar todas las luminarias pedidas en un problema, sino un número menor o igual (por ejemplo, si se quieren poner 3 luminarias, que el algoritmo pueda poner 2 o 1, si así se obtienen mejores resultados), introducimos el concepto de luminaria vacía. Esta luminaria sirve para relajar la cantidad de luminarias indicadas en el algoritmo. El algoritmo puede utilizar esta luminaria (que consume 0 watts y no produce ningún tipo de emisión) de forma que es posible conseguir configuraciones válidas con menor o igual cantidad de luminarias que las posibles en una configuración. Esta luminaria posee un hemi-cubo $\mathbf{I}_0$ donde cada píxel contiene el valor 0 y es evaluada al momento de ordenar las luminarias para conocer cuándo esta debe ser considerada en los vecindarios.

Capítulo 4

Análisis experimental

Esta sección presenta los experimentos realizados para evaluar las técnicas implementadas y los resultados obtenidos en estos.

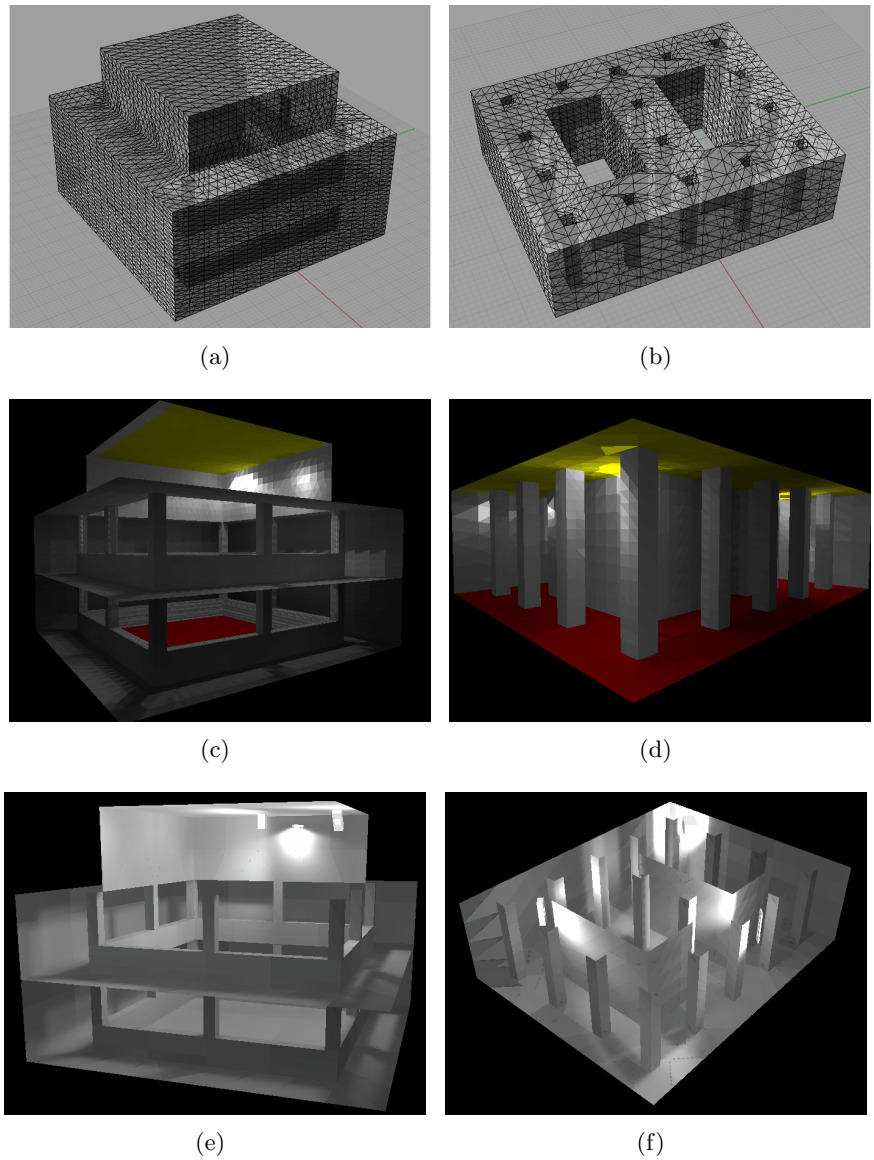
4.1. Introducción

Los experimentos realizados en esta tesis buscan analizar el cumplimiento de estándares de iluminación, considerando objetivos y restricciones realistas en el ámbito del diseño de iluminación. El primer experimento de la sección analiza la convergencia del método (Sección 4.2), luego se analiza la optimización de configuraciones con el objetivo de mejorar su eficiencia en consumo eléctrico (Sección 4.3). A continuación se presenta un experimento que busca maximizar la cantidad de luz recibida en una superficie dada (Sección 4.4), para luego evaluar con un nuevo experimento la búsqueda de uniformidad en la luz recibida en una superficie (Sección 4.5). Luego se estudia la aplicación de técnicas de evaluación paralela de configuraciones con el objetivo de mejorar la eficiencia computacional del algoritmo (Sección 4.6) y, con otro experimento diferente, qué tan efectivo es el algoritmo de ordenado (Sección 4.7). Luego se realizan dos experimentos para evaluar la performance del algoritmo (Sección 4.8), donde se estudia el efecto que tiene el aumento de la cantidad de luminarias que pueden ser utilizadas en la optimización (Sección 4.8.1), y por último cuánto afecta el tamaño del hemi-cubo al cálculo del pre-cómputo (Sección 4.8.2). En la Sección 4.9 se comparan las implementaciones del VNS y el algoritmo evolutivo para determinar cuál retorna los mejores resultados. Por último (Sección 4.10) se analiza la aplicación del algoritmo VNS en dos casos reales, donde se optimiza la configuración de luminarias en el edificio Palacio de los Tribunales (Poder Judicial).

Las tablas que se presentan en cada experimento muestran los resultados de aplicar las optimizaciones con y sin rotaciones, indicando con $\neg Rot$ cuando no se aplican rotaciones y Rot cuando sí son aplicadas. Las columnas *peor* y *mejor* contienen los resultados de las instancias de las pruebas de optimización que retornaron los peores y mejores resultados respectivamente. Todas las instancias de todos los experimentos fueron ejecutadas 20 veces usando una base de datos que contiene 1517 luminarias (1516 luminarias más la luminaria vacía) para los experimentos teóricos y 734 (incluida la luminaria vacía) en el caso de estudio del edificio del Palacio de los Tribunales (Sección 4.10). Las posibles rotaciones son determinadas por los ángulos z1 $\in$ [0, 45, 90, 135, 180, 225, 270, 315], y2 $\in$ [0, 45, 90, 135], z3 $\in$ [0, 90, 180, 270] lo que resulta en 128

rotaciones. Esta simplificación es debida a que en la práctica no se utilizan rotaciones de a grados, y el algoritmo puede ejecutar y converger de mejor manera si se considera un conjunto de rotaciones simple y reducido.

Cuando se utiliza la escena del patio (Figura 4.1(a)) es posible ubicar las luminarias en 49 posiciones distintas (distribuidas en el techo de la escena), mientras que en la escena con forma de ocho (Figura 4.1(b)) la cantidad de posiciones es 100 (también distribuidas en el techo).



(a) (b) (c) (d) (e) (f)

Figura 4.1: Escenas utilizadas en los experimentos teóricos. La primera columna muestra la escena del Patio, compuesta por 21824 parches. La segunda columna muestra la escena con forma de ocho, que consiste de 9088 parches. Las posibles posiciones donde se pueden ubicar las luminarias están coloreadas en amarillo, mientras que las superficies objetivo lo están en rojo (ver (c) y (d)). En (e) y (f) se muestran los resultados del experimento de la búsqueda de uniformidad (Sección. 4.5), renderizado por medio de Dialux.

Todas las simulaciones fueron realizadas en un PC configurado con un procesador

Intel i7 processor (3.4 GHz) de cuatro núcleos y 16 Gbytes de RAM.

4.2. Convergencia

El propósito de este experimento es comprobar la capacidad del algoritmo para converger a una solución previamente definida. Para realizar este experimento se toma como objetivo de optimización una radiosidad dada B_{Obj}, que es obtenida mediante el cálculo de una de las configuraciones de emisión posibles (definida por luminarias, ubicaciones y orientaciones). La función objetivo es aquella que minimiza la distancia entre la radiosidad generada para la configuración que está siendo evaluada (B) y dicho B_{Obj}. Esto se puede formular como sigue:

$$\text{mín} : \|B - B_{Obj}\|_2 \tag{4.1}$$

Los resultados de este experimento se pueden ver en la Tabla 4.1. La primera columna muestra el número de luminarias utilizadas en las instancias de optimización, luego se muestra la cantidad de veces que se encontró la solución exacta B_{Obj}, seguido por el número medio de iteraciones calculado a partir de la cantidad de iteraciones que tomó encontrar dicha solución o el máximo número de iteraciones realizadas (en caso de que el algoritmo no haya encontrado la solución exacta). Por último, las tres columnas restantes muestran la media de la distancia obtenida para la solución con respecto al objetivo, su error máximo (relativo), y la dimensión del dominio que es tomado en cuenta en la optimización, respectivamente. El máximo número de iteraciones permitido para este experimento es 100000, y la escena utilizada es el Patio (Fig. 4.1(a)).

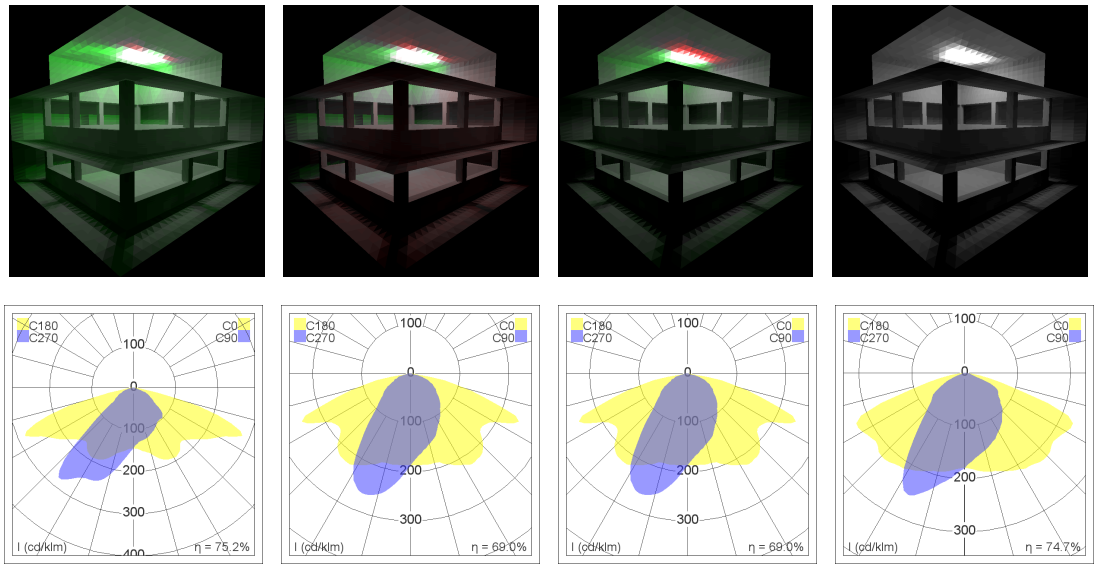
Tabla 4.1: Resultados de la optimización con el objetivo de minimizar la distancia a una configuracion determinada.

# lum	# exactas	$\mu(\#it)$	$\mu\left(\frac{\|B-B_T\|_2}{\|B_T\|_2}\right)$	máx $\left(\frac{\|B-B_T\|_2}{\|B_T\|_2}\right)$	dominio
1	20	6617	0	0	$9,5 \times 10^6$
2	1	91210	$1{,}1\times10^{-1}$	$2{,}3\times10^{-1}$	$9,0 \times 10^{13}$
3	0	100000	$1{,}2\times10^{-1}$	$2{,}9\times10^{-1}$	$8,6 \times 10^{20}$

Los resultados muestran que el algoritmo converge de buena forma cuando se consideran configuraciones con una única luminaria, retornando la solución exacta en todas las ejecuciones (realizando un promedio de 6617 iteraciones). Para dos luminarias, el dominio es casi 10^7 veces más grande, sin embargo el algoritmo es capaz de encontrar la solución exacta en 1 de las 20 instancias ejecutadas, y con errores relativos pequeños (las soluciones obtenidas no están alejadas de la solución objetivo) con la misma cantidad de iteraciones máxima. Configuraciones con mayor número de luminarias configuradas requerirán de un mayor número de iteraciones si se desea encontrar la solución exacta, debido al crecimiento exponencial del dominio de búsqueda. De todas formas la media y peor caso se mantienen acotados.

La Figura 4.2 muestra la evolución de las variables en la configuración para el experimento que considera una luminaria en la configuración.

Es importante destacar que el tamaño del dominio de búsqueda depende directamente de las variables consideradas en la optimización y los valores que estas pueden tomar. El número de configuraciones posibles para una luminaria dada se determina como el



(a) # 1 (b) # 157 (c) # 401 (d) # 516

Figura 4.2: Cada columna representa el estado de las variables en un paso de la optimización (de izquierda a derecha) para una ejecución que encontró el óptimo B_{Obj}. La primer fila muestra la iluminación en la escena con código de colores que indican la diferencia entre B y B_{Obj}, verde indica que $B > B_{Obj}$, rojo indica que $B < B_{Obj}$, y en otro caso el color original es utilizado (escala de grises). Luego en la segunda fila se muestra la curva polar de la luminaria seleccionada en el paso. La primera columna (de izquierda a derecha) muestra la configuración para la iteración inicial (que es seleccionada de forma aleatoria), y para la cual se tiene la luminaria rotada en los ángulos z_1=180, y_2=90 y z_3=270. La segunda columna muestra la configuración para la iteración número 157, rotada según z_1=180, y_2=90 y z_3=270. La tercer columna para la iteración número 401, rotada según z_1=180, y_2=135 y z_3=270. Y por último la cuarta columna para la última iteración (516), rotada según z_1=0, y_2=180 y z_3=90 donde se encontró la solución exacta.

número de luminarias que se consideran en la base de datos, multiplicado por la cantidad de posiciones en la que se puede ubicar una luminaria, multiplicado por la cantidad de rotaciones consideradas. Cuando hay múltiples luminarias el número anterior se lo eleva a la cantidad de luminarias consideradas en la configuración. La columna llamada dominio en la Tabla 4.1 muestra los diferentes tamaños de dominios de búsqueda para la escena del Patio.

4.3. Eficiencia en consumo eléctrico

El objetivo de este experimento es minimizar el consumo eléctrico de la configuración sujeto a que las iluminancias se encuentren acotadas entre 2 valores preestablecidos. El problema se puede expresar matemáticamente como sigue:

$$\text{mín} \sum_{i=1}^{n} P_i \ , \ i \in 1..n \ , \ s.t. : \begin{cases} min(I) \geq 50\text{lx} \\ max(I) \leq 250\text{lx} \end{cases} \tag{4.2}$$

donde n es la cantidad de luminarias que se utilizan en la configuración, P_i es la potencia (medida en watts) consumida por la luminaria i-ésima. I es el vector que contiene los valores de iluminancia para cada parche en la superficie objetivo.

En este experimento la superficie objetivo es una de las paredes en el segundo piso de la escena del Patio (Figura 4.1(a)) y el máximo número de iteraciones permitido es de 25000.

La Tabla 4.2 presenta los resultados de este experimento. En la columna *mejor* se destaca que orientar las luminarias retorna mejores resultados. Sin embargo la columna donde se encuentra la media y desviación estándar ($\mu \pm \sigma$) muestra que mejores resultados pueden ser conseguidos si no se utilizan rotaciones, al menos luego de 25000 iteraciones. Esto es debido al tamaño del dominio de búsqueda ($8,6 \times 10^{20}$ configuraciones), que hace necesaria la realización de una mayor cantidad de iteraciones para encontrar buenas configuraciones. De todas formas, las soluciones obtenidas son similares a las obtenidas sin rotaciones.

En la Tabla 4.2, la fila para $\#lum = 1$ no es presentada ya que no existen soluciones al problema definido en la Ecuación 4.2

Tabla 4.2: Resultados de la optimización para el experimento de eficiencia eléctrica. El objetivo es minimizar la cantidad de potencia eléctrica consumida por las luminarias en la configuración pero restringido a conseguir un rango mínimo y máximo de iluminancia en la superficie objetivo.

#lum	mejor (W)		peor (W)		$\mu \pm \sigma$ (W)	
	$\neg Rot$	Rot	$\neg Rot$	Rot	$\neg Rot$	Rot
2	424	**334**	**431**	442	426 ± 4	**418** ± 31
3	362	**345**	**431**	453	**381** ± 21	413 ± 38

4.4. Maximizado de iluminación

Este experimento consiste en maximizar la iluminación en una superficie objetivo dada. La superficie que se intenta maximizar en este caso es la pared que se encuentra en el segundo piso de la escena del Patio (Figura 4.1(a)) y como restricción se plantea que la suma de la potencia eléctrica consumida por las luminarias no supere los 100 watts. Además se establece en 25000 la cantidad máxima de iteraciones que el algoritmo puede realizar.

La Tabla 4.3 muestra los resultados para este experimento. Para cualquier número de luminarias, las soluciones con rotaciones generan mejores iluminaciones. Incluso en el peor caso los resultados son mejores cuando hay rotaciones.

Tabla 4.3: Resultados de optimización para el experimento de maximizar la cantidad de iluminación recibida en una superficie objetivo dada, sujeto a una cantidad máxima de consumo eléctrico (100 W) en la suma de las luminarias utilizadas en la configuración.

#lum	mejor (lx)		peor (lx)		$\mu \pm \sigma$ (lx)	
	$\neg Rot$	Rot	$\neg Rot$	Rot	$\neg Rot$	Rot
1	29	**38**	26	**30**	29 ± 1	**36** ± 2
2	40	**51**	28	**30**	35 ± 6	**38** ± 6
3	38	**52**	29	**35**	36 ± 2	**44** ± 5

4.5. Uniformidad de la luz en una superficie

El objetivo de alcanzar uniformidad en la luz producida en un diseño de iluminación es de suma importancia para alcanzar ambientes más atractivos y funcionales [119]. En esta sección evaluamos objetivos de uniformidad en la luz producida en la escena.

El coeficiente de varianza (CV $= \sigma/\mu$) [120] es una forma de medir que tan uniformemente se distribuye una magnitud (a menor CV la magnitud es más uniforme). Esto es la medida de dispersión normalizada. En este experimento medimos la uniformidad de la luz y tenemos como objetivo minimizar el CV para así buscar la solución con la mayor uniformidad posible. Por lo tanto, el objetivo es como sigue:

$$\text{mín} : \frac{\sigma(\tilde{B})}{\mu(\tilde{B})} = \text{mín} : \text{CV} \tag{4.3}$$

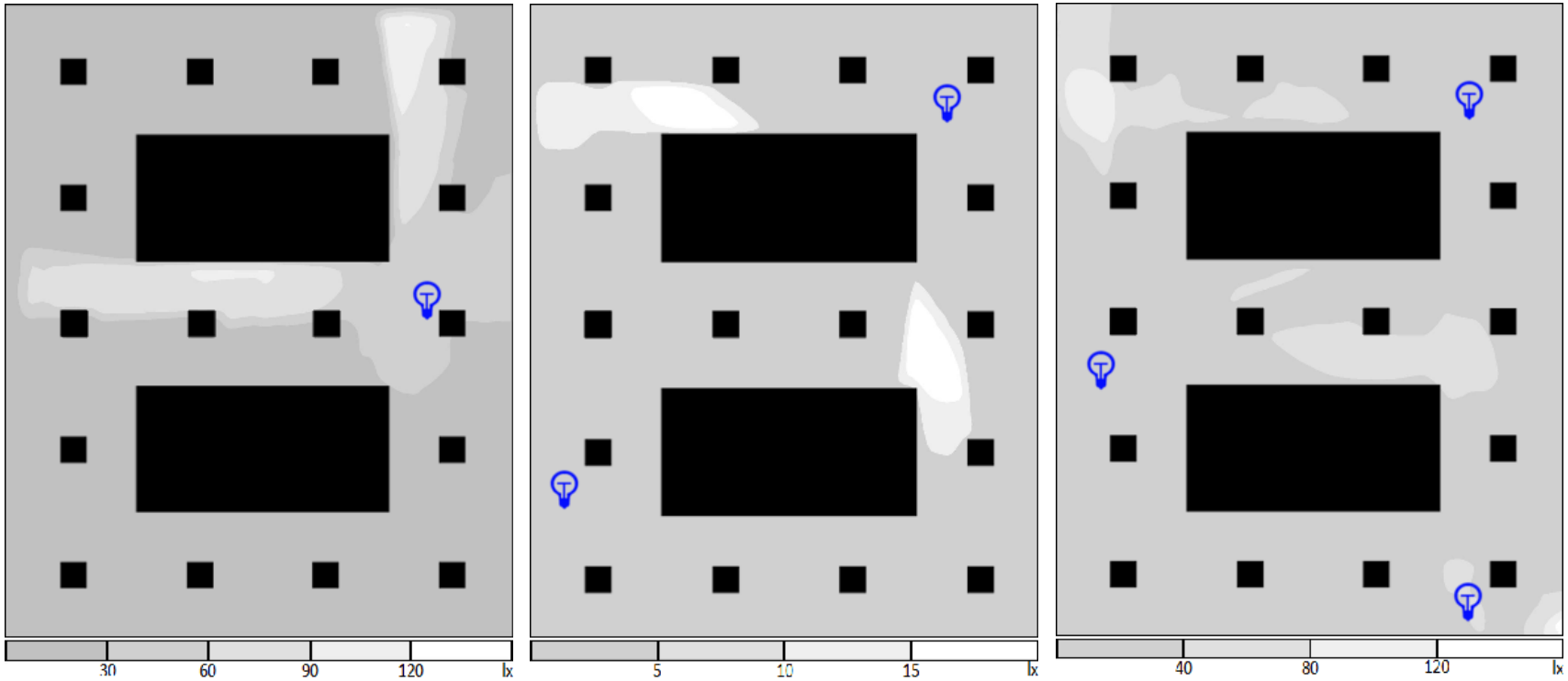
La máxima cantidad de iteraciones permitida en este experimento es de 25000 y las escenas utilizadas son el Patio y el Ocho (Figuras 4.1(a) y 4.1(b)). Debido a la complejidad de la geometría de la escena con forma de ocho, esta resulta más interesante para ser evaluada en este experimento. La superficie objetivo es en ambos casos el piso de cada escena.

La Tabla 4.4 muestra los resultados numéricos obtenidos para las optimizaciones. Como se puede ver en dicha tabla, la uniformidad de la luz obtenida en la escena mejora cuando se permite al algoritmo utilizar más luminarias por configuración. Esto es debido a que a medida que se da mayor libertad en la cantidad de luminarias a utilizar, estas se pueden ubicar y rotar de forma que la luz emitida por las mismas se complementen entre sí. Es importante destacar nuevamente que el uso de la luminaria vacía permite que una configuración tenga menos luminarias que las máximas indicadas.

Tabla 4.4: Resultados de la optimización para el experimento de uniformidad en la luz donde se muestra el coeficiente de varianza obtenido para cada escena y cantidad de luminarias evaluados. La columna *Patio* contiene los resultados para el Patio y la columna *Ocho* los resultados para la escena con forma de ocho.

#lum	mejor CV ($\times 10^{-2}$)		peor CV ($\times 10^{-2}$)		$\mu(\text{CV}) \pm \sigma(\text{CV})$ ($\times 10^{-2}$)	
	Patio	*Ocho*	*Patio*	*Ocho*	*Patio*	*Ocho*
1	4,0	90	6,0	96	$4,4 \pm 0,8$	93 ± 3
2	3,1	42	5,2	52	$4,1 \pm 0,6$	46 ± 3
3	2,6	21	4,8	33	$3,7 \pm 0,6$	26 ± 3

En la Figura 4.3 se pueden ver los intervalos de iluminación (renderizadas utilizando Dialux) calculados para la superficie objetivo. Cuando se utiliza una mayor cantidad de luminarias, la emisión de estas crea una iluminación con una mejor distribución, más uniforme. La Figura 4.1(f) muestra la iluminación producida en la escena correspondientes a los intervalos de iluminación en la Figura 4.3.



(a) 1 luminaria, CV = 0.90.

(b) 2 luminarias, CV = 0.42.

(c) 3 luminarias, CV = 0.21.

Figura 4.3: Intervalos de iluminación generados para el experimento de uniformidad, al variar la cantidad de luminarias (1, 2, 3 de izquierda a derecha). Imágenes generadas con Dialux. Las bombillas muestran dónde se localizan las luminarias (en el techo de la escena).

4.6. Evaluación paralela de individuos

El experimento consiste en analizar el efecto de aplicar técnicas de evaluación paralela para la metaheurística VNS con el objetivo de conseguir una mejora en la eficiencia computacional (como fue descrito en la Sección 3.9). Este experimento se realiza con los mismos objetivos y restricciones que los utilizados en el experimento de eficiencia en consumo eléctrico (Sección 4.3), comparando los tiempos necesarios para realizar las iteraciones con dicho experimento.

Para este experimento se configuró como máximo número de iteraciones a 25000 y como escena al Patio.

La Tabla 4.5 muestra los tiempos necesarios para realizar la optimización usando la evaluación paralela de individuos. *Par* representa la optimización utilizando la evaluación paralela, mientras que *Sec* representa cuando se utiliza la evaluación secuencial de individuos. Como se puede observar se obtiene una aceleración del algoritmo paralelo frente al secuencial. El aumento de velocidad (speedup) es calculado como el cociente $^{Sec}/_{Par}$ para los valores medios de *Par* y *Sec*.

Tabla 4.5: Resultados de mejora de performance para la evaluación paralela de individuos. Los tiempos son medidos y promediados para 20 ejecuciones independientes. Los individuos evaluados en paralelo, por el producto matriz-matriz, están entre 200 y 1700.

#lum	mejor (m)		peor (m)		$\mu \pm \sigma$ (m)		speedup
	Sec	*Par*	*Sec*	*Par*	*Sec*	*Par*	
1	25	**22**	34	**31**	29 ± 4	$\mathbf{23} \pm 3$	1,26
2	54	**44**	76	**47**	62 ± 8	$\mathbf{45} \pm 1$	1,38
3	69	**67**	107	**69**	76 ± 14	$\mathbf{67} \pm 1$	1,13

Es importante notar que si bien el cálculo de la radiosidad es realizado en paralelo (producto matriz-matriz de la Sección 3.9), el resto de las operaciones del algoritmo siguen siendo secuenciales.

4.7. Algoritmo de ordenamiento

Todos los experimentos y resultados, excepto el de esta sección, fueron evaluados realizando el ordenamiento de las luminarias por medio del Algoritmo 2. Este experimento se realiza con los mismos objetivos y restricciones que los del experimento de eficiencia eléctrica (Sección 4.3) pero sin utilizar el ordenamiento de luminarias.

Comparando los resultados presentados en las Tablas 4.2 y 4.6 es posible identificar que cuando las luminarias están ordenadas, se tienen soluciones con un mayor consumo eléctrico que los obtenidos cuando sí están ordenadas. Por lo tanto es posible concluir que el ordenamiento resulta en beneficio de la búsqueda de óptimos cuando se utiliza el VNS como metaheurística de búsqueda.

Tabla 4.6: Resultados de optimización sin utilizar el algoritmo de ordenamiento.

# lum	mejor (W)	peor (W)	$\mu \pm \sigma$ (W)
2	371	442	433 ± 14
3	365	453	417 ± 30

4.8. Desempeño computacional

En esta sección se evalúan algunas de las variables y sus efectos en cuanto al desempeño computacional.

4.8.1. Cantidad de luminarias

Los datos que se muestran en esta evaluación de rendimiento son tomados de ejecuciones realizando las optimizaciones con los objetivos y restricciones del experimento de eficiencia en consumo eléctrico (Sección 4.3). Para este experimento se configuró como máximo número de iteraciones a 25000 y como escena al Patio. En este experimento se estudia el efecto en el desempeño computacional que causa la cantidad de luminarias utilizadas en una configuración.

La relación entre la cantidad de luminarias y el tiempo necesario para realizar las optimizaciones se muestra en la Tabla 4.7, donde se puede observar que existe una relación aproximadamente lineal entre el aumento de la cantidad de luminarias y el tiempo que toma realizar la optimización. Esto es debido a que agregar una luminaria a la configuración implica calcular su emisión en la escena y acumularlo en E, o sea incrementa en 1 los pasos realizados en la sumatoria presentada en la Ecuación 3.6.

Tabla 4.7: Resultados en minutos de rendimiento para el algoritmo evaluando el uso de rotaciones y cantidad de luminarias.

#lum	mejor (m)		peor (m)		$\mu \pm \sigma$ (m)		$\#it/s$	
	$\neg Rot$	Rot	$\neg Rot$	Rot	$\neg Rot$	Rot	$\neg Rot$	Rot
1	**13**	22	**15**	31	$\mathbf{14} \pm 1$	23 ± 3	**30**	18
2	**22**	44	**24**	47	$\mathbf{23} \pm 1$	45 ± 1	**18**	9
3	**32**	67	**37**	69	$\mathbf{33} \pm 1$	67 ± 1	**13**	6

Además podemos concluir que el algoritmo con rotaciones toma unas 2 veces más tiempo que el algoritmo sin rotaciones para realizar la misma cantidad de iteraciones (ver Tabla 4.7).

4.8.2. Tamaño de hemi-cubo

En este experimento se estudia el efecto en el desempeño computacional que causa el tamaño de hemi-cubos utilizado por el algoritmo para realizar los cálculos. Para este experimento se configuró como máximo número de iteraciones 25000 y la escena del Patio, además se utilizaron los mismos objetivos y restricciones que en el experimento de eficiencia en consumo eléctrico (Sección 4.3). Los experimentos son realizados sin utilizar rotaciones.

La Tabla 4.8 muestra el consumo eléctrico de las configuraciones obtenidas en la optimización y los tiempos que toma encontrarlas para los distintos tamaños de hemi-cubos evaluados. Como se puede notar hemi-cubos más pequeños resultan en un mejor desempeño pero retornan soluciones con peor consumo eléctrico. Esto es debido a que hemi-cubos más pequeños contienen menor información de la escena y por lo tanto resultan en cálculos menos precisos.

Para el paso de pre-cómputo, también se realiza un experimento que mide los tiempos necesarios para realizarlo evaluando distintos tamaños de hemi-cubos. En la Tabla 4.9,

el aumento del tamaño de los hemi-cubos tiene como efecto el aumento del tiempo necesario para realizar el pre-cómputo.

Tabla 4.8: Resultados de optimización para distintos tamaños de hemi-cubos.

H	$\mu\pm\sigma$ (W)	$\mu\pm\sigma$ (s)	error	speedup
512×512	152±10	1061±12	-	-
256×256	162±11	408±8	0,068	2,6
128×128	186±13	262±14	0,225	4,1
64×64	260±35	224±13	0,711	4,8

Tabla 4.9: Resultados de tiempos necesarios para realizar la etapa de pre-cómputo.

H	tiempo total (s)	speedup
512×512	20654	-
256×256	4860	4,3
128×128	1382	14,9
64×64	354	58,3

4.9. Algoritmo Evolutivo

Este experimento evalúa la técnica de algoritmos evolutivos utilizada como alternativa para realizar la búsqueda de soluciones (candidatos). Cabe destacar que para esta técnica no es necesario tener ordenadas las luminarias, obteniendo el consiguiente ahorro en tiempo de pre-cómputo.

En este experimento se utiliza la función de fitness descrita en la Ecuación (4.4) y se evalúan los resultados con el algoritmo de VNS. La función de fitness representa la sumatoria de las potencias de las luminarias seleccionadas y se agregan dos funciones de penalización (Ecuaciones (4.5) y (4.6)) para aquellas soluciones que no cumplan las restricciones impuestas.

$$\begin{cases} \sum_{i=1}^{n} P_i + pMin(Config) + pMax(Config) \\ Config = (x_1, y_1, l_1, ..., x_i, y_i, l_i, ..., x_n, y_n, l_n) \end{cases} \tag{4.4}$$

donde P es un vector con las potencias en watts de cada luminaria (P_i la potencia de la i-ésima luminaria en la configuración), x_i e y_i son las coordenadas espaciales de la i-ésima luminaria seleccionada en la configuración, l_i es el índice en la base de datos de la i-ésima luminaria seleccionada en la configuración, n es la cantidad total de luminarias que puede tener la configuración, pMax y pMin son las funciones de penalización siguientes:

$$\begin{cases} pMin(Config) = ((max(100 - (min(B_{Sup})/A_{Sup}), 0)^2) * 100) + max(P) * n, \\ \qquad\qquad cuando\ max(100 - (min(B_{Sup})/A_{Sup}), 0)^2) * 100 > 0 \\ y\ vale\ 0\ en\ el\ resto\ de\ los\ casos \end{cases} \tag{4.5}$$

$$\begin{cases} pMax(Config) = ((max((max(B_{Sup})/A_{Sup}) - 750,0)^2) * 100) + max(P) * n, \\ \qquad\qquad cuando\ max((max(B_{Sup})/A_{Sup}) - 750,0)^2) * 100 > 0 \\ y\ vale\ 0\ en\ el\ resto\ de\ los\ casos \end{cases} \tag{4.6}$$

En las funciones de penalización A_{Sup} representa el área de cada parche de la superficie objetivo y B_{Sup} es un vector que contiene los valores de radiosidad para cada parche de la superficie objetivo.

El análisis en esta sección se enfoca en realizar la evaluación del desempeño del algoritmo en base a la calidad de soluciones obtenidas en los casos donde se evalúa la iluminación utilizando 2, 3 y 4 luminarias. Para este experimento se utilizó la escena del Patio.

4.9.1. Ajuste paramétrico

En el algoritmo evolutivo se realizó una etapa de ajuste paramétrico. Debido a que los algoritmos evolutivos resultan intrínsecamente no determinísticos, previo al análisis experimental, es necesario realizar un ajuste de los parámetros de probabilidad de cruzamiento (P_c), probabilidad de mutación (P_m) y el tamaño de la población ($TPob$). El objetivo del ajuste es encontrar la combinación de estos parámetros que logre alcanzar los mejores resultados para el caso estudiado. Para realizar dicho ajuste se crearon las combinaciones resultantes del producto cartesiano de los conjuntos; probabilidad de cruzamiento $P_m = \{0{,}01;\ 0{,}02;\ 0{,}03\}$, probabilidad de mutación $P_c = \{0{,}45;\ 0{,}60;\ 0{,}75\}$ y tamaño de población $TPob = \{20; 40; 60\}$ (ver 4.7).

$$\begin{aligned} Casos &= P_m \times P_c \times TPob \\ &\quad donde \\ P_m &= \{0{,}01;\ 0{,}02;\ 0{,}03\} \\ P_c &= \{0{,}45;\ 0{,}60;\ 0{,}75\} \\ TPob &= \{20; 40; 60\} \end{aligned} \tag{4.7}$$

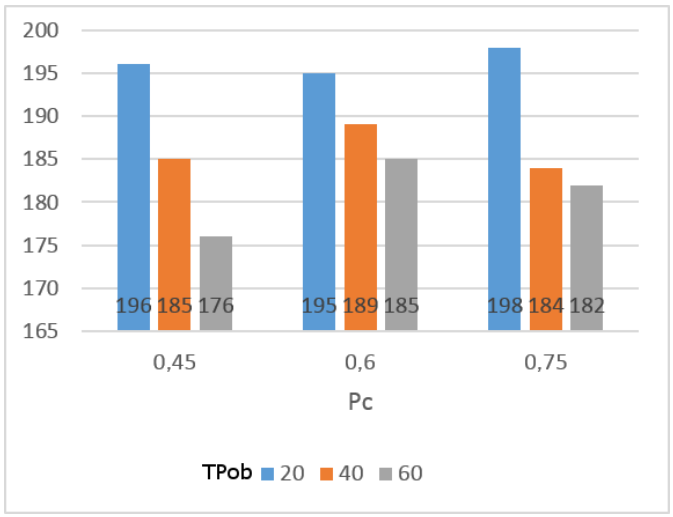
Resultando en 27 casos. Luego se corrieron 20 ejecuciones independientes con criterio de parada el alcanzar 300 generaciones o que 100 generaciones consecutivas cumplan que el promedio de cambio promedio relativo a la función de fitness sea menor a 1×10^{-3}.

Los resultados obtenidos luego del análisis paramétrico (ver Tabla 4.10 y Figura 4.4) determinan que los mejores resultados para las configuraciones estudiadas, se dan en la configuración determinada por los paramétros, $P_m = 0,03$, P_m=$0,45$, $TPob = 60$.

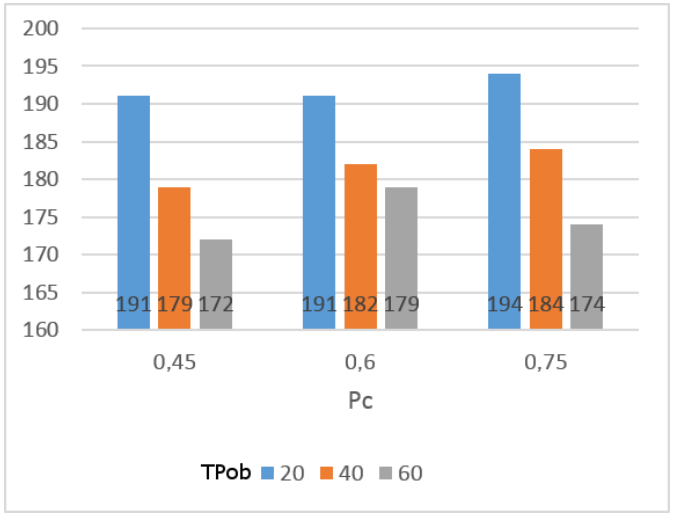
Luego de ajustados los parámetros se realizaron optimizaciones con la mejor configuración paramétrica. En este caso se realizaron para cada instancia 30 ejecuciones independientes (en lugar de las 20 realizadas en el ajuste de parámetros). El algoritmo implementado consiste en un algoritmo evolutivo basado en el modelo de evolución (μ, λ) y en el uso de los operadores evolutivos antes mencionados (ver Sección 3.8.2).

Tabla 4.10: Resultados de ejecuciones de 20 instancias para cada combinación de probabilidad de mutación con probabilidad de cruzamiento y con tamaño de la población. Tiempo reportado en segundos. De esta tabla es posible determinar la combinación de parámetros, de entre los estudiados, que retorna mejores soluciones en promedio. Se reporta la media y desviación estándar de la sumatoria de potencias de las luminarias en la configuracón y del tiempo necesario para realizar la optimización.

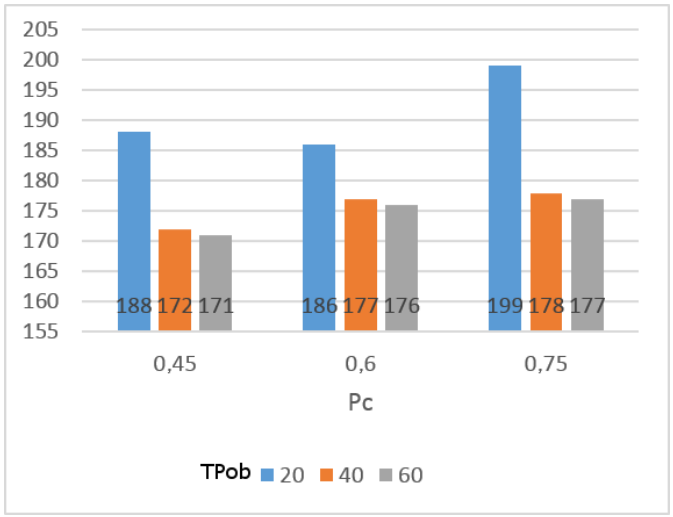
P_m	P_c	$TPob$	$\mu(F) \pm \sigma$	$\mu(t) \pm \sigma$
0,01	0,45	20	196 ± 16	432 ± 130
0,01	0,45	40	185 ± 16	328 ± 113
0,01	0,45	60	176 ± 13	441 ± 150
0,01	0,60	20	195 ± 13	158 ± 42
0,01	0,60	40	189 ± 12	286 ± 83
0,01	0,60	60	185 ± 13	392 ± 117
0,01	0,75	20	198 ± 14	151 ± 49
0,01	0,75	40	184 ± 14	276 ± 88
0,01	0,75	60	182 ± 14	383 ± 104
0,02	0,45	20	191 ± 11	179 ± 69
0,02	0,45	40	179 ± 12	308 ± 101
0,02	0,45	60	172 ± 10	369 ± 104
0,02	0,60	20	191 ± 15	164 ± 58
0,02	0,60	40	182 ± 13	264 ± 86
0,02	0,60	60	179 ± 14	374 ± 126
0,02	0,74	20	194 ± 17	186 ± 65
0,02	0,75	40	184 ± 14	330 ± 132
0,02	0,75	60	174 ± 15	405 ± 126
0,03	0,45	20	188 ± 14	175 ± 54
0,03	0,45	40	172 ± 12	321 ± 128
0,03	0,45	60	$\mathbf{171} \pm 11$	450 ± 125
0,03	0,60	20	186 ± 27	171 ± 52
0,03	0,60	40	177 ± 13	287 ± 85
0,03	0,60	60	176 ± 14	456 ± 135
0,03	0,75	20	199 ± 27	174 ± 65
0,03	0,75	40	178 ± 12	342 ± 138
0,03	0,75	60	177 ± 14	363 ± 109



(a) Pm =0,01



(b) Pm =0,02

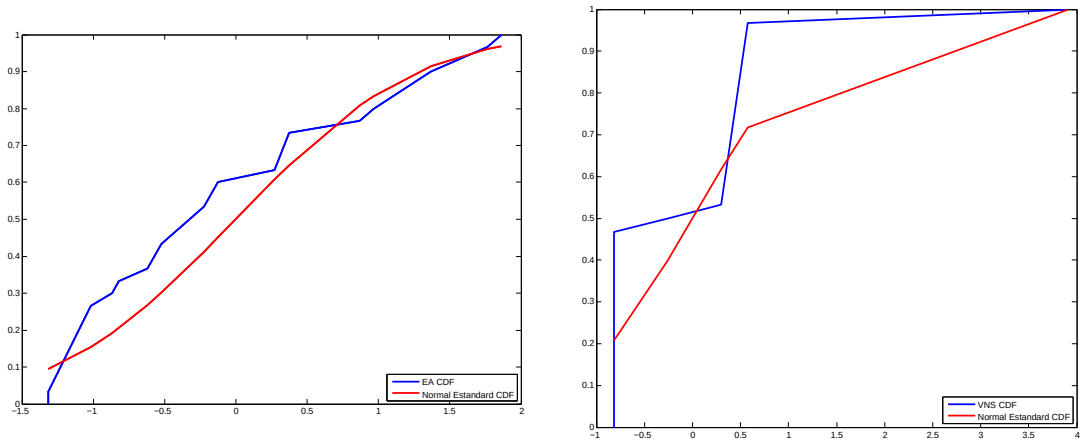


(c) Pm =0,03

Figura 4.4: Resultados de ajuste paramétrico con poblaciones de 20, 40 y 60 individuos para cada probabilidad de mutación (Pm) y cada probabilidad de cruzamiento (Pc). Eje y representa la suma, medida en watts, del consumo de las luminarias en la configuración.

4.9.2. Test de Kolmogorov-Smirnov

Se aplicó el test de Kolmogorov-Smirnov para determinar si es posible descartar que los fitness obtenidos mediante la aplicación de los algoritmos resultan normales. En ambos casos, el algoritmo evolutivo y el algoritmo basado en VNS, se obtuvo que no es posible descartar la hipótesis de que las muestras no cumplan tener una distribución normal, retornando un p-valor de $6,96 \times 10^{-28}$. En la Figura 4.5 se pueden ver los gráficos de comparación de los resultados obtenidos para cada algoritmo en comparación a la curva normal, notándose que se aproximan a esta. Esto nos permite realizar el test de Student para comparar los resultados obtenidos en ambos algoritmos y estudiar cuál de estos retorna mejores resultados.



(a) Algoritmo Evolutivo (b) VNS

Figura 4.5: Gráfico de comparación con curva normal. En ambos casos las curvas son similares a la de la distribución normal.

4.9.3. Comparación entre algoritmos

En la Tabla 4.11 y Figura 4.6 se reportan los resultados obtenidos para las tres instancias evaluadas (dos, tres y cuatro luminarias) para ambos algoritmos. Como se puede ver en dicha tabla el algoritmo EA solamente logró mejorar el mejor fitness del VNS para la instancia con tres luminarias. Sin embargo, en los demás casos y para la media siempre se tienen mejores resultados mediante la aplicación del algoritmo basado en VNS notándose grandes diferencias en los resultados de las madias entre ambos algoritmos.

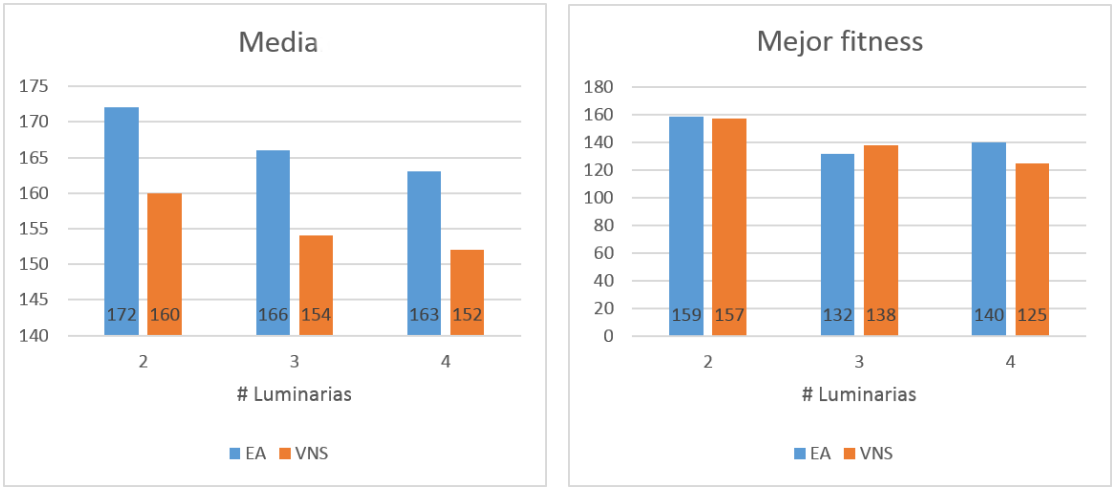
En cuanto a la performance los resultados muestran que el algoritmo basado en VNS retorna mejores soluciones en todos los casos.

4.9.4. Test de Student

Dado el resultado del test de normalidad de Kolmogorov-Smirnov se puede realizar el test de Student a los resultados obtenidos por los algoritmos. En la Tabla 4.12 se

Tabla 4.11: Resultados de instancias evaluadas para cada algoritmo

$\#lum$	$Alg.$	$min(F)$	$\mu(F) \pm \sigma$	$\mu(t) \pm \sigma$
2 lum	EA	159	172 ± 10	631 ± 22
2 lum	VNS	**157**	$\mathbf{160} \pm 3$	$\mathbf{558} \pm 28$
3 lum	EA	**132**	166 ± 9	956 ± 29
3 lum	VNS	138	$\mathbf{154} \pm 11$	$\mathbf{808} \pm 8$
4 lum	EA	140	163 ± 11	$1,1 \times 10^3 \pm 35$
4 lum	VNS	**125**	$\mathbf{152} \pm 10$	$\mathbf{1,0 \times 10^3} \pm 11$



(a) Media

(b) Mejor fitness

Figura 4.6: Resultados ejecuciones de instancias con 2, 3 y 4 luminarias. Eje de las ordenadas representa la suma, medida en watts, del consumo de las luminarias en la configuración.

pueden ver los resultados de la prueba de Student aplicada a dos muestras (EA y VNS) para cada una de los tres casos evaluados en este reporte.

Tabla 4.12: Resultados de test de Student. $h = 1$ indica que el test de Student no rechaza la hipótesis de que la distribución del VNS es peor que la obtenida para el algoritmo evolutivo, con un p-valor del 5 %. La columna p-valor muestra la probabilidad de que se cumpla dicha hipótesis.

$\#lum$	h	p-valor
2 lum	1	$4,17 \times 10^{-8}$
3 lum	1	$1,54 \times 10^{-5}$
4 lum	1	$1,07 \times 10^{-4}$

De estos resultados es posible deducir que el algoritmo VNS da mejores resultados en todos los casos.

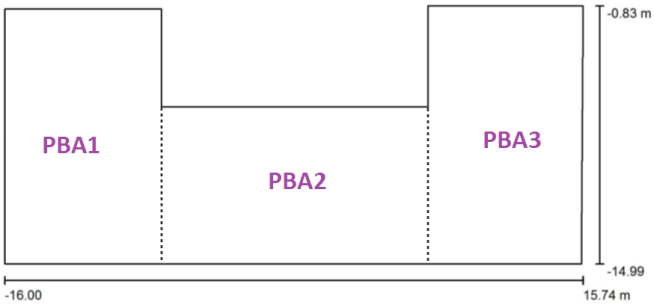
4.10. Caso de estudio, edificio Palacio de los Tribunales (Poder Judicial)

En esta sección se estudia la aplicación del algoritmo en dos casos de estudio reales. El primer caso de estudio (realizado en la Sección 4.10.1) busca optimizar la configuración encontrada actualmente en las oficinas del edificio Palacio de los Tribunales. Luego, en un segundo caso de estudio (realizado en la Sección 4.10.2), se optimiza la configuración que es propuesta por los expertos en iluminación. El edificio Palacio de los Tribunales consta de cinco pisos y una planta baja. Los pisos poseen exactamente la misma geometría, por lo tanto para el estudio de la iluminación artificial alcanza con estudiar la iluminación en la planta baja (modelado con 9692 parches) y en alguno de los pisos (modelado con 14012 parches). Los planos de dichas áreas se pueden ver en la Figura 4.7.

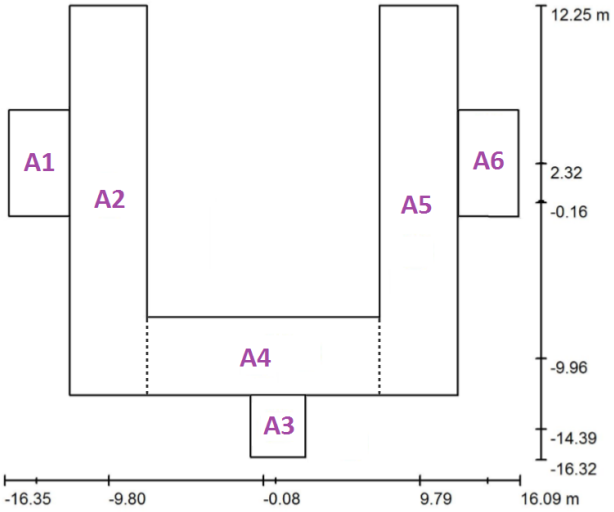
Las escenas son pisos en un edificio de oficinas público para el cual su iluminación fue diseñada hace 20 años haciendo uso de luminarias actualmente obsoletas. El surgimiento de la tecnología LED permite mejorar de manera sustancial la eficiencia energética de cualquier configuración, lo que resulta en grandes ahorros en edificios de este tipo. La mayoría de los edificios de oficinas hacen un uso intensivo de la iluminación artificial para satisfacer estándares de iluminación que permitan a sus ocupantes desempeñar actividades en condiciones óptimas.

4.10.1. Optimización según valores de la configuración actual

El principal objetivo de este experimento es comparar los resultados, en un caso de estudio real, con aquel propuesto por expertos en iluminación. Ambas configuraciones de luminarias intentan optimizar el uso de la energía. Se utilizan restricciones para que las configuraciones obtenidas tengan valores de iluminación similares, o mejores, a los actuales (no se pretende empeorar las condiciones de iluminación actuales en ningún sentido). Las geometrías de los pisos del edificio estudiado se pueden ver en las Figuras 4.7 y 4.8. Los valores que se desean optimizar son calculados en las superficies de trabajo que están por encima del piso, a una altura de 0.8m. En el caso de la planta baja, la configuración se compone de 80 luminarias distribuidas en dos grupos, mientras que el quinto piso posee 120 luminarias distribuidas en 3 grupos (ver Figura 4.9). Cabe destacar que se desea mantener dichos grupos de luminarias y que por lo tanto la posición de



(a) Planta baja



(b) Quinto piso

Figura 4.7: Planos de las escenas a optimizar obtenidos por medio de informes de Dialux.

cada una de las luminarias está predeterminada. Por otra parte, no tiene sentido aplicar todas las rotaciones posibles. Solo se aplican rotaciones de a 90º en el eje vertical.

Las condiciones que se desean mantener o mejorar son: la cantidad de luz recibida (medida como la media de iluminancia) en la superficie, y la uniformidad de la iluminancia (medida como el mínimo valor de esta dividido por la media), para cada una de las superficies de trabajo estudiadas (Figura 4.7). Este cálculo de uniformidad es el que utiliza Dialux y es diferente al realizado en la Sección 4.5. Formalmente, para el quinto piso la optimización pude ser resumida como sigue:

$$\text{mín } 36P_1 + 64P_2 + 20P_3, \quad s.a. : \frac{I(A_j)_{min}}{I(A_j)_{media}} \geq \frac{I(A_j)^{actual}_{min}}{I(A_j)^{actual}_{media}}, \quad I(A_j)_{media} \geq I(A_j)^{actual}_{media}, \quad j \in \{1 \dots 6\} \tag{4.8}$$

donde P_i es el consumo eléctrico medido en watts de la i-ésima luminaria en la configuración e I es el vector que contiene los valores de iluminancia para cada parche en cada superficie objeto de estudio A_j (ver Figura 4.7). I^{actual} son los valores de iluminancia para la configuración que se encuentra actualmente y que se busca mantener o mejorar por el algoritmo, I_{min} es el mínimo valor de iluminancia en la superficie e I_{media} su valor medio. I_{min}/I_{media} es la medida de que tan uniformemente distribuida se encuentra la iluminancia en la superficie siendo estudiada.

Por otro lado, la optimización para la planta baja pude ser resumida como sigue:

$$\text{mín } 48P_1 + 32P_2, \quad s.a. : \frac{I(PBA_j)_{min}}{I(PBA_j)_{media}} \geq \frac{I(PBA_j)^{actual}_{min}}{I(PBA_j)^{actual}_{media}}, \quad I(PBA_j)_{media} \geq I(PBA_j)^{actual}_{media}, \quad j \in \{1 \dots 3\} \tag{4.9}$$

donde PBA_j es una superficie en la planta baja. En ambos casos (planta baja y quinto piso), la optimización busca encontrar los mismos o mejores valores de iluminancia que los encontrados actualmente en el edificio Palacio de los Tribunales, mejorando a su vez el consumo eléctrico. La cantidad máxima de iteraciones permitida es 25000.

Tabla 4.13: Resultados de la iluminación para la planta baja, medidos con Dialux para la configuración actual (*actual*), la propuesta por los expertos en iluminación (*expertos*) y la obtenida por el algoritmo (*algoritmo*). Los mejores valores se destacan con negritas, los valores que sean menores a los esperados (menores que los que se encuentran en la configuración actual, i.e., columna *actual*) son marcados con un asterisco.

Superficie	I_m (lx)			I_{min}/I_m		
	actual	*expertos*	*algoritmo*	*actual*	*expertos*	*algoritmo*
PB-A1	107	219	**343**	0,71	0,69*	**0,72**
PB-A2	152	**313**	252	0,75	0,64*	**0,83**
PB-A3	107	218	**341**	0,69	0,64*	**0,70**

En la Tabla 4.15 se exponen los resultados para la planta baja. Dicha tabla muestra los resultados de iluminación obtenidos para cada configuración. Como se puede observar, la configuración propuesta por los expertos no mantiene ni mejora los valores actuales de uniformidad pero si mantiene los niveles de iluminancia media. Por otro lado, la configuración encontrada por el algoritmo sí mantiene o mejora tanto los niveles medios de iluminancia como también su uniformidad. Además la Tabla 4.15 muestra el ahorro energético obtenido con cada una de las configuraciones (medido como el porcentaje de watts ahorrados por las nuevas configuraciones con respecto a la existente en la actualidad). De esta última tabla es posible concluir que la configuración propuesta por el algoritmo es la que retorna los mejores niveles de ahorro energético, existiendo un ahorro de cerca de 9 % más que la configuración propuesta por los expertos. A partir

Tabla 4.14: Resultados de la iluminación para el quinto piso, medidos con Dialux para la configuración actual (*actual*), la propuesta por los expertos en iluminación (*expertos*) y la obtenida por el algoritmo (*algoritmo*). Los mejores valores se destacan con negritas, los valores que sean menores a los esperados (menores que los que se encuentran en la configuración actual, i.e., columna *actual*) son marcados con un asterisco.

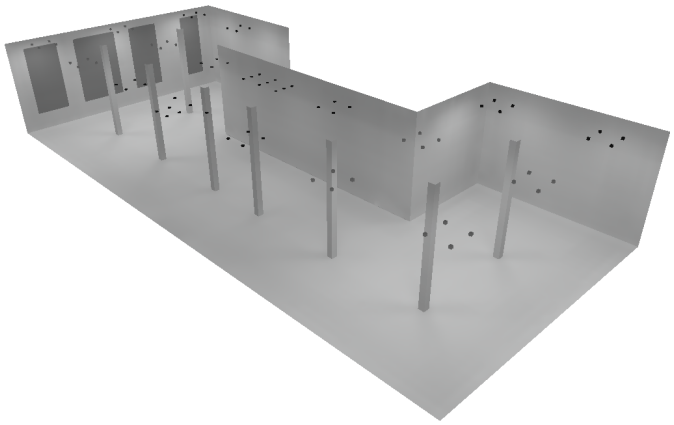
Superficie	I_m (lx)			I_{min}/I_m		
	actual	*expertos*	*algoritmo*	*actual*	*expertos*	*algoritmo*
A1	474	348*	**542**	0,58	**0,59**	**0,59**
A2	**648**	537*	623*	0,55	0,49*	**0,57**
A3	398	285*	**436**	0,54	0,54	**0,56**
A4	661	**670**	665	0,77	0,61*	**0,78**
A5	**656**	544*	627*	0,55	0,48*	**0,57**
A6	473	344*	**533**	0,57	0,58	**0,59**

Tabla 4.15: Resultados para la planta baja. Consumo eléctrico y ahorro obtenidos para la configuración actual (*actual*), la propuesta por los expertos (*expertos*) y la encontrada por el algoritmo (*algoritmo*). Los mejores valores se destacan con negritas.

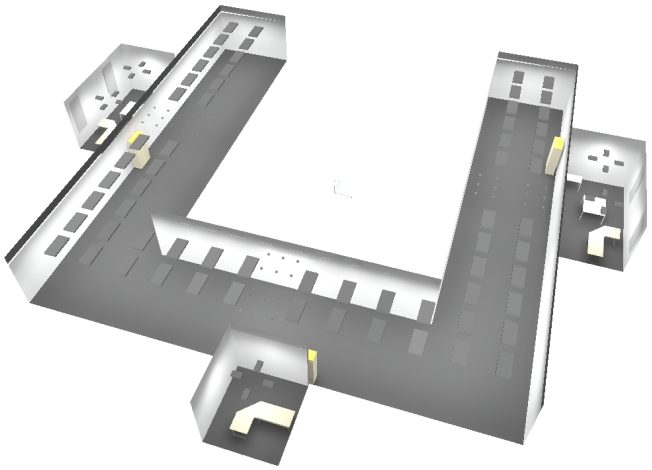
	actual	*expertos*	*algoritmo*
Consumo eléctrico (W)	5440	2240	**1746**
Ahorro energético (%)	-	58,8 %	**67,9** %

Tabla 4.16: Resultados para el quinto piso. Consumo eléctrico y ahorro obtenidos para la configuración actual (columna *actual*), la configuración propuesta por los expertos (columna *expertos*) y la configuración optima encontrada por el algoritmo (columna *algoritmo*). Los mejores valores son destacados con negrita.

	actual	*expertos*	*algoritmo*
Consumo eléctrico (W)	11664	**3546**	4011
Ahorro energético (%)	0 %	**69,6** %	65,6 %



(a) Planta baja



(b) Quinto piso

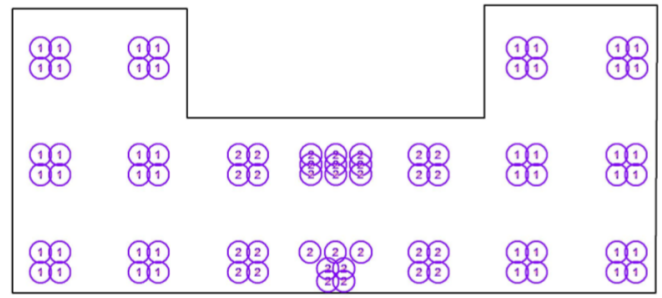
Figura 4.8: Geometrías de los pisos del edificio Palacio de los Tribunales renderizadas con Dialux.

de los datos de ambas tablas es posible concluir que la configuración planteada por el algoritmo es mejor que la propuesta por los expertos, ya que es capaz de mantener o mejorar los niveles actuales de iluminación a la vez que mejora el ahorro energético.

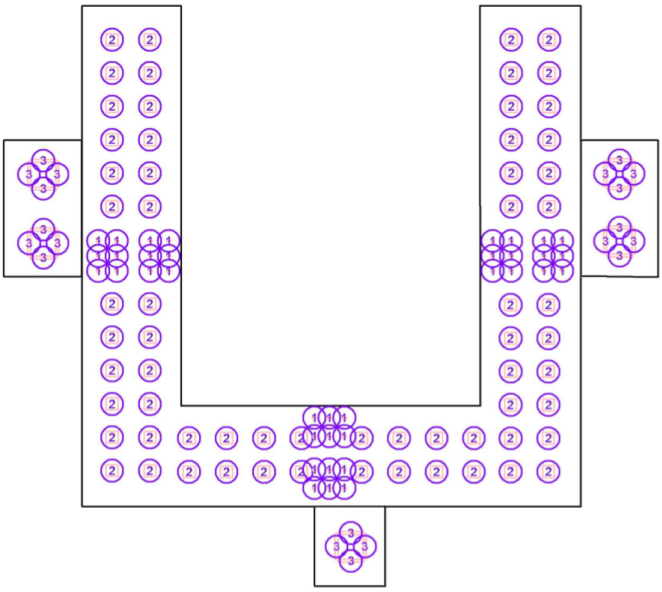
Por otro lado, para el quinto piso tenemos la Tabla 4.14 donde se muestran los resultados de iluminación obtenidos para cada configuración. En este caso, la propuesta de los expertos no cumple las restricciones de mantener o mejorar la iluminación media y la uniformidad. Por otra parte, el algoritmo retorna valores que siempre mejoran la uniformidad de la luz en cada una de las superficies estudiadas, y brinda buenos valores de iluminancia media (solo dos superficies tienen valores levemente inferiores a las actuales).

La Tabla 4.16 muestra el ahorro energético para el quinto piso. Donde se puede ver que la propuesta de los expertos es la que retorna los valores de consumo eléctrico.

Los resultados obtenidos en ambas tablas (4.14 y 4.16) no permiten sacar conclusiones concretas. Esto es debido a que sí bien la propuesta realizada por los expertos es la que retorna los mejores valores en cuanto al consumo eléctrico, empeora los niveles de iluminación que se encuentran actualmente en el edificio. Esta relajación de las restricciones no es realizada por el algoritmo, lo que lleva a que este no considere como



(a) Planta baja



(b) Quinto piso

Figura 4.9: Posición e identificador de grupo de cada luminaria. Existen dos grupos distintos de luminarias en la configuración de la planta baja y tres para el quinto piso.

válidas soluciones similares a las propuestas por los expertos. Queda por tanto conocer si estableciendo como restricciones los valores de iluminación alcanzados por la propuesta realizada por los expertos, el algoritmo es capaz de obtener una mejor eficiencia energética. Este experimento se desarrolla en la Sección 4.10.2.

Luminarias seleccionadas en cada configuración

A continuación se muestran los nombres y códigos de las luminarias utilizadas en cada configuración.

Planta baja:

Luminaria actuales

- Grupo 1: LUMENAC S.A. OT6769P6.ASC (código = OT6769P6.ASC)
- Grupo 2: LUMENAC S.A. OT6769P6.ASC (código = OT6769P6.ASC)

Configuración propuesta por los expertos

- Grupo 1: FLOS 03.6610.3B L-SUPLY FIX BLN 4000K CRI80 (código = 03.6610.3B)
- Grupo 2: FLOS 03.6610.3B L-SUPLY FIX BLN 4000K CRI80 (código = 03.6610.3B)

Luminarias seleccionadas en la configuración por el algoritmo

- Grupo 1: CREE CR-LE-40LHE-30K-XXX Troffer 3000K No Trim (código = CR-LE-40LHE-30K-XXX)
- Grupo 2: Cree Inc LR6-10L-40K-GU24 Downlight with White Aluminum Housing and Diffuse Lens (código =LR6-10L-40K-GU24)

Quinto piso:

Luminaria actuales:

- Grupo 1: LUMENAC S.A. Embutido bajo consumo, con aro en policarbonato blanco y reflector de policarbonato metalizado 210-126 (código = Embutido bajo consumo, con aro en policarbonato blanco y reflector de policarbonato metalizado)
- Grupo 2: LUMENAC S.A. OFFICE C336 PS-90 Lum.p/embutir, reflector aluminio anodizado, con louver parabolico simple satinado economico (código = OFFICE C336 PS-90)
- Grupo 3: LUMENAC S.A. OT6769P5.ASC Ensayo Fotometrico (código = OT6769P5.ASC)

Luminarias en la configuración propuesta por los expertos

- Grupo 1: Disano Illuminazione SpA 841 3x led CLD CELLE 841 Minicomfort LED x3 -UGR¡16 (código = 841 3x led CLD CELL-E)
- Grupo 2: FLOS 03.6610.3B L-SUPLY FIX BLN 4000K CRI80 (código = 03.6610.3B)
- Grupo 3: FLOS N70D114GA IN-FINITY 70 L=1125 SUSPENSION DOWN GENERAL LIGHTING (código = N70D114GA)

Luminarias seleccionadas en la configuración por el algoritmo

- Grupo 1: CREE CR-LE-40LHE-30K-XXX Troffer 3000K No Trim (código = CR-LE-40LHE-30K-XXX)
- Grupo 2: Cree Inc FP24-50L-35K-10V 2'X4'flat panel fixture consisting of a white metal frame, a diffusion panel, and a light guide. (código = FP24-50L-35K-10V)
- Grupo 3: Cree Inc LR6-10L-27K-GU24 Downlight with White Aluminum Housing and Diffuse Lens (código = LR6-10L-27K-GU24)

4.10.2. Optimizado de configuración propuesta por los expertos

A partir de los resultados obtenidos en el experimento presentado en la Sección 4.10.1 se puede ver que en el caso del quinto piso los expertos relajan algunas de las restricciones de iluminación, con el objetivo de mejorar el consumo eléctrico de la configuración resultante. Sin embargo, dicha relajación no es aprovechada por el algoritmo, que se limita a cumplir las restricciones de forma estricta. Sospechamos que esa es la principal causa de que la solución encontrada por el algoritmo sea energéticamente menos eficiente. Por esta causa se desarrolló este experimento que busca optimizar la configuración planteada por los expertos utilizando los valores de la misma en lugar de los valores en la configuración actual. En este experimento se optimiza la superficie de trabajo que está por encima del piso a una altura de 0.8m, y se realiza un número máximo de 25000 iteraciones.

Tabla 4.17: Tabla que contiene los resultados de iluminación, medidos por medio de Dialux, para la propuesta de los expertos (*expertos*) y la encontrada por medio del algoritmo (*algoritmo*) partiendo de la configuración planteada por los expertos. Los mejores valores obtenidos para cada fila son destacados con letra negrita, aquellos valores que resulten inferiores a los esperados (aquellos menores a los de la columna *expertos*) son destacados con un asterisco.

Superficie	I_m (lx)		I_{min}/I_m	
	expertos	*algoritmo*	*expertos*	*algoritmo*
A1	**348**	**348**	**0,59**	**0,59**
A2	537	**543**	**0,49**	**0,49**
A3	**285**	**285**	**0,54**	**0,54**
A4	670	**676**	**0,61**	0,60*
A5	544	**549**	**0, 48**	**0, 48**
A6	**344**	**344**	**0, 58**	**0, 58**

Tabla 4.18: Consumo eléctrico y ahorro obtenidos por las configuraciones en relación a la existente en el edificio Palacio de los Tribunales (*actual*), la configuración propuesta por los expertos (*expertos*) y la configuración optima obtenida por el algoritmo (*algoritmo*) utilizando la solución planteada por los expertos como punto de partida y buscando mejorarla. La mejor solución es destacada con letra negrita.

	actual	*expertos*	*algoritmo*
Consumo eléctrico (W)	11664	3546	**3041**
Ahorro energético (%)	0 %	69,6 %	**73,9** %

La Tabla 4.17 muestra los resultados en la iluminación obtenidos para cada una de las configuraciones. Como se puede observar, la configuración encontrada por el algoritmo obtiene prácticamente los mismos resultados en la iluminación que los resultados obtenidos por la configuración propuesta por los expertos. Por otro lado la Tabla 4.18 muestra que la configuración encontrada por el algoritmo mejora el consumo eléctrico en casi un 4 %. De los resultados es posible observar que, con las nuevas restricciones usadas, el algoritmo es capaz de mejorar la configuración propuesta por los expertos, obteniendo una valor 14 % menor con un ahorro energético casi 4 % mayor con respecto a la configuración actual. Además la configuración resultante tiene valores prácticamente idénticos a la propuesta realizada por los expertos.

Luminarias seleccionadas en la nueva configuración del quinto piso

Los tipos de luminaria seleccionados en esta nueva configuración son como siguen:

- Grupo 1: Disano Illuminazione SpV 841 3x led CLD CELL-D-D-E 841 Mini-comfort LED x3 -UGR $<$ 16 (código = 841 3x led CLD CELL-D-D-E).
- Grupo 2: ERCO GmbH 32001000_V01 Quintessence Downlight (código = 32001000_V01).
- Grupo 3: FLOS N70D114GA IN-FINITY 70 L=1125 SUSPENSION DOWN GENERAL LIGHTING (código = N70D114GA).

Capítulo 5

Conclusiones y trabajo futuro

5.1. Conclusiones

En esta tesis se introduce un método que resuelve problemas inversos de iluminación como problemas de optimización. La optimización de la iluminación en una escena considera como variables la posición, la orientación y el tipo de luminaria utilizada (su emisión). Se utiliza una base de datos que consiste de miles de luminarias distintas, y se consideran cientos de posiciones y orientaciones (rotaciones) distintas. Este método posibilita la inclusión de luminarias complejas, cuyas emisiones están definidas por archivos fotométricos que aproximan la emisión real de las mismas. Debido a esto, es necesario también contemplar la orientación de dichas luminarias.

Los resultados obtenidos cuando se orientan las luminarias muestran ser mejores en algunos experimentos, aumentando los tiempos de ejecución de estos. Esto último es debido al gran incremento que se produce en el tamaño del dominio de soluciones y a los cálculos realizados en cada iteración para calcular la iluminación. Por otra parte estos tiempos siguen siendo viables para su uso en el tipo de problemas estudiado. De la evaluación experimental se deduce que agregar rotaciones puede no ser la mejor opción en todos los casos. Por tanto, el uso de rotaciones debe ser evaluado para cada problema.

Si bien el método de radiosidad tiene sus debilidades frente a otras técnicas de iluminación global, como por ejemplo el que solo se puedan incluir superficies lambertianas, posee como una enorme fortaleza su velocidad de cálculo (que en nuestro caso fue además acelerado usando LRR). Esta ventaja comparativa permite resolver problemas de optimización con miles de cálculos de iluminación global, que por medio de otros métodos resultarían prohibitivos.

Los resultados obtenidos por el método fueron utilizados como entrada en Dialux, utilizado por los diseñadores de iluminación para obtener variedad de datos a partir de las configuraciones de luminarias en la escena. Los diseñadores de iluminación pueden utilizar este nuevo recurso para partir de soluciones iniciales que satisfacen sus intenciones de iluminación, y luego iterando sobre ellas alcanzar las soluciones definitivas.

En los experimentos, el algoritmo considera dominios de hasta 10^{21} configuraciones, y encuentra buenas soluciones luego de evaluar no más de 10^5 de ellas.

El ordenamiento de las luminarias funciona adecuadamente y facilita enormemente la introducción del tipo de luminaria como una variable más en el proceso de optimización. Esta técnica permite trabajar con total facilidad con conjuntos enormes de luminarias, aspecto no observado anteriormente en la literatura.

Una versión simplificada de algoritmo evolutivo fue implementada y comparada con VNS para determinar cuál de estos resulta mejor para la búsqueda de óptimos. Los resultados experimentales obtenidos no superan a los obtenidos con VNS, pero resultaron aceptables y alentadores. Es posible que los algoritmos evolutivos brinden resultados mucho mejores, si se realizan las mejoras adecuadas en el código.

El algoritmo fue probado en dos formas diferentes. Por un lado se evaluó su capacidad para aportar soluciones iniciales, y se lo comparó con las soluciones iniciales brindadas por los diseñadores. Por otra parte, se intentó mejorar una solución inicial brindada por los diseñadores. Los resultados de ambos experimentos fueron exitosos, retornando en ambos casos buenas configuraciones que cumplen estrictamente con los objetivos de intensidad y uniformidad de la iluminación y/o brindan mayores ahorros energéticos. Con esto se comprueba el potencial de los métodos implementados para brindar soluciones y hacer recomendaciones primarias a los diseñadores. Su inclusión en el proceso de diseño de iluminación dependerá de la facilidad con que se salve el abismo que separa los prototipos de las aplicaciones profesionales.

5.2. Trabajo futuro

Las principales líneas de trabajo futuro vislumbradas en esta tesis son:

- Ampliar las técnicas desarrolladas para contemplar también la existencia de luz natural en los espacios a iluminar.
- Clasificar las luminarias según el tipo de rotación permitida para cada una de ellas, y utilizar esta información en la optimización. Por ejemplo, existen luminarias que solo deben rotarse en el eje vertical.
- Realizar un cálculo de la iluminación global que contemple la distribución espectral de la luz.
- Extender el análisis a escenas más complejas, para cuantificar mejor las posibilidades y limitaciones de las técnicas desarrolladas.
- Estudiar el efecto que la discretización de los hemi-cubos, tanto de vista como de emisión, tiene sobre el cálculo de la iluminación de la escena.
- Estudiar métodos alternativos al del hemi-cubo para realizar el cálculo de la interacción de la luz emitida por la luminaria en la escena.
- Analizar problemas multi-objetivo. En esta tesis se trabaja con problemas mono-objetivo, utilizando restricciones para modelar otras intenciones de iluminación y así orientar las búsquedas. Un ejemplo de problema multi-objetivo es el de maximizar la uniformidad de la iluminación mientras se minimiza el consumo eléctrico.
- Algunas mejoras pueden ser aplicables al algoritmo evolutivo implementado. Un ejemplo de esto es la evaluación de nuevos operadores evolutivos más complejos, realizando el respectivo estudio y ajuste paramétrico. Otra posible mejora puede ser el uso de otro framework de algoritmos evolutivos que permita la utilización de algoritmos evolutivos paralelos, por ejemplo MALLBA [121], combinado con el código existente para el cálculo de la radiosidad.

Bibliografía

[1] S. Koziel and X. S. Yang. *Computational Optimization, Methods and Algorithms*. Studies in Computational Intelligence. Springer Berlin Heidelberg, 2011.

[2] G. E. Moore. Readings in computer architecture. chapter Cramming More Components Onto Integrated Circuits, pages 56–59. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2000.

[3] T. Ritschel, C. Dachsbacher, T. Grosch, and J. Kautz. The state of the art in interactive global illumination. *Comput. Graph. Forum*, 31(1):160–188, 2012.

[4] M. Contensin. Inverse lighting problem in radiosity. *Inverse Problems in Engineering*, 10(2):131–152, 2002.

[5] G. Patow and X. Pueyo. A survey of inverse rendering problems. *Computer Graphics Forum*, pages 663–687, 2003.

[6] E. Fernández. *Efficient Global Illumination Calculation for Inverse Lighting Problems*. PhD thesis, Universidad de la Republica, February 2014.

[7] B. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer Publishing Company, Incorporated, 4th edition, 2007.

[8] T. Whitted. An improved illumination model for shaded display. *Communications of the ACM*, 23(6):343–349, 1980.

[9] H. W. Jensen. Global illumination using photon maps. In *Proceedings of the eurographics workshop on Rendering techniques '96*, pages 21–30, London, UK, UK, 1996. Springer-Verlag.

[10] C. Goral, K. Torrance, D. Greenberg, and B. Battaile. Modeling the interaction of light between diffuse surfaces. *ACM SIGGRAPH Computer Graphics Quarterly*, 18(3):213–222, 1984.

[11] M. Cohen, J. Wallace, and P. Hanrahan. *Radiosity and realistic image synthesis*. Academic Press Professional, Inc., San Diego, CA, USA, 1993.

[12] S. Russell. *The Architecture of Light - Architectural Lighting Design Concepts and Techniques: A Textbook of Procedures and Practices for the Architect, Interior Designer and Lighting Designer*. Conceptnine, 2012.

[13] DIALux. Dial. light building software. http://www.dial.de, April 2016.

[14] Relux. Relux. lighting simulation tools. http://www.relux.biz, April 2016.

[15] S. R. Marschner. *Inverse Rendering for Computer Graphics*. PhD thesis, Ithaca, NY, USA, 1998.

[16] S. H. Shikder, M. M. Mourshed, and A. D. F. Price. Luminaire position optimization using radiance based simulation: a test case of a senior living room. In *Computing in Civil and Building Engineering*, 2010.

[17] I. E. Uygun, Z. T. Kazanasmaz, and S. Kale. Optimization of energy efficient luminaire layout design in workspaces. In Jean-Louis Scartezzini, editor, *CISBAT2015*, pages 301–306, Lausanne, 2015. LESO-PB, EPFL.

[18] M. Schwarz and P. Wonka. Procedural design of exterior lighting for buildings with complex constraints. *ACM Trans. Graph.*, 33(5):166:1–166:16, 2014.

[19] J. Foley, R. Phillips, J. Hughes, A. van Dam, and S. Feiner. *Introduction to Computer Graphics*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1994.

[20] H. Jensen. *Realistic image synthesis using photon mapping*. A. K. Peters, Ltd., Natick, MA, USA, 2001.

[21] G. Sakas and P. Shirley. *Photorealistic Rendering Techniques*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2001.

[22] J. T. Kajiya. The rendering equation. *SIGGRAPH Comput. Graph.*, 20(4):143–150, 1986.

[23] A. Glassner. *Principles of Digital Image Synthesis*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1994.

[24] S. Chandrasekar. *Radiative Transfer*. Oxford Univ. Press, 1950.

[25] A. Keller. Instant radiosity. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 49–56, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[26] B. Walter, S. Fernandez, A. Arbree, K. Bala, M. Donikian, and D. P. Greenberg. Lightcuts: a scalable approach to illumination. In *ACM SIGGRAPH 2005 Papers*, SIGGRAPH '05, pages 1098–1107, New York, NY, USA, 2005. ACM.

[27] M. Hašan, F. Pellacini, and K. Bala. Matrix row-column sampling for the many-light problem. *ACM Transactions on Graphics*, 26(3):26, 2007.

[28] Christensen P. Point-based approximate coolor bleeding. Technical report, Pixar, 2008.

[29] T. Ritschel, T. Engelhardt, T. Grosch, H. P. Seidel, J. Kautz, and C. Dachsbacher. Micro-rendering for scalable, parallel final gathering. In *ACM SIGGRAPH Asia 2009 papers*, SIGGRAPH Asia '09, pages 132:1–132:8, New York, NY, USA, 2009. ACM.

[30] P. Sloan, J. Kautz, and J. Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 527–536, New York, NY, USA, 2002. ACM.

[31] M. Cohen, D. Greenberg, D. Immel, and P. Brock. An efficient radiosity approach for realistic image synthesis. *IEEE Computer Graphics and Applications*, 6(3):26–35, 1986.

[32] G. Ward, F. Rubinstein, and R. Clear. A ray tracing solution for diffuse interreflection. In *Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, pages 85–92, New York, NY, USA, 1988. ACM.

[33] R. Kuchkuda. An introduction to ray tracing. In *F40, Theoretical Foundations of Computer Graphics and CAD, edited by R.A. Earnshaw, Springer-Verlag*, pages 1039–1060. Springer, 1988.

[34] C. Dachsbacher and M. Stamminger. Reflective shadow maps. In *Proceedings of the 2005 Symposium on Interactive 3D Graphics and Games*, pages 203–231, New York, NY, USA, 2005. ACM.

[35] M. Nijasure, S. Pattanaik, and V. Goel. Realtime global illumination on GPU. *Journal of Graphics Tools*, 10(2):55–71, 2005.

[36] P. Gautron, J. Křivánek, K. Bouatouch, and S. Pattanaik. Radiance cache splatting: a gpu-friendly global illumination algorithm. In *ACM SIGGRAPH 2008 classes*, pages 1–10, New York, NY, USA, 2008. ACM.

[37] T. Ritschel, T. Grosch, M. H. Kim, H.-P. Seidel, C. Dachsbacher, and J. Kautz. Imperfect shadow maps for efficient computation of indirect illumination. *ACM Transactions on Graphics*, 27(5):1–8, 2008.

[38] R. Wang, R. Wang, K. Zhou, M. Pan, and H. Bao. An efficient GPU-based approach for interactive global illumination. In *ACM SIGGRAPH 2009 papers*, pages 1–8, New York, NY, USA, 2009. ACM.

[39] E. Fernández, P. Ezzatti, S. Nesmachnow, and G. Besuievsky. Low-rank radiosity using sparse matrices. In Paul Richard, Martin Kraus, Robert S. Laramee, and José Braz, editors, *GRAPP/IVAPP*, pages 260–267. SciTePress, 2012.

[40] A. Appel. Some techniques for shading machine renderings of solids. In *Proceedings of the April 30–May 2, 1968, Spring Joint Computer Conference*, AFIPS '68 (Spring), pages 37–45, New York, NY, USA, 1968. ACM.

[41] S. A. Pedersen. Progressive photon mapping on gpus. Master's thesis, 2013.

[42] E. Catmull. *A subdivision algorithm for computer display of curved surfaces.* PhD thesis, The University of Utah, 1974.

[43] P. Dutre, K. Bala, P. Bekaert, and P. Shirley. *Advanced Global Illumination.* AK Peters Ltd, 2006.

[44] M. Cohen and D. Greenberg. The hemi-cube: a radiosity solution for complex environments. In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques*, pages 31–40, 1985.

[45] G. Coombe, M. Harris, and A. Lastra. Radiosity on graphics hardware. In *Proceedings of Graphics Interface 2004*, pages 161–168. Canadian Human-Computer Communications Society, 2004.

[46] W. Heidrich and H. Seidel. View-independent environment maps. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS workshop on Graphics hardware*, pages 39–45, New York, NY, USA, 1998. ACM.

[47] J. Blinn and M. Newell. Texture and reflection in computer generated images. *Communications of the ACM*, 19(10):542–547, 1976.

[48] F. X. Sillion and C. Puech. A General Two-Pass Method Integrating Specular and Diffuse Reflection. In *Proceedings of SIGGRAPH '89*, pages 335–344, Boston, United States, 1989. ACM Press.

[49] I. E. Sutherland, R. F. Sproull, and R. A. Schumacker. A characterization of ten hidden-surface algorithms. *ACM Comput. Surv.*, 6(1):1–55, 1974.

[50] G. V. G. Baranoski, R. Bramley, and J. G. Rokne. Eigen-analysis for radiosity systems. In *Proceedings of the Sixth International Conference on Computational Graphics and Visualization Techniques (Compugraphics '97)*, pages 193–201, Vilamoura, Algarve, Portugal, December 1997.

[51] I. Ashdown. Eigenvector radiosity. Master's thesis, Department of Computer Science, University of British Columbia, Vancouver, British Columbia, April 2001.

[52] E. Fernández. Low-rank radiosity. In O. Rodríguez, F. Serón, R. Joan-Arinyo, and E. Coto J. Madeiras, J. Rodríguez, editors, *Proceedings of the IV Iberoamerican Symposium in Computer Graphics*, pages 55–62. Sociedad Venezolana de Computación Gráfica, DJ Editores, C.A., June 2009.

[53] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 2013.

[54] E. Fernández and G. Besuievsky. Inverse opening design with anisotropic lighting incidence. *Computers and Graphics*, 47(1):113–122, 2015.

[55] S. Goreinov, E. Tyrtyshnikov, and N. Zamarashkin. A theory of pseudoskeleton approximations. *Linear Algebra and its Applications*, 261:1–21, 1997.

[56] J. P. Aguerre and E. Fernández. A hierarchical factorization method for efficient radiosity calculations. *Computers & Graphics*, 60:46–54, 2016.

[57] G. Stewart. *Matrix Algorithms – Volume I: Basic Decompositions*. Society for Industrial and Applied Mathematics, 1998.

[58] A. Tarantola. *Inverse Problem Theory and Methods for Model Parameter Estimation*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2004.

[59] P. C. Hansen, T. Sekii, and H. Shibahashi. The modified truncated svd method for regularization in general form. *SIAM J. Sci. Stat. Comput.*, 13(5):1142–1150, September 1992.

[60] J. Hadamard. Sur les problèmes aux dérivés partielles et leur signification physique. *Princeton University Bulletin*, 13:49–52, 1902.

[61] R. Ramamoorthi and P. Hanrahan. On the relationship between radiance and irradiance: determining the illumination from images of a convex lambertian object. *J. Opt. Soc. Am. A*, 18(10):2448–2459, October 2001.

[62] V. Harutunian, J. C. Morales, and J. R. Howell. Radiation exchange within an enclosure of diffuse-gray surfaces: The inverse problem. Inverse Problems in Heat Transfer, ASME/AIChE National Heat Transfer Conference, Portland, USA, 1995.

[63] G. Dantzig. The nature of mathematical programming, 2010.

[64] W. J. Cook, W. H. Cunningham, W. R. Pulleyblank, and A. Schrijver. *Combinatorial Optimization*. John Wiley & Sons, Inc., New York, NY, USA, 1998.

[65] M. R. Garey and D. S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

[66] Message Passing Interface Forum. Mpi: A message-passing interface standard, version 2.2. Specification, September 2009.

[67] T. White. *Hadoop: The Definitive Guide*. O'Reilly Media, Inc., 1st edition, 2009.

[68] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica. Spark: Cluster computing with working sets. In *Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing*, HotCloud'10, pages 10–10, Berkeley, CA, USA, 2010. USENIX Association.

[69] J. Pearl. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1984.

[70] I. Osman and G. Laporte. Metaheuristics: A bibliography. *Annals of Operations Research*, 63(5):511–623, 1996.

[71] C. Blum and A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv.*, 35(3):268–308, 2003.

[72] C. H. Papadimitriou and K. Steiglitz. *Combinatorial optimization: algorithms and complexity*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1982.

[73] H. Kargupta and D. E. Goldberg. Search, blackbox optimization, and sample complexity. In *In R.K. Belew & M. Vose (Eds.) Foundations of Genetic Algorithms 4*, pages 291–324. Morgan Kaufman, 1997.

[74] M. J. Atallah. *Handbook of algorithms and theory of computing*. CRC Press, 1999.

[75] A. Krzysztof. *Principles of Constraint Programming*. Cambridge University Press, New York, NY, USA, 2003.

[76] K. Deb and D. Kalyanmoy. *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, Inc., New York, NY, USA, 2001.

[77] C. Coello, G. B. Lamont, and D. A. van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[78] E. Talbi. *Metaheuristics: From Design to Implementation*. Wiley Publishing, 2009.

[79] D. G. Luenberger and Y. Ye. *Linear and Nonlinear Programming*. Springer Publishing Company, Incorporated, 2015.

[80] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[81] G. H. Golub and C. F. Van Loan. *Matrix computations (3rd ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.

[82] S. M. Goldfeld, R. E. Quandt, and H. F. Trotter. *Maximization by Quadratic Hill-Climbing*, volume 34. The Econometric Society, 1966.

[83] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2 edition, 2003.

[84] F. Castro, E. Acebo, and M. Sbert. Heuristic-search-based light positioning according to irradiance intervals. In Andreas Butz, Brian Fisher, Marc Christie, Antonio Krüger, Patrick Olivier, and Roberto Therón, editors, *Smart Graphics*, volume 5531 of *Lecture Notes in Computer Science*, pages 128–139. Springer Berlin Heidelberg, 2009.

[85] F. Castro, E. del Acebo, and M. Sbert. Energy-saving light positioning using heuristic search. *Engineering Applications of Artificial Intelligence*, 25(3):566–582, 2012.

[86] C. Schoeneman, J. Dorsey, B. Smits, J. Arvo, and D. Greenberg. Painting with light. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, pages 143–146, New York, NY, USA, 1993. ACM.

[87] J. K. Kawai, J. S. Painter, and M. F. Cohen. Radioptimization: goal based rendering. In Mary C. Whitton, editor, *SIGGRAPH*, pages 147–154. ACM, 1993.

[88] A. C. Costa, A. A. de Sousa, and F. N. Ferreira. Lighting design: A goal based approach using optimisation. In Dani Lischinski and Gregory Ward Larson, editors, *Rendering Techniques*, Eurographics, pages 317–328. Springer, 1999.

[89] G. J. Ward. The radiance lighting simulation and rendering system. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '94, pages 459–472, New York, NY, USA, 1994. ACM.

[90] F. Pellacini, F. Battaglia, R. K. Morley, and A. Finkelstein. Lighting with paint. *ACM Trans. Graph.*, 26(2):Article 9, 2007.

[91] N. Mladenovic and P. Hansen. Variable neighborhood search. *Computers & Operations Research*, 24(11):1097–1100, 1997.

[92] M. Gendreau and J. Potvin. *Handbook of Metaheuristics*. Springer Publishing Company, Incorporated, 2nd edition, 2010.

[93] J. Brownlee. *Clever Algorithms: Nature-Inspired Programming Recipes*. Lulu.com, 1st edition, 2011.

[94] P. Hansen, N. Mladenovic, and D. Pérez-Brito. Variable neighborhood decomposition search. *J. Heuristics*, 7(4):335–350, 2001.

[95] T. Baar, P. Brucker, and S. Knust. *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*. Springer US, 1 edition, 1999.

[96] E. Fernández and G. Besuievsky. Inverse lighting design for interior buildings integrating natural and artificial sources. *Computers & Graphics*, 36(8):1096–1108, 2012.

[97] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1989.

[98] T. Back, D. B. Fogel, and Z. Michalewicz, editors. *Handbook of Evolutionary Computation*. IOP Publishing Ltd., Bristol, UK, UK, 1st edition, 1997.

[99] M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA, USA, 1998.

[100] I. Rechenberg. Cybernetic solution path of an experimental problem. In *Royal Aircraft Establishment Translation No. 1122, B. F. Toms, Trans.* Ministry of Aviation, Royal Aircraft Establishment, Farnborough Hants, 1965.

[101] H. P. Schwefel. *Evolutionsstrategie und numerische Optimierung*. Technische Universität Berlin, 1975.

[102] J. H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, 1975. second edition, 1992.

[103] W. M. Spears. *Evolutionary algorithms : the role of mutation and recombination*. Springer, Berlin, Germany, 2000.

[104] J. E. Tena, I. Rudomin, A. Eugenio, G. Sada, and C. Gordo. An interactive system for solving inverse illumination problems using genetic algorithms. *Proceedings of computación visual*, 1997.

[105] K. P. Ferentinos and L. D. Albright. Optimal design of plant lighting system by genetic algorithms. *Engineering Applications of Artificial Intelligence*, 18(4):473–484, 2005.

[106] S. Delepoulle, C. Renaud, and M. Chelle. Genetic algorithms for light sources positioning. In *Proceedings of the 11th 3IA: International Conference on Computer Graphics and Artificial Intelligence*.

[107] M. J. Flynn. Some computer organizations and their effectiveness. *IEEE Trans. Comput.*, 21(9):948–960, 1972.

[108] T. Davidović and T. G. Crainic. *Parallelization strategies for variable neighborhood search.* CIRRELT, Centre interuniversitaire de recherche sur les réseaux déntreprise, la logistique et le transport= Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation, 2013.

[109] I. Decia, R. Leira, M. Pedemonte, E. Fernández, and P. Ezzatti. A VNS with parallel evaluation of solutions for the inverse lighting problem. In *Applications of Evolutionary Computation - 20th European Conference, EvoApplications 2017, Amsterdam, The Netherlands, April 19-21, 2017, Proceedings, Part I*, pages 741–756, 2017.

[110] P. Benner, P. Ezzatti, E. Quintana-Ortí, and A. Remón. On the impact of optimization on the time-power-energy balance of dense linear algebra factorizations. In *Proceedings of the 13th International Conference on Algorithms and Architectures for Parallel Processing - Volume 8286*, ICA3PP 2013, pages 3–10, New York, NY, USA, 2013. Springer-Verlag New York, Inc.

[111] S. Das, S. Maity, B. Qu, and P. N. Suganthan. Real-parameter evolutionary multimodal optimization a survey of the state-of-the-art. *Swarm and Evolutionary Computation*, 1(2):71–88, 2011.

[112] A. Pétrowski. A clearing procedure as a niching method for genetic algorithms. In *International Conference on Evolutionary Computation*, pages 798–803, 1996.

[113] IESNA-Computer-Committee. *IESNA standard file format for electronic transfer of photometric data.* IESNA lighting measurements series. Illuminating Engineering Society of North America, 1995.

[114] Heart Consultants Limited. Freeware eulumdat to ies lm-63-1995 conversion utility. `http://www.helios32.com/resources.htm#Formats`.

[115] Philips. Philips lighting holding b.v., http://www.lighting.philips.com/main/support/support/dialux-and-other-downloads.html, September 2016.

[116] Cree. Cree europe s.r.l a s.u., http://www.cree-europe.com/en/doc-tecnica-dbf.php, September 2016.

[117] F. Glover and M. Laguna. *Tabu Search.* Kluwer Academic Publishers, Norwell, MA, USA, 1997.

[118] The MathWorks Inc. Ga solver. `http://www.mathworks.com/help/gads/ga.html`.

[119] British Standards Institute Staff, British Standards Institution, and European Committee for Standardization. CPL/34/10. *Light and Lighting - Lighting of Work Places: Indoor work places. Part 1.* BSI, 2011.

[120] G. C. Canavos. *Applied probability and statistical methods.* Little, Brown, 1984.

[121] Mallba library. `http://neo.lcc.uma.es/mallba/easy-mallba/html/mallba.html`.

[122] K. Wong, C. Wu, R. K. P. Mok, C. Peng, and Z. Zhang. Evolutionary multimodal optimization using the principle of locality. *Inf. Sci.*, 194:138–170, 2012.

Apéndice

Técnicas de evaluación paralela para la metaheurística VNS

A.1. Introducción

En este anexo se estudia la aceleración de la resolución de problemas ILP en el caso de utilizar una aproximación de bajo rango de la matriz de radiosidad, y la búsqueda del óptimo por medio de la utilización del método VNS. En este contexto, el cálculo de la radiosidad es realizado a partir del producto secuencial del vector que está siendo evaluado con la matriz que contiene los datos de la escena, obteniéndose así un candidato a solución del problema, $B_i = \mathbf{M}E_i$ (donde $\mathbf{M} = (\mathbf{I} - \mathbf{RF})^{-1}$). Por lo que se estudiará el hecho de que dicha secuencia de productos puede ser reducida a una única multiplicación matriz-matriz $\mathbf{B} = \mathbf{ME}$, donde $\mathbf{B} = (B_1 \quad B_2 \quad ... \quad B_n)$, $\mathbf{E} = (E_1 \quad E_2 \quad ... \quad E_n)$. Si bien la cantidad de operaciones de punto flotante resulta ser la misma en ambos casos, se tiene que el producto matriz-vector es más lento debido a que la tasa de transferencia entre la memoria y CPU degrada el desempeño general del cálculo. Este tipo de aceleración es útil cuando es necesario realizar los cálculos en un computador solamente, sin tener acceso a GPUs o a un clúster de ordenadores para realizar los cálculos de forma paralela. Por medio de esta aceleración se logra aprovechar mejor los recursos de CPU, memoria y cache en beneficio de los tiempos de respuesta del algoritmo.

También se estudiará la aplicación de la técnica de Clearing (Petrowski [112]) para mejorar la diversidad en las soluciones obtenidas. Esto es interesante para brindar variedad de buenas y diferentes soluciones a los diseñadores.

A.2. Trabajo relacionado

Un ejemplo de aplicación de estas técnicas para la mejora de los resultados obtenidos por el algoritmo VNS es el realizado por T. Davidović and T. Craini [108] donde proponen varias estrategias de paralelización, utilizando una arquitectura multiprocesador, logrando mejorar tanto la calidad de los resultados como los tiempos totales de ejecución de los algoritmos, en comparación a un VNS sin técnicas de paralelización aplicadas.

Por otro lado la optimización multimodal es el conjunto de técnicas enfocadas en la búsqueda de múltiples óptimos globales y locales en oposición a la búsqueda de un único óptimo global. Su motivación se basa en que el conocimiento de varios óptimos. El conocimiento de múltiples buenas soluciones (óptimos locales o globales) es útil en

ingeniería, especialmente cuando el costo o restricciones de obtener el óptimo global lo hacen inviable. En dicho escenario, si se conocen múltiples soluciones (local y o globalmente óptimas), la implementación puede cambiarse rápidamente a otra solución y obtener el mejor rendimiento posible [122]. Mediante las técnicas de Algoritmos Evolutivos es posible obtener múltiples soluciones mediante la simulación de una población, en oposición a los métodos convencionales que requieren reiniciar la búsqueda o realizar múltiples ejecuciones. Las técnicas de optimización evolutiva, conocidas como técnicas de "*niching*" (nicho), han sido desarrolladas en las últimas décadas para la localización de óptimos locales y globales. Dichas técnicas pueden ser incorporadas a Algoritmos Evolutivos para lograr mantener múltiples subpoblaciones dentro de una población objetivo logrando así localizar múltiples óptimos globales o locales de manera simultánea. Para realizar la búsqueda multimodal de soluciones, las técnicas de nicho conducen la búsqueda a distintas áreas en el espacio de búsqueda, con el objetivo de que estas converjan a distintos picos (óptimos) simultáneamente. Como resultado se mantienen diversas subpoblaciones dentro de la población general [111]. Las técnicas de nicho pueden ser clasificadas según las siguientes cuatro categorías:

1. Secuencial: aquellas que localizan los nichos de manera iterativa.

2. Paralela: aquellas que localizan todos los nichos de forma paralela.

3. Cuasi secuencial: aquellas que localizan los nichos de manera secuencial y realizan la búsqueda de nuevos nichos de forma paralela mientras que los actuales son mantenidos.

4. Jerárquica: es una versión híbrida entre las técnicas secuencial y paralela diseñada para sobreponer las limitantes de la primera.

Aquí analizamos el uso de estas técnicas a los resultados de una población retornada por la ejecución paralela de múltiples VNSs.

En este anexo también se utiliza la técnica de Clearing, introducida por Petrowski [112] para mantener la diversidad de las configuraciones de iluminación obtenidas. Esto permite ofrecer un conjunto de buenas y diferentes soluciones, que cumplen las LIs. Esta técnica se basa en el concepto de la competencia por la obtención de recursos limitados de un entorno, por una subpoblación dada, eliminando así dentro de cada nicho los individuos menos aptos.

El estudio realizado en este Anexo es parte del trabajo realizado en Decia [109]. Aquí solo se reportan los resultados obtenidos en la ejecución en CPU.

A.3. Propuestas de optimización

En esta sección se describen las propuestas de ejecución paralela que serán evaluadas en el resto del anexo. Estas técnicas buscan aprovechar el uso del producto matriz-matriz, antes mencionado, con el objetivo de aumentar el desempeño computacional de la metaheurística VNS. Las distintas propuestas explotan distintas formas de agrupar en una matriz los vectores de emisión, en distintos pasos del algoritmo VNS.

A.3.1. Múltiples ejecuciones de la metaheurística VNS (N-VNS)

En la propuesta de evaluación paralela de N instancias de VNS (algoritmo que llamaremos N-VNS de aquí en más) se convierte el producto matriz-vector en un producto matriz-matriz. La nueva matriz armada contiene el vector, para el paso actual, de cada una de las N instancias de la metaheurística VNS que se están evaluando en paralelo. Por lo tanto en esta propuesta se realiza el cálculo de N ejecuciones de la metaheurística VNS que corren sus pasos al mismo tiempo, donde en cada producto matriz-matriz se calculan los resultados de la evaluación de un miembro del vecindario para cada una de las instancias del VNS que se calculan en simultáneo (Figura 1).

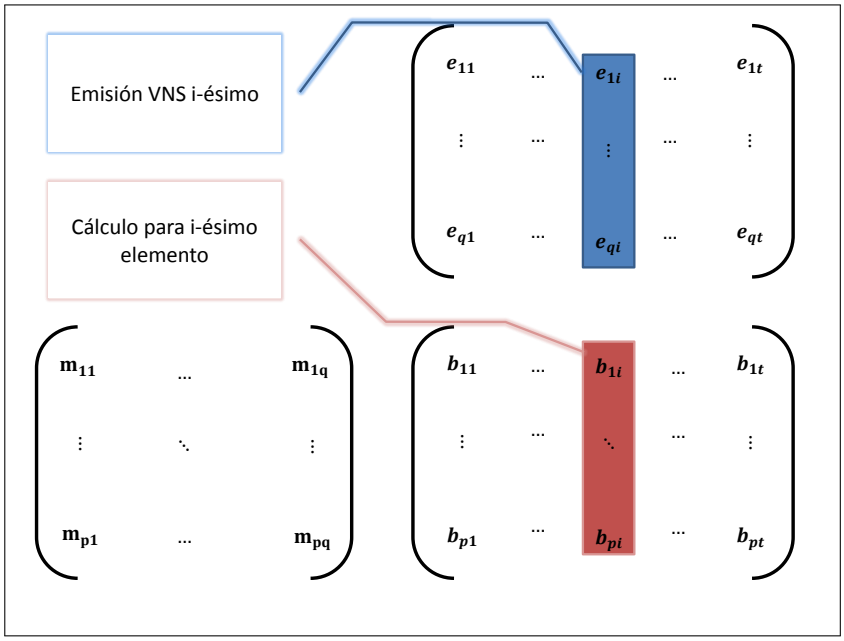


Figura 1: Imagen mostrando agrupación de emisiones para algoritmo N-VNS. Cálculo de emisión total de la escena en el algoritmo que corre varias instancias del VNS que se calculan en simultáneo. Cada i-ésimo vector corresponde al i-ésimo VNS. La matriz **M** es fija, no varía durante todo el proceso de cálculo.

Este tipo de evaluación puede resultar interesante ya que el método puede no llegar siempre a la solución óptima. Dependerá de la cantidad de ejecuciones. Es posible que se tengan varios óptimos o que estos sean numéricamente cercanos pero muy diferentes en cuanto a su resultado, es decir la configuración de la iluminación será tan buena numérica o matemáticamente pero a su vez diferente en cuanto a la iluminación producida o cómo es esta producida (dando mayor cantidad de soluciones al diseñador el cual tendrá mayor flexibilidad a la hora de seleccionar la solución a utilizar). Es importante comprender que no siempre es bueno conocer la solución óptima sino que puede ser útil conocer un conjunto de buenas soluciones. Por lo tanto, acelerar el proceso de ejecución de varias instancias del algoritmo que corran con distintos estados iniciales puede resultar muy interesante en la práctica.

A.3.2. Múltiple cálculo de vecinos (VNS-EV)

La metaheurística VNS se caracteriza por realizar la exploración de un vecindario seleccionando un grupo reducido de candidatos al azar dentro del mismo. Debido a esto, es interesante estudiar los beneficios de realizar el cálculo de todos los miembros seleccionados del vecindario en un solo producto matriz-matriz. Llamamos a este algoritmo VNS-EV de aquí en más. Por lo tanto en esta propuesta se toman los datos de todos los miembros (seleccionados) del vecindario y luego se calcula el producto como se muestra en la Figura 2. Una vez obtenidos los resultados estos se evalúan buscando entre ellos un resultado mejor. Cabe destacar que en este caso es conveniente realizar un cambio de comportamiento en el algoritmo VNS, debido a que en la forma secuencial, al encontrarse un valor mejor que el actual, el VNS se detiene (terminando la búsqueda en el vecindario actual y reiniciando en el vecindario inicial). Sin embargo, esta nueva propuesta termina de analizar todo el vecindario ya que el cálculo más costoso (el producto matriz-matriz) ya fue realizado, y buscar el menor valor del conjunto resultante es, en comparación, computacionalmente despreciable. Además, dado que calcula todas las soluciones, este proceso podría aumentar las posibilidades de encontrar una mejor solución.

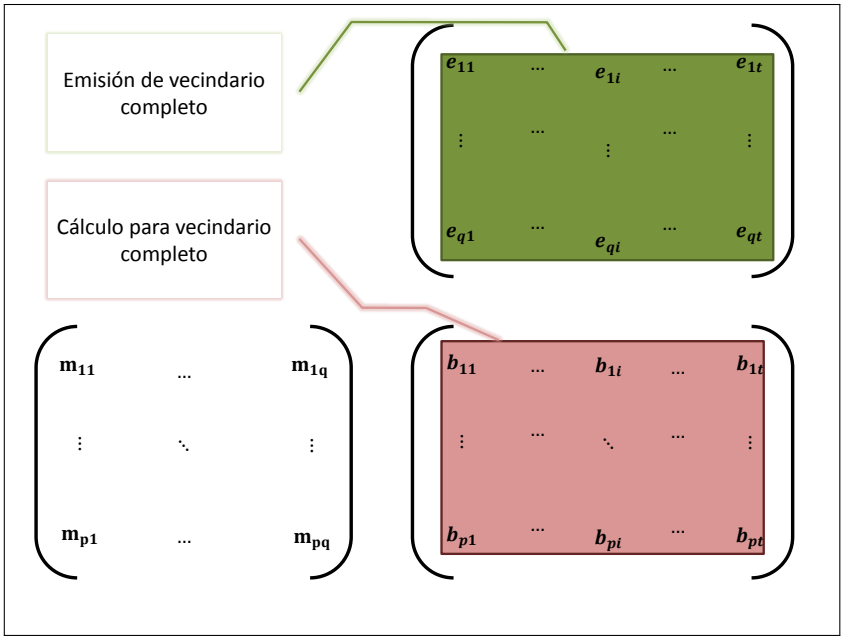


Figura 2: Imagen mostrando agrupación de emisiones para algoritmo VNS-EV. Cálculo de emisión de la escena en algoritmo que calcula todos los elementos de un vecindario utilizando el producto matriz-matriz, cada i-ésimo vector representa un miembro del vecindario que se está evaluando.

A.3.3. Múltiples ejecuciones de VNS con múltiple cálculo de vecinos (N-VNS+EV)

Una última propuesta que resulta interesante explorar es aquella donde se combinen ambas técnicas anteriores. Por lo tanto se plantea la realización de un tercer algoritmo que aplique en cada producto matriz-matriz los pasos de N-VNS que se calculan en simultáneo y que para cada uno de estos calcule además todos los elementos del vecindario que esté analizando, ver Figura 3. Llamamos a esta técnica N-VNS+EV.

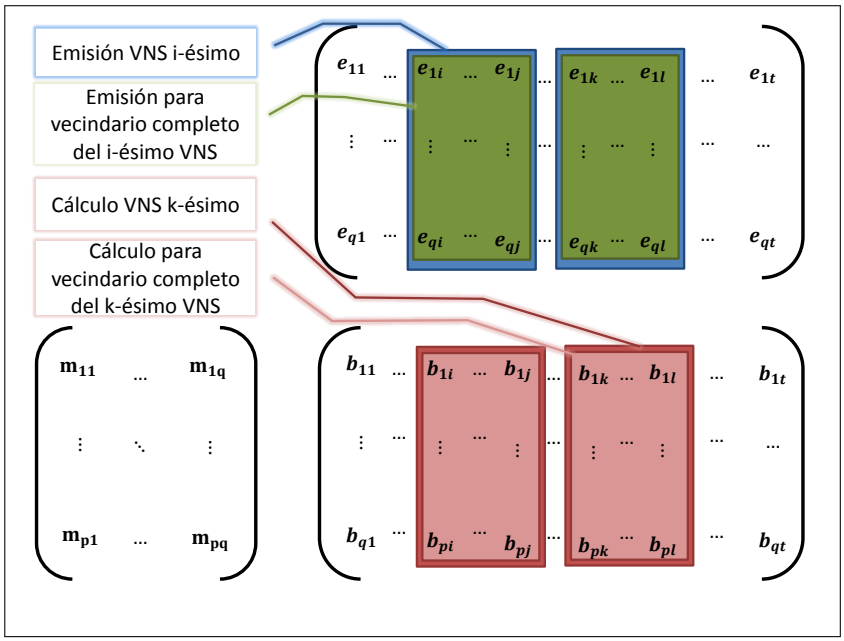


Figura 3: Imagen mostrando agrupación de emisiones para algoritmo N-VNS+EV. Cálculo de emisión total de la escena en algoritmo que calcula todos los elementos de un vecindario y a su vez múltiples VNS que se calculan en simultáneo utilizando un único producto matriz-matriz. Cada elemento resultado del i-ésimo VNS es una matriz que contiene un vector por cada miembro del vecindario que esté siendo evaluado.

A.4. Evaluación Experimental

En esta sección se describen y analizan los resultados obtenidos en la aplicación de las propuestas comentadas anteriormente, mostrando los tiempos de ejecución de cada algoritmo, calculando el aumento de velocidad obtenido (o *speedup* en inglés, que es calculado como $S_N = T_1/T_n$), y comparando frente al algoritmo de VNS simple que realiza el cálculo utilizando productos matriz-vector (que llamaremos de aquí en más VNS simple). Además se realizará una evaluación de los resultados obtenidos en la optimización para cada uno de los algoritmos.

Todos los algoritmos en los experimentos realizados realizan el cálculo de la iluminación utilizando emisores lambertianos cuya emisión es determinada por el conjunto de parches el algoritmo seleccione.

A.4.1. Escena, datos y hardware utilizados, restricciones y objetivos

El hardware utilizado para realizar la evaluación es el siguiente: Intel(R) Core(TM) i7-3770 CPU @ 3.40 Ghz y 16GB RAM. La escena elegida para realizar la evaluación es la Cornell Box [10], utilizada para el análisis de renderización. La representación de dicha escena contiene 9216 polígonos, ver Figura 4. Para la misma se tiene como objetivo la búsqueda de dos conjuntos de parches emisores que causen la mayor iluminación posible, o sea que la sumatoria de la iluminación recibida en cada parche (o polígono) de la escena sea lo mayor posible, y que se cumpla con las restricciones siguientes:

- El área de la suma de cada uno de los parches en cada conjunto emisor (que iluminan la escena) no es mayor a 0,5m.

- Los parches de cada conjunto emisor (que iluminan la escena) están contenidos en el plano horizontal superior (en el techo) de la escena.

En base a este conjunto de objetivos y restricciones se hace el análisis de los resultados.

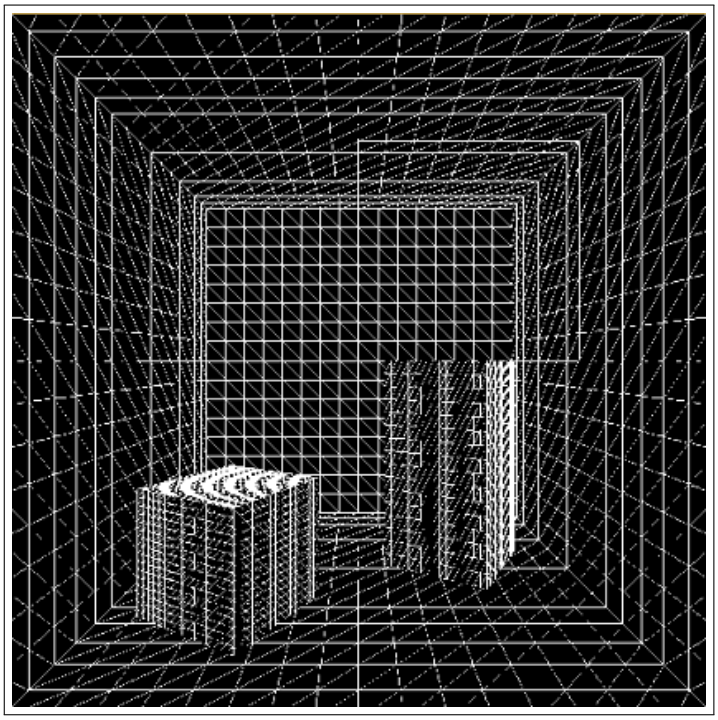


Figura 4: Escena utilizada para experimentos sobre técnicas de evaluación paralela para la metaheurística VNS. Cornell Box 9216 polígonos.

A.4.2. Múltiples ejecuciones de VNS (N-VNS)

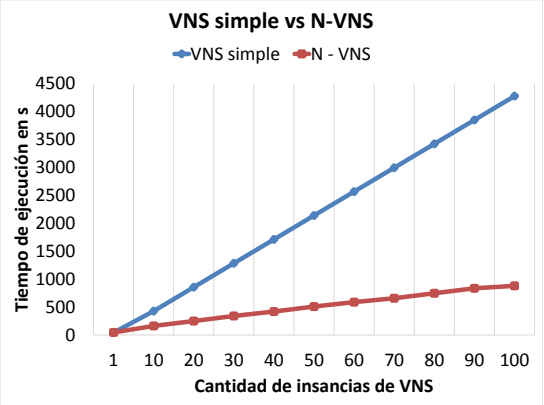
Para el caso de la ejecución en simultáneo de N instancias del algoritmo VNS se obtuvieron los datos presentados en la Tabla 1. Donde se puede notar que el algoritmo que ejecuta el cálculo de N-VNS como un producto matriz-matriz comienza con un aumento de velocidad de 2,6 con respecto al algoritmo simple si se lo corriese N veces en el caso de 10 instancias en paralelo, y que a medida que se incrementa la cantidad de VNS a calcular el aumento de velocidad tiende a mejorar, alcanzando el valor 4,8 para 100 instancias. La mejora del tiempo de ejecución se puede notar fácilmente en el gráfico presentado en la Figura 5(a), el aumento de velocidad obtenido puede ser consultado en la Figura 5(b), acercándose a 5 cuando 100 instancias de VNS son calculadas en simultáneo, tendiendo a desacelararse y estancarse en un valor cercano a 5.

A.4.3. Múltiple cálculo de vecinos (VNS-EV)

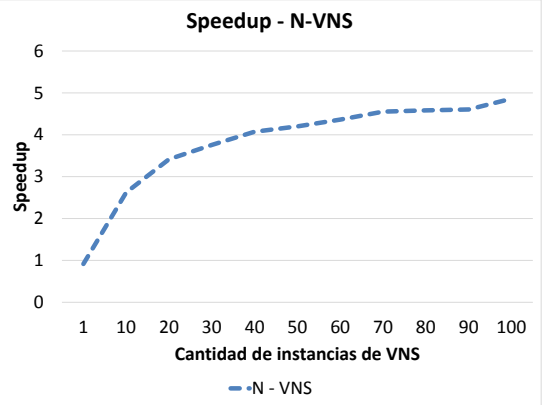
Para el caso de la ejecución de todo el vecindario en un solo producto matriz-matriz se obtuvieron los datos presentados en la Tabla 2. Donde se puede notar que el algoritmo que ejecuta el cálculo del vecindario completo comienza con un aumento de velocidad de 2,6 frente al algoritmo simple, y que a medida que se aumenta el tamaño del vecindario a

Cantidad de VNS	1	10	20	40	60	80	100
VNS simple	42,8 s	428s	856s	1712s	2568s	3424s	4280s
N-VNS	46,8 s	163,2 s	250,6 s	420,3 s	588,8 s	747,3 s	880,4 s
Speedup	0,9	2,6	3,4	4,1	4,4	4,6	4,9

Tabla 1: Datos generados para algoritmo simple y N-VNS que se calculan en simultáneo, tomando 15000 iteraciones.



(a) Tiempo de ejecución de algoritmo simple de VNS comparado con N instancias de VNS utilizando producto matriz-matriz.



(b) Aumento de velocidad calculado para la ejecución de N ejecuciones de VNS en simultáneo utilizando producto matriz-matriz en comparación con algoritmo simple ejecutado N cantidad de veces.

Figura 5: Gráficos con tiempos de los algoritmos de VNS simple y N-VNS, y aumento de velocidad del algoritmo N-VNS frente al VNS simple.

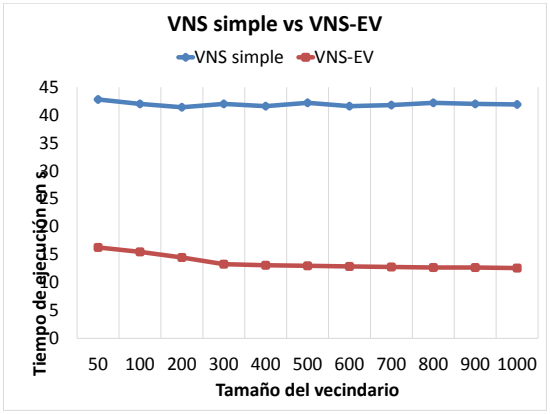
analizar se tiende a consumir una porción del tiempo menor, alcanzando un aumento de velocidad de 3,3 para un vecindario inicial con 1000 estados. Esto mismo se ve reflejado en el gráfico presentado en la Figura 6(a). Por otro lado el aumento de velocidad obtenido puede ser consultado en la Figura 6(b), acercándose a un aumento de velocidad de 3,5 con vecindarios de tamaño inicial igual a 1000 y donde se puede notar que el aumento de velocidad se estanca en un valor cercano a 3,3.

Tamaño del vecindario	50	100	200	400	600	800	1000
VNS simple	42,8 s	42,0 s	41,4 s	41,6 s	41,6 s	42,2 s	41,9 s
VNS-EV	16,3 s	15,5 s	14,5 s	13,1 s	12,9 s	12,7 s	12,6 s
Speedup	2,6	2,7	2,9	3,2	3,2	3,3	3,3

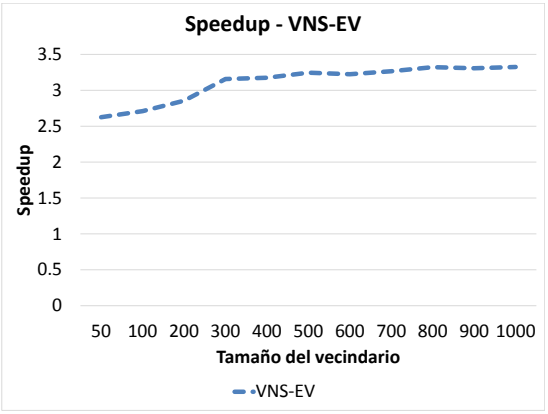
Tabla 2: Datos generados para algoritmo simple y VNS-EV, tomando 15000 iteraciones.

A.4.4. Múltiples ejecuciones de VNS con múltiple cálculo de vecinos (N-VNS+EV)

Para este último caso es conveniente realizar la comparación frente al algoritmo que calcula múltiples ejecuciones de VNS (N-VNS) y no, como en los análisis anteriores, contra el algoritmo simple. Esto es debido a que se intenta evaluar cuánto mejora el tiempo de ejecución del mismo al agregar el cálculo simultáneo de todo el vecindario.



(a) Aumento de velocidad calculado para el cálculo del algoritmo N-VNS en comparación con algoritmo simple ejecutado N cantidad de veces.



(b) Aumento de velocidad calculado para el cálculo completo de todos los elementos del vecindario seleccionados en un solo producto matriz-matriz.

Figura 6: Gráficos con tiempos de los algoritmos de VNS simple y VNS-EV, y aumento de velocidad del algoritmo VNS-EV frente al VNS simple.

Por lo tanto para esta evaluación se corrió el algoritmo mezclando distintos tipos de tamaño de vecindario para distinta cantidad de instancias de VNS, ver Tabla 3.

Algoritmo\Cantidad de VNS	1	10	20	30
N-VNS	46,8 s	163,2 s	250,6 s	341,7 s
N-VNS+EV (50)	16,5 s	130,9 s	261,1 s	409,7 s
N-VNS+EV (250)	**10,1 s**	**94,8 s**	**194,3 s**	**288,3 s**
N-VNS+EV (500)	10,3 s	99,1 s	198,0 s	296,6 s
N-VNS+EV (750)	11,0 s	106,0 s	207,2 s	318,4 s
N-VNS+EV (1000)	11,0 s	108,1 s	211,9 s	332,1 s

Tabla 3: Datos generados para algoritmo simple y N-VNS con exploración de vecindario. Los mejores resultados son resaltados con letra negrita.

De estos datos se puede notar que la nueva implementación es más eficiente en todos los casos evaluados menos para el caso en que se corre con un tamaño de vecindario de 50, en este caso solo es más eficiente cuando se tienen menos de 20 cálculos de VNS en simultáneo, esto se puede observar fácilmente en la Figura 7.

Lo interesante a destacar de estos resultados es que el tiempo de ejecución no mejora si se amplía el tamaño del vecindario sino que podemos notar la existencia de un punto de inflexión donde si se amplía el tamaño del vecindario el algoritmo demora más tiempo en computar, lo cual no parece ser consistente con los resultados de la evaluación de la exploración de vecindario mostrados anteriormente. Esto es debido a que en determinado punto el beneficio que se obtiene de la eficiencia del producto matriz-matriz disminuye y la ineficiencia introducida por el código del algoritmo aumenta causando peores resultados. En la Figura 7 se puede notar que la serie para el vecindario de tamaño 250 se mantiene por debajo de todas las demás, por lo tanto es el que resulta dar un mejor incremento de performance de todas los vecindarios que fueron analizados. Por otro lado en la Figura 9 se puede notar cómo varía el tiempo de ejecución pero tomando como variable el tamaño del vecindario, donde se puede ver que a medida que aumenta la

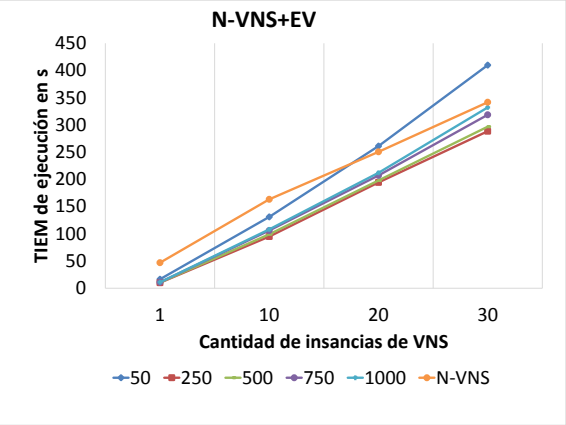


Figura 7: Gráfico con tiempo de ejecución de algoritmo N-VNS y N-VNS+EV para múltiples cantidades de VNS en simultáneo y múltiples tamaños de vecindarios. Cada serie representa la ejecución de un algoritmo con el parámetro de tamaño de vecindario fijo (en el caso de N-VNS el tamaño es fijo en 50 pero este no afecta de ninguna manera el tiempo de ejecución en dicho algoritmo).

cantidad de VNSs que son calculados en simultáneo, el aumento de velocidad obtenido es menor.

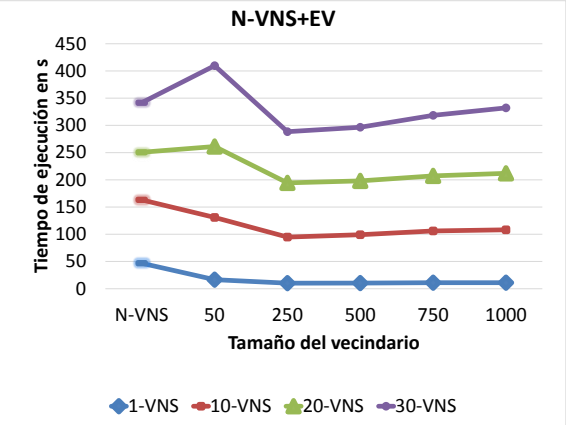


Figura 8: Gráfico de tiempo de ejecución de algoritmo VNS-EV y ejecución múltiple variando el tamaño de los vecindarios explorados. En cada serie se tiene la misma cantidad de VNS ejecutados al mismo tiempo (1, 10, 20 y 30) y su primer valor es el tiempo obtenido para el algoritmo N-VNS (que se intenta mejorar).

El aumento de velocidad obtenido se puede consultar en la Figura 10, donde es posible ver que la ganancia en el mejor de los casos (con tamaño 250 de vecindario) decrementa a medida que se aumenta la cantidad de instancias de VNS a calcular tendiendo a 1, o sea a la misma velocidad que el algoritmo N-VNS, y que si se aumenta la cantidad de instancias de VNS a ejecutar incluso sucedería que el aumento de velocidad sería inferior a 1 o sea sería menos eficiente que el original.

A.4.5. Evaluación de resultados de la optimización

Resulta interesante realizar un análisis de los resultados obtenidos por cada algoritmo pero esta vez en cuanto a la calidad de los resultados, viendo si los cambios aplicados

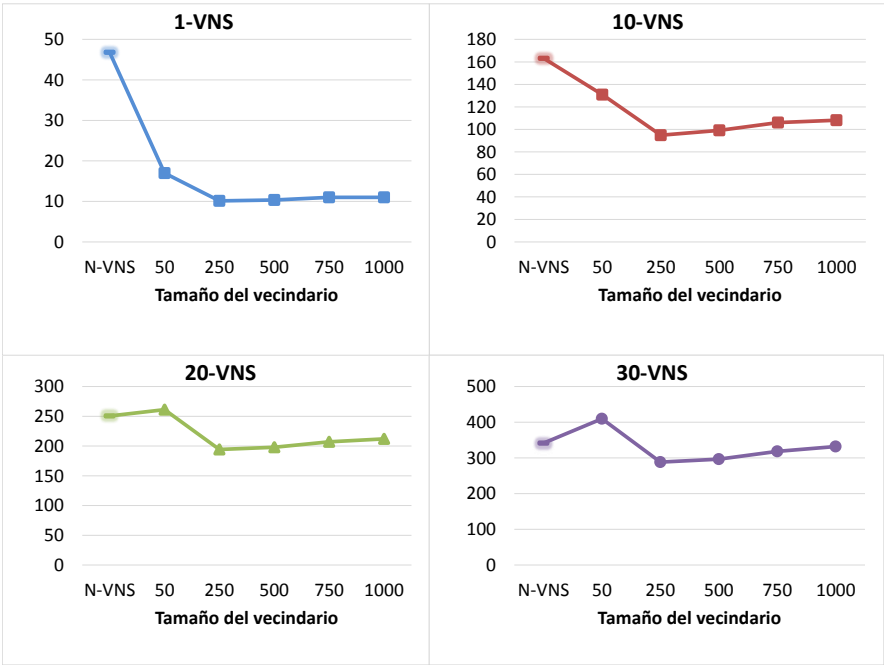


Figura 9: Tiempo de ejecución calculado para algoritmo N-VNS+EV. Cada gráfico representa una serie de la Figura 8 y el punto inicial representa el tiempo obtenido para el algoritmo que ejecuta varios VNS en simultáneo.

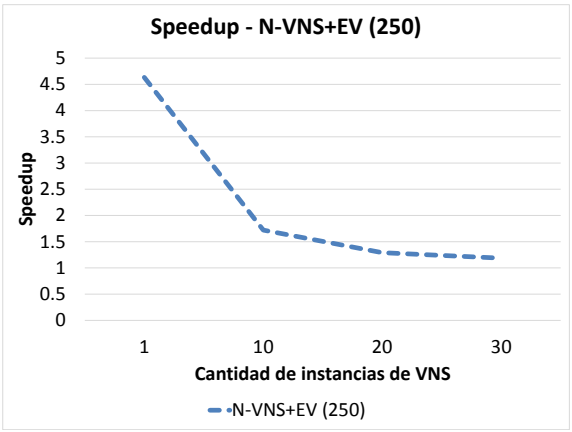


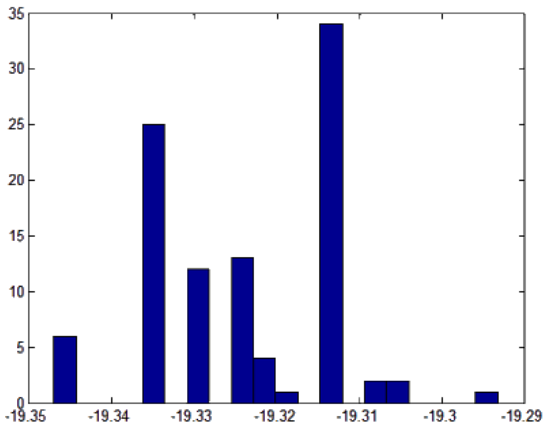
Figura 10: Aumento de velocidad calculado para las ejecuciones del algoritmo N-VNS+EV variando el tamaño de los vecindarios explorados y la cantidad e instancias de VNS a ejecutar. Los datos utilizados para construir este gráfico es la fila N-VNS+EV (250), en negrita, de la Tabla 3. En esta figura es posible ver mejor el comportamiento de cada serie en particular.

a estos pueden resultar en mejoras o en el deterioro en cuanto a los resultados finales obtenidos luego de la optimización. Por esto se analizó el resultado de la ejecución de 100 instancias distintas para cada algoritmo, o sea se obtuvieron 100 óptimos a partir de la ejecución de cada uno de los algoritmos y se calcularon los histogramas y las medidas de media y desviación estándar. Los resultados de estos datos se pueden ver en la Tabla 4. En la Figura 11 se ven los histogramas con los resultados de cada uno de lo algoritmos.

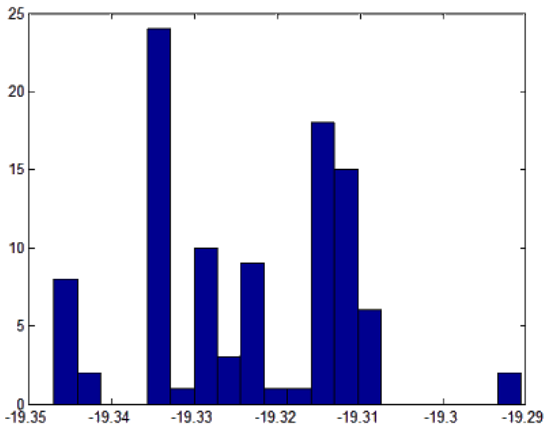
Algoritmo	μ (lx)	σ (lx)
VNS simple	19,3240	$1,09 * 10^{-2}$
N-VNS	19,3240	$1,23 * 10^{-2}$
VNS-EV	19,3204	$6,73 * 10^{-3}$
N-VNS + EV	19,3252	$1,35 * 10^{-2}$

Tabla 4: Datos generados para 100 optimizaciones realizadas con cada uno de los algoritmos; simple, N-VNS, VNS-EV y N-VNS+EV. Todos los algoritmos fueron probados con tamaño de vecindario de 250. Se muestra la media (μ) y la desviación estándar (σ).

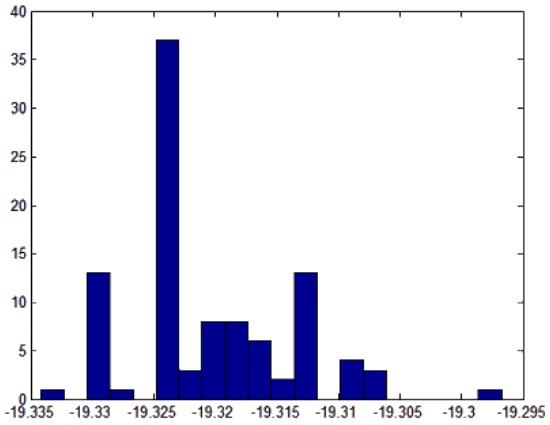
Analizando los resultados obtenidos, la diferencia en la media se encuentra en el orden de 10^{-2}. Por lo tanto, los resultados obtenidos por todos los algoritmos son similares, mientras que los resultados de la desviación estándar retornan resultados semejantes, donde la diferencia se encuentra en el orden de 10^{-2}. El caso del algoritmo N-VNS+EV es el que obtiene la mejor media en sus resultados.



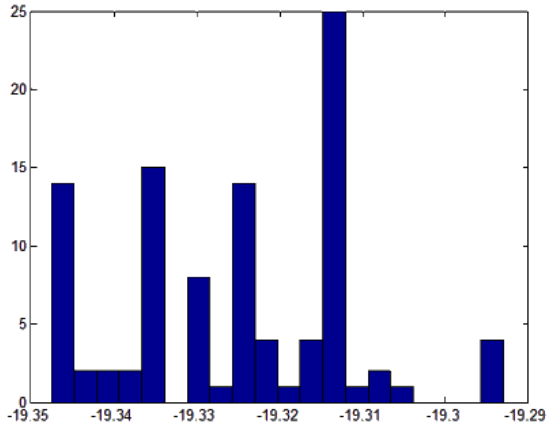
(a) 100 instancias de VNS calculadas con el algoritmo VNS simple.



(b) 100 instancias de VNS calculadas con el algoritmo N-VNS.



(c) 100 instancias de VNS calculadas con el algoritmo VNS-EV.



(d) 100 instancias de VNS calculadas con el algoritmo N-VNS+EV.

Figura 11: Histograma con los resultados para cada algoritmo. Los valores son negativos ya que se está maximizando por medio del minimizado del opuesto.

A.5. Mejora en la diversidad de los resultados

Si bien de la Sección A.4 podemos determinar que los algoritmos obtienen la misma calidad de resultados mejorando los tiempos de ejecución, estos no toman en cuenta la variedad de resultados. En otras palabras la ejecución paralela de los algoritmos lleva a que ciertas soluciones obtenidas sean similares, o sea no se controla la variedad de los resultados. El inconveniente de este comportamiento es el posible desperdicio de tiempo de cómputo debido al procesamiento de varias soluciones que no son de mayor utilidad ya que son iguales o similares. Por esto resulta interesante la implementación de técnicas que permitan utilizar la información de todas las instancias de VNS para así controlar la variedad de soluciones obtenidas.

El método seleccionado para realizar lo comentado es Clearing [112] y consiste en realizar la competencia entre los individuos de las últimas dos generaciones, creando una nueva generación que mantiene únicamente los individuos que tengan mayor fitness dentro de cada nicho. Luego, si la cantidad de individuos correspondientes a la nueva

generación es menor que la cantidad de instancias de VNS que están corriendo en simultáneo, se crean nuevas instancias de VNS que comienzan la búsqueda desde el inicio en áreas del espacio de búsqueda seleccionadas aleatoriamente. Por lo tanto, se obtiene una nueva generación que cumple lo siguiente:

- Existe únicamente un individuo por nicho.
- Cada individuo es el de mayor fitness para su nicho.
- Se continúa con el algoritmo VNS de los individuos que pertenecen a la última generación, pero se reinicia el vecindario de búsqueda.
- Se continúa con el algoritmo VNS de los individuos que pertenecen a la penúltima generación.
- Se reinician aquellas instancias de VNS para las cuales no exista ningún individuo seleccionado.

Debido al costo computacional que requiere esta técnica, esta se realiza una vez cada cierta cantidad de iteraciones.

La comparación de las soluciones se realiza en base a la radiosidad resultante. Por lo tanto, dos individuos pertenecen al mismo nicho si la distancia entre los vectores de radiosidad de estos es inferior al radio utilizado en el algoritmo.

A.5.1. Evaluación de resultados de la diversidad

El algoritmo que incluye la técnica de Clearing, que llamaremos N-VNS+C de aquí en más, toma 984,1s en comparación a los 880,4s necesarios para en ejecutar el algoritmo N-VNS por lo tanto la utilización del algoritmo no degrada demasiado el tiempo de ejecución del mismo. El Clearing fue aplicado una vez cada 4000 iteraciones y se realizaron 15000 iteraciones. Los resultados obtenidos para el algoritmo N-VNS+C se pueden consultar en la Tabla 5 y el histograma en la Figura 12.

Algoritmo	μ (lx)	σ (lx)
N-VNS+C	19,2949	$3,77 * 10^{-2}$

Tabla 5: Datos generados para 100 optimizaciones realizadas para el algoritmo N-VNS utilizando Clearing (N-VNS+C) una vez por cada 1/20 del espacio de iteraciones (sin incluir iteración inicial y final). Se muestra la media (μ) y la desviación estándar (σ).

Como se puede notar, comparando con los resultados mostrados en la Figura 11(b), la calidad de los resultados varía notoriamente con respecto a los obtenidos previamente por los algoritmos que no utilizan Clearing. Esto puede ser debido a que algunas instancias de VNS fueron reiniciadas en la última ejecución de la técnica de Clearing, por lo tanto no llegaron a buenas soluciones antes de completar la cantidad de iteraciones totales para el algoritmo. Por otro lado, el promedio de soluciones que pertenecen a un mismo nicho obtenido mediante la aplicación de esta técnica es de 3,39 para el algoritmo N-VNS+C mientras que se encuentra en el orden de 5,43 para el algoritmo N-VNS. Y el histograma para los resultados del algoritmo N-VNS+C muestan que se obtienen varios grupos con buenos resultados.

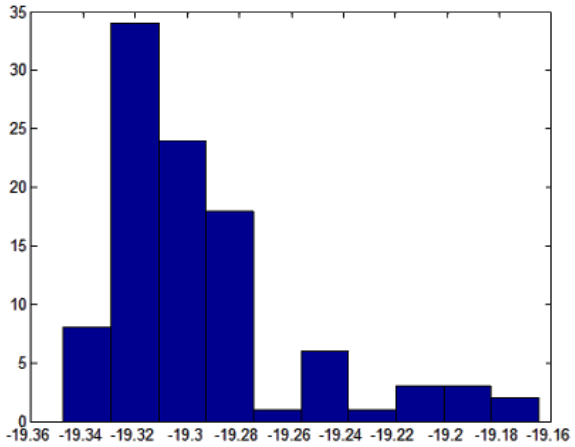


Figura 12: Histograma con el resultado de 100 instancias de N-VNS+C. En el eje de las abscisas se tiene la cantidad de iluminación recibida en la escena (medida en luxes y con valores negativos ya que se está buscando maximizar) y en el eje de las ordenadas se tiene la cantidad de veces que el resultado fue obtenido.

Cabe destacar nuevamente que el método de Clearing se aplica una vez cada cierta cantidad de iteraciones para no degradar la performance, ya que es un método costoso computacionalmente.

A.6. Conclusiones

A partir de los resultados obtenidos en la Sección A.4 podemos concluir que las tres propuestas de mejora de performance implementadas logran mejorar los tiempos de ejecución que se obtendría en caso de utilizar el producto matriz-vector, manteniendo a su vez la calidad de los resultados. Por otra parte, resulta claro que la aplicación de cada una de estas propuestas debe ser estudiada, ya que no existe una única solución que sea la mejor en todos los casos. Los beneficios que cada técnica brinda son bastante diferentes. Disponer de un conjunto de soluciones distintas pero con buenos valores puede resultar beneficioso para brindar mayores opciones al diseñador. Por otro lado evaluar los vecindarios en un solo producto matriz-matriz resulta beneficioso para acelerar el cálculo simple de un VNS. Luego la técnica de N-VNS+EV implementada resulta interesante como mejora a la técnica N-VNS, pero no mejora los resultados en todos los casos. Además a medida que se aumenta el tamaño de la matriz el código que guarda las estructuras de datos empieza a pesar por sobre la ganancia que se obtiene en el cálculo del producto matriz-matriz.

En cuanto a la Sección A.5, la aplicación del método de Clearing para mejorar la diversidad de los resultados obtenidos resulta de especial utilidad para lograr podar la ejecución de instancias de VNS que son similares, manteniendo siempre la solución con los mejores resultados obtenidos hasta el momento. De esta manera se logra optimizar la utilización de los recursos de cómputo en la búsqueda de nuevas y distintas soluciones.

En resumen:

1. Múltiples ejecuciones de VNS (N-VNS)
   - Notable aceleración comparado con ejecución secuencial, acercándose a un aumento de velocidad de 5 veces en el cálculo de 100 instancias en simultáneo.
   - Aplicación directa, no es necesario evaluar demasiado en qué casos se obtiene beneficios de su uso ya que aumentando la cantidad de VNS calculados, en un rango que aplica a situaciones reales, se obtienen mejores tiempos.
2. Múltiple cálculo de vecinos (VNS-EV)
   - Buena aceleración con respecto al cálculo secuencial de elementos del vecindario.
   - No corta la búsqueda en el vecindario cuando se encuentra un mejor valor sino que termina de evaluar el vecindario.
3. Múltiples ejecuciones de VNS con múltiple cálculo de vecinos (N-VNS+EV)
   - Aceptable aceleración respecto al algoritmo N-VNS.
   - El aumento de la cantidad de VNS evaluados al mismo tiempo reduce la aceleración.
4. Método de Clearing para búsqueda de diversidad (N-VNS-C)
   - Usa eficientemente los recursos de cómputo en la búsqueda de variedad en las soluciones, podando soluciones similares y posibilitando la búsqueda en nuevas regiones del espacio de búsqueda.