LOCAL SEARCH BASED OPTIMIZATION OF A SPATIAL LIGHT DISTRIBUTION MODEL

David Kaljun
Faculty of Mechanical Engineering
University of Ljubljana, Slovenia
david.kaljun@fs.uni-lj.si

Janez Žerovnik
Faculty of Mechanical Engineering
University of Ljubljana, Slovenia
and
Institute of Mathematics, Physics and Mechanics
Ljubljana, Slovenia
janez.zerovnik@fs.uni-lj.si

Abstract Recent development of LED technology enabled production of lighting systems with nearly arbitrary light distributions. A nontrivial engineering task is to design a lighting system or a combination of luminaries for a given target light distribution. Here we use heuristics for solving the problem restricted to symmetrical distributions. A genetic algorithm and several versions of local search heuristics are compared showing that practically useful approximations can be achieved with majority of the algorithms.

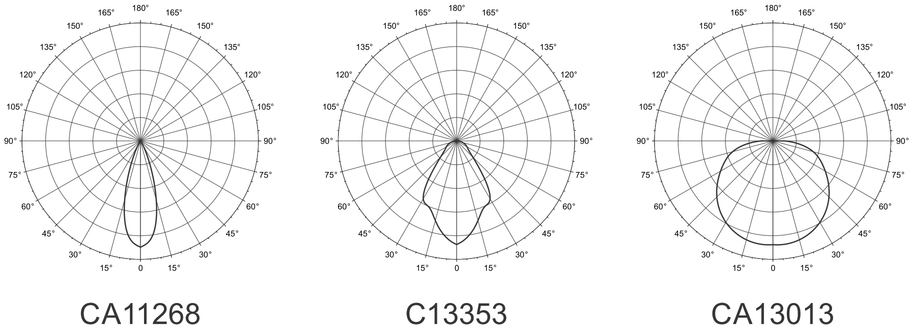
Keywords: Genetic algorithm, Light distribution model, Local search, Iterative improvement.

1. Introduction

Nowadays, technology of Light Emitting Diodes (LEDs) enables to lower the energy consumption of luminaires and to design more efficient lighting systems that make it possible to deliver the light to the environment in a controlled way. The many possible designs lead to new problems of choosing the optimal or at least a very good design depending on possibly different goals such as optimization of energy consumption, production cost, and, last but not least, the light pollution of the environment. Nowadays, in many cases trial and error method followed by simulation is used in practice. We believe that using analytical models and optimization tools may speed up the design and at the same time possibly improve the quality of solutions. Here we adopt an analytical model for a version of the general problem, and use heuristic methods based on the model to provide nearly optimal solutions. The heuristics used in this study are three versions of local search and a genetic algorithm. We also compute solutions provided by blind random search to avoid trivialities. The rest of the paper is organized as follows. In the next section we briefly explain the practical motivation for this research. Section 3 gives the analytical model and the optimization problem that is addressed in the paper. In Section 4, the algorithms are outlined. Experimental results are presented in Section 5. The paper ends with a summary of conclusions and idea for future work.

2. Motivation

Only a few years ago, emerging new technology of Light Emitting Diodes (LEDs) was in the first stage of implementation [1]. Meanwhile the demand to lower the energy consumption of luminaires and to build more efficient lighting systems that can deliver the light where needed has pushed the development of high power LEDs. Following the development of LEDs, many luminaire manufacturers developed LED luminaires as a replacement for the existing energy inefficient luminaires. Naturally, the use of LEDs introduces new and unique challenges to the development engineers. One of these challenges is to design and simulate an efficient light engine for the luminaire. The light engine consist of the source which in this case are LEDs, and the appropriate secondary optics. The choice of the secondary optics is the key in developing a good system. For designing a good system nowadays technology enables two options. Having the know-how and the resources, a specific lens to accomplish the task can be developed. However, the resources coupled with the development and production of optical elements may be enormous. Therefore a lot of manufactures are using the second option, that is to use readymade of the shelf lenses. There are specialized companies in the world that produce different type of lenses for all of the major brands of LEDs. The trick here is to choose the best combination of lenses to get the most efficient system. The current practice in development process is a trial and error procedure, where the developer chooses a combination of lenses, and then simulates the system via Monte Carlo



CA11268 C13353 CA13013

Figure 1. Modeled spatial light distribution presented in polar diagrams.

ray-tracing methods. The success heavily depends on the engineers' intuition and experience and still needs sizeable computation resources for checking the proposed design by simulation.

A natural avenue of research related to the second approach is to replace trial and error method by a more efficient design method based on analytical and algorithmic tools. For these aim, a theoretical framework is needed. Among the first known theoretical results is the analytical model [8] that was proposed for LEDs without secondary optics. Namely, the usual practical situation is that we have the target light distribution given by large dataset of points in the space with (desired or measured) light intensity. The idea of [8] that is at the same time already a part way towards the solution is to fit the data with suitable functions that in turn can provide a construction of a light engine which approximates the target light distribution. Later we will explain a modification of the analytical model of [8] that we use in our study where we successfully approximate symmetric spatial distributions. The general problem is much more challenging. Development of a useful analytical model for general case is to the best of our knowledge an open research problem. Having the target light distribution in a manageable form, the design of the light engine may be possible. Depending on the application, several questions/tasks are natural, for example: (1) design the engine having exactly (or, approximately) the target light distribution (2) design such engine using as few LEDs as possible (3) design such engine as cheap as possible. A combination of these goals may be of interest which in turn leads to a number of multicriteria optimization problem(s).

3. Analytical Model and Problem Definition

The fact that there are many different LEDs with different beam patterns and many different secondary optics to choose from indicates that providing a general analytical model for all of them is presumably a very challenging open research problem. Therefore in this study we restrict attention only to LED-lens combinations that have symmetrical spatial light distribution. In other words, the cross section of the surface which represents the spatial distribution with a section plain that is coincident with the vertical axis of the given coordinate system is alike at every azimuthal angle of offset. This enables us to define the analytical model in two dimensions, so it describes a curve rather a surface. To produce the desired surface, we just revolve the given curve around the central vertical axis with the full azimuthal angle of 360°. Three examples from our dataset are given in Figure 1. For the special case of symmetrical spatial light distribution, an analytic model for the radiation pattern of a single LED without the secondary optics was proposed in [8]. The author proposed two different models, one based on Gaussian and one on cosine-power function. Based on our preliminary manual test fittings of the models to measured data of three randomly chosen lenses from the dataset [5] we use here we have concluded that the cosine-power functions $I(\Theta) = a * \cos(\Theta - b)^c$ have slight advantage over the Gaussian ones. Another argument is that the cosine-power functions seem to be a more natural choice in this context because there are basic LEDs with simple secondary optics for which the light distribution can be approximated with a single cosine-power function. We therefore start with the analytical model from [8] using cosine functions:

$$M(\Theta) = \sum_i a_i * \cos(\Theta - b_i)^{c_i} \tag{1}$$

It was observed in [8] that a sum of only three cosine-power functions is sufficient in most cases. Our preliminary tests confirmed this observation, so we assumed that the sum of three cosine-power functions will probably be enough to fit LEDs with lenses that have symmetric radiation patterns with sufficient quality. In addition to the parameters of the original model, we introduce a normalizing parameter I_{max}, as this simplifies (unifies) the range of the other three parameters: $a = \{0, 0.001, 0.002, \dots, 1\}$, $b = \{-90, -89.9, -89.8, \dots, 90\}$ and $c = \{0, 1, 2, \dots, 100\}$, for all test lenses. Doing all of the above we have rewritten the definition (1) as follows:

$$I(\Theta) = I_{max} \sum_i a_i * \cos(\Theta - b_i)^{c_i} \tag{2}$$

The expression (2) thus represents our analytical model to fit LEDs with attached secondary optics and symmetric spatial light distribution.

The goodness of fit is as usual [8, 10] defined to be minimizing the root mean square error (RMS), formally defined as:

$$RMS = \sqrt{\frac{1}{M}\sum_i \left[I(\Theta_i)_m - I(\Theta_i)\right]^2} \qquad (3)$$

For a sufficiently accurate fit, the RMS value must be less than 5% [8, 10]. On the other hand, current standards and technology allows up to 2% noise in the measured data. Therefore, the target results of the fitting algorithms are at less than 5% RMS error, but at the same time there is no practical need for less than 1% or 2% RMS error.

We will assume that all the data are written in the form of vectors v = (polar angle [Θ], intensity [I]). In reality, measured photometric data from the lens manufacturers are available in one of the two standard coded formats. That are the IESNA photometric digital format *.ies [11] used primarily in the USA and the European format EULUMDAT *.ldt [2]. Conversion of the data in the two standard formats can easily be transformed into the list of vectors. In addition, due to the novel parameter I_{max} each dataset will be normalized during the preprocessing so that in each instance the maximal intensity of the vectors will be 1, and the normalizing value I_{max} is given as additional input value to the algorithms.

The problem can formally be written as:

INPUT: I_{max} and a list of vectors v= (polar angle [Θ], intensity [I])
TASK: Find parameters (a_1, b_1, c_1, a_2, b_2, c_2, a_3, b_3, c_3) that minimize the RMS error (3).

Different fitting algorithms were used to minimize RMS error. The algorithms are presented and compared in the next sections.

4. Fitting Algorithms

In this section we describe five fitting algorithms. As the main objective of this study was to obtain good solutions to the practical problem, the algorithms were chosen with this primary goal in sight. The selection is thus quite arbitrarily, and there may be some other algorithms or some other versions that might outperform the selected versions.

Note that here the problem is a continous optimization problem and hence, compared to discrete optimization, there are even more possibilities to define a neighborhood for the local search based heuristics. In fact, the neighborhoods we use can be seen as variable neighborhoods [7], although they are all similar. Of course, there may be other neighborhoods that would be worth consideration. The reason we keep the selected neighborhoods and did not try to look for other possibilities is simply the fact that they already gave us results of sufficient quality. Another natural question that may be asked here is why use discrete optimization heuristics on a continuous optimization problem. First, there is no analytical solution for MST best approximation of this type of functions, and second, in order to apply continuous optimization methods such as the Newton method, usually we need a good approximation in order to assure convergence. As the target RMS error is between 1% and 5%, the fine approximation based on continuous optimization methods could be used as postprocessing. On the other hand, in view of the at least 2% noise in the data, this postprocessing is not of practical interest in this case.

We have started our experiments with two basic local search algorithms, steepest descent and iterative improvement, where in both cases the neighborhoods were defined in the same way, explained in more detail below. We call this neighborhood fixed stepsize neighborhood. The third local search algorithm is a variation of iterative improvement where we introduce random step size; roughly speaking, given a step size and direction as before, we randomly make a step in the direction that is at most as long as in the fixed size neighborhood search. Naturally, whenever local search is used, the multi start version is worth consideration. As preliminary testing of multi start version was not competitive with single longer runs, therefore we decided to use a more advanced heuristics that would on one hand take advantage of the seemingly successful local search and possibly accumulate information obtained by independent local searches. Our choice was to use a genetic algorithm. Finally, we also run and compare results of a simple generation of random solutions.

4.1 Steepest Descent – SD

The algorithm begins with the initialization of the initial function parameter values that are $a_1 = a_2 = a_3 = 0.5$, $b_1 = b_2 = b_3 = 0$, and $c_1 = c_2 = c_3 = 1$. Next it initializes the search step values which are for $da = 0.01$, for $db = 1$ and for $dc = \frac{I_{max}}{10}$ giving the 512 neighbors of the initial solution: $(a_1 \pm da, b_1 \pm db, c_1 \pm dc,\ a_2 \pm da, b_2 \pm db, c_2 \pm dc, a_3 \pm da, b_3 \pm db, c_3 \pm dc)$. If there is a neighbor with better RMS value, the search moves to the neighbor with minimal RMS value (if there are more minimal neighbors, all are chosen with the same probability). If none of the 512 is better than the current solution a new set of neighboring solutions are generated, this time with a double step. It goes for ten steps and if there still is no better solution it breaks the search, multiplies the step value with 0.9, so the step is finer, and begins the search from start in the neighborhood of the current solution. The algorithm stops when the number of generated solutions reaches T_{max}.

4.2 Iterative Improvement – Fixed Neighborhood – IF

The algorithm uses the same neighborhood as SD. Instead of considering all 512 neighbors at once, the algorithm generates a neighbor randomly, and moves to the neighbor if its RMS value is better than the current RMS value. If no better neighbor is found after 1000 trials, it is assumed that no better neighbor exists. As above, the algorithm changes to a new neighborhood, this time with a double step. It goes for ten steps and if there is still no better solution, it breaks the search, multiplies the step value with 0.9, so the step is finer and begins search from the start in the neighborhood of the current solution. The algorithm stops when the number of generated solutions reaches T_{max}.

4.3 Iterative Improvement – Random Neighborhood – IR

The search begins as the previous two algorithms. It initializes the same initial function parameter values. Next it initializes the search step value within a range, rather than a static fixed value. The ranges are for $da_1 = da_2 = da_3 = \{-0.1, -0.099, -0.098, \ldots, 0.1\}$, for $db_1 = db_2 = db_3 = \{-9, -8.9, -8.8, \ldots, 9\}$ and $dc_1 = dc_2 = dc_3 = \{-10, -9, -8, \ldots, 10\}$. It begins generating solutions, using the step range around the initial solution and calculating their RMS error. As soon as it generates a better solution, it stops, shifts the focus on that solution, resets the step range to the initial value, and continues the search in the neighborhood of the new best solution. If after 400000 generated solutions no better is found, than the step range gets doubled, and the search continues in the current neighborhood with a larger neighborhood. The stopping condition are the same as before, whichever is achieved first stops the search.

4.4 Genetic Algorithm – GA

The genetic algorithm mimics the evolutionary behavior [4,6,9]. Three genetic operators are used. The natural selection [4] where the best in a population survive, mutation [9] where randomly chosen parameters of a surviving solution are changed producing a new solution, and crossover [4,6,9] or breading where a new solution is created by randomly combining and crossing parameters from two randomly chosen solutions that have survived from the previous generation.

The algorithm begins with the generation and calculation of one hundred and fifty thousand solutions for the zero population. It then chooses via the natural selection ten best solutions. These ten are then locally optimized with the algorithm that implements the iterative improvement with random neighborhood for sixty thousand iterations. After that it generates the next generation from the ten best solutions of the previous generation. It generates twenty-five thousand mutant solutions and twenty-five thousand crossover solutions. More precisely, the mutation operator works as follows: in the randomly chosen individual, one to nine parameters are chosen to be changed (mutated) by adding to the current parameter value a randomly chosen value for $da_1 = da_2 = da_3 = \{-0.01, -0.009, -0.008, \ldots, 0.01\}$, for $db_1 = db_2 = db_3 = \{-0.25, -0.24, -0.23, \ldots, 0.25\}$ and $dc_1 = dc_2 = dc_3 = \{-2.5, -2.4, -2.3, \ldots, 2.5\}$. The crossover operation takes two randomly chosen individuals and choses one to nine parameters to be changed. The new parameter values are generated by calculating the difference of the according parameter pair of the two individuals $(a_1 i a_2 i, b_1 i - b_2 i, c_1 i - c_2 i)$, randomly choosing a value larger than zero and smaller than the calculated difference and adding the chosen value to the smaller parameter value of the pair. Then it chooses the best ten from that generation and optimizes them. It compares the optimized solutions from the previous generation and the current one. Again it chooses the ten best from both. After that it begins with the generation of the next generation, following the same steps as before.

The stopping condition is based on the number of generated solutions as for other algorithms. In our experiment, the usual number of generated solutions was four million, which here means that the algorithm stops when it optimizes the best solutions of generation five.

4.5 Blind Random Search – RAN

T_{max} times generates random values of the parameters, evaluates the RMS error, and remembers the best so far solution.

5. Results

We discuss the results of a comparative experiment in which all the algorithms were run with $T_{max} = 4$ million. All algorithms were saving a log file during the runtime process, so we can extract the values at any particular time of the process.

Table 1. RMS error after 4 million calculating operations.

Lens/Algorithm	SD	IF	RAN	IR	GA
C13353	3.991	**3.408**	7.013	8.671	**3.061**
CA11265	2.775	**2.372**	4.936	4.798	**2.729**
CA11268	**2.227**	**2.229**	4.100	2.471	2.578
CA11483	**3.100**	**3.066**	4.130	3.387	3.141
CA11525	3.150	**1.108**	3.217	1.907	**1.087**
CA11934	3.940	**2.514**	4.196	3.543	**2.909**
CA12392	**1.636**	**1.641**	3.424	2.445	2.277
CA13013	1.202	**0.695**	2.136	2.241	**0.916**
CP12632	5.537	5.493	**4.918**	4.974	**4.362**
CP12633	2.431	**2.415**	4.063	3.708	**2.347**
CP12636	**2.348**	**2.107**	4.571	4.217	2.479
FP13030	**2.267**	**2.257**	3.762	3.659	2.414

For the purpose of the algorithm evaluation, we have chosen a set of real available lenses to be approximated. The set was chosen from the online catalogue of one of the biggest and most present manufacturer in the world Ledil Oy Finland [5]. The choosing from the broad spectrum of lenses in the catalogue was based on the decision that the used LED is Cree XP-E [3], and the demand that the lenses have a symmetric spatial light distribution. We have preserved the lens product codes from the catalogue, so the reader can find the lens by searching the catalogue for the code from the first column in table 1 .

5.1 Quality Comparison

In the Table 1 below the overall best solutions after the long runs of all algorithms on all twelve instances from the dataset are given. Recall that the results are acceptable if they have RMS values lover than 5% and that the approximation better than 1% is not of any use because of the noise in data. The best two results for each instance are in bold.

First, observe that all the algorithms in most of the cases give acceptable results, i.e. lower than 5 which is the same as 5% recalling the meaning of the normalizing parameter I_{max}. If we take a closer look, at the values we can see that the iterative improvement with fixed size IF is the winner when counting the number of best solutions, achieving the best solution in six out of twelve instances. The second best is the genetic algorithm with four best solutions, followed by the steepest

Table 2. RMS error after 750 thousand calculating operations.

Lens/Algorithm	SD	IF	RAN	IR	GA
C13353	6.617	**4.284**	9.252	9.909	**3.784**
CA11265	**3.,477**	**2.700**	7.282	5.073	4.183
CA11268	**2.376**	2.620	5.893	**2.471**	2.932
CA11483	4.181	**3.400**	4.130	3.784	**3.641**
CA11525	3.813	**3.395**	4.811	3.789	**1.601**
CA11934	4.032	**1.662**	4.988	**3.543**	3.789
CA12392	**1.814**	**1.661**	3.597	2.717	2.577
CA13013	2.804	3.115	**2.136**	2.241	**1.331**
CP12632	9.501	9.839	8.474	**5.054**	**4.703**
CP12633	**2.465**	4.511	4.757	4.296	**2.613**
CP12636	5.000	6.297	5.506	**4.217**	**3.803**
FP13030	**2.800**	5.679	6.611	3.659	**3.233**

descent with two. Second, comparing the three local search algorithms and the genetic algorithm in terms of the quality of their best solutions on particular instances, we see that all best solutions are within 1%. We can conclude that all four are fairly comparable in terms of the expected quality of the solution. On the other hand, the blind random search on average does not produce as good results as the other four, however it may luckily guess good solutions, in one case even the best solution obtained (on instance CP12632).

As we have so many results of acceptable quality, a natural question is whether the time limit chosen above could be shortened. The long runs in our implementations took 30 minutes for every run on a Intel Core I3-4130 @ 3,5 Ghz, programmed in C++. (The code was not optimized). Therefore it is interesting to compare shorter runs, see Table 2.

The shorter runs again show that most algorithms achieve the 5% error bound already in short runs. It may be interesting to note that the genetic algorithm is the only one that in the short runs finds solutions under 5% bound for all instances. In addition, it is also the winner in eight out of twelve cases looking the best obtained solution. We also observe that in short runs, the two algorithms based on fixed size neighborhood outperform the random size neighborhood iterative improvement. As expected, blind random search is not competitive on average, however curiously it is the winner on one instance.

Finally, comparing the speed of convergence we observe that all of the algorithms have a very steep convergence curve, a typical example is given in Figure 2.

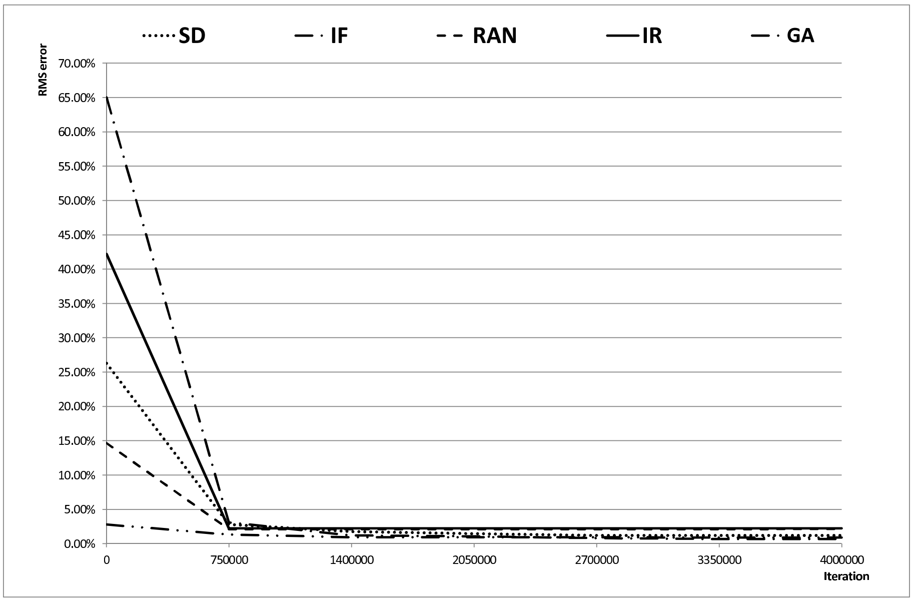


Figure 2. Linear interpolation of the approximation runtime process of CA13013 lens.

6. Conclusions

The goal of this part of the research was to design an efficient algorithm to fit an analytic model to the measured data of LED and secondary lens combination with symmetric spatial light distribution. We have designed several algorithms, and tested them on real lens data. The results of the test showed that, four of the algorithms produce approximations of acceptable quality. As the dataset used for testing includes a good variation of realistic LED and secondary lens combinations we can conclude that the practical approximation and design problem has been solved.

From theoretical viewpoint it is interesting to note that the genetic algorithm was very competitive, in short runs comparison even the best among the tested algorithms. Of course, it is well known that with fine tuning of parameters most of metaheuristics can be adopted to be very successful on a particular dataset. However, here we should add that our study started with testing the neighborhoods for local search and only in the last part of the research we used the genetic algorithm. So in this case the parameter tuning of all algorithms took about the same effort. We thus believe that the comparison is fair also from this viewpoint.

The study presented here gave important information about the number of complexity of solving the general problem in the case of instances with symmetric spatial light distribution. Future work includes adaptation of the model to lenses with asymmetric spatial light distribution. Based on the new models, the heuristic data fitting that would lead to the construction of desired light engines, analogous to the work presented here may be possible. The general model will presumably include a larger number of parameters which in turn most probably means larger search spaces and more challenging optimization problems.

Acknowledge

This work is supported in part by ARRS, the research agency of Slovenia.

References

[1] Escaping the bulb culture: the future of LEDs in architectural illumination. *LEDs magazine*, (1):13–15, April 2005.

[2] I. Ashdown. Thinking Photometrically Part II. In *LIGHTFAIR 2001 Pre-Conference Workshop*, March 2001.

[3] Cree Inc. `http://www.cree.com/led-components-and-modules/products/xlamp/discrete-directional/xlamp-xpe`. Accessed 2014.

[4] R. L. Haupt and S. E. Haupt. *Practical Genetic Algorithms. 2nd Edition.* John Wiley & Sons, 2004.

[5] Ledil Oy. `http://www.ledil.com/`. Accessed 2014.

[6] M. Mitchell. *An Introduction to Genetic Algorithms. 5th Edition.* MIT Press, 1999.

[7] N. Mladenovic, P. Hansen, and J. Brimberg. Sequential clustering with radius and split criteria. *Cent. Eur. J. Oper. Res.*, 21(Suppl. 1):95–115, 2013.

[8] I. Moreno and C.-C. Sun. Modeling the radiation pattern of LEDs. *Opt. Express.*, 16(3):1808–1819, February 2008.

[9] D. Simon. *Evolutionary Optimization Algorithms.* John Wiley & Sons, 2013.

[10] C.-C. Sun, T.-X. Lee, S.-H. Ma, Y.-L. Lee, and S.-M. Huang. Precise optical modeling for led lighting verified by cross correlation in the midfield region. *Opt. Lett.*, 31:2193–2195, 2006.

[11] The Subcommittee on Photometry of the IESNA Computer Committee. IESNA standard file format for the electronic transfer of photometric data and related information. Technical Report ANSIDESNA LM-63-02, Illuminating Engineering Society of North America, 2002.