Chapter 8
Automatic Design of Optimal LED Street Lights

Balázs L. Lévai and Balázs Bánhelyi

Abstract The issue of light pollution, unnecessary lighting of outdoor areas, came into focus in the last 10 years. This is the reason why observatories should not be built in highly populated areas, it also disturbs the wild life, and it raises questions about energy conservation too. Based on its capabilities, LED technology offers a solution to this problem. Nowadays, travellers can visit many cities in developed countries and encounter LED street lights in streets as application of this technology spreading in public lighting. Designing orientation of LEDs in such street lights is a difficult problem as we need to use multiple LED packages to light an as large area as an incandescent light bulb can. Determining correct angles is a global optimization problem, a complex mathematical task related to the field of covering problems. In this chapter, we present an automatic designing method to construct LED configurations for street lights and a light pattern computation technique to evaluate these configurations. To speed up the whole designing process, a possible way of parallelization is also discussed.

Keywords Global optimization • Genetic algorithm • Covering problem • LED • Public lighting

8.1 Designing LED Street Lights

When we are looking at the view of a city at night, the first thought usually coming into our mind is how beautifully everything is lighted, admiring the luminous streets, buildings, and bridges. The last thing we realize is the price of shinning, the light pollution by name.

Nowadays, we reached a harmful level of light emission. It affects the wild life, especially the insects. Such a species as the fireflies, whose mating ritual essentially involves light signals, suffered a heavy drop in their numbers. Confused by artificial light, males and females cannot find each other. On a much global scale, light is

B.L. Lévai • B. Bánhelyi (✉)
Institute of Informatics, University of Szeged, 6701 Szeged, Hungary
e-mail: levaib@inf.u-szeged.hu; banhelyi@inf.u-szeged.hu

© Springer International Publishing Switzerland 2015

G. Fasano, J.D. Pintér (eds.), *Optimized Packings with Applications*, Springer Optimization and Its Applications 105, DOI 10.1007/978-3-319-18899-7_8

also an important factor in animals' navigation and migration. Not just the timing of human created light is the problem, but its polarization too, because many animals use the natural polarization of sun light as information. The list of malicious effects on plants and animals could be continued, see [9].

Energy consumption is another relevant aspect. A rough estimation of 25 % of our total energy needs is required for lighting purposes. The introduction of daylight saving periods from April to October happened for a reason benefiting a large amount of energy saving every year. You can read more about this topic among other harmful effects of light pollution in the papers [2, 4, 12].

LED technology offers a possible solution to light pollution [5, 8, 10]. The light of LEDs is much more focusable [3] and can also be dimmed, even adaptively to traffic density [15]. LEDs have longer lifetime and consume less energy [14] than incandescent light bulbs, but there are drawbacks of this technology too. LEDs illuminate a relatively small area, therefore application of multiple LED packages in LED street lights is necessary to replace the currently operating public lighting. This fact leads us to the question of how LEDs should be directed in the housing of lamps.

Angles of LEDs in lamps have to be set carefully to distribute light emission equally on the target surface. Configuring LED directions is a complex task, and it depends on a lot of factors, dimensions of the street and the lamppost, the minimal and maximal allowed intensity of light, and so forth. The regulation of public lighting, the future surroundings of street lights, and the cost-effectiveness should be considered simultaneously. One may focus on only one aspect, while neglecting the others, to be able to manually create designs, but it is most likely that resulted configurations will not be competitive due to high cost, or large energy consumption, or something else. We have to consider everything at the same time and that is why automatic designing solutions are required.

The quality of lighting in public areas like roads, parks, etc. is regulated by law in protection of motorists. This means that the intensity and uniformity of light have to be in specified ranges. Considering the conical lighting characteristics of LEDs, the intersection of the target surface and the light cone cast by a single LED is an ellipse. Because intensities of different light sources simply add up, providing the required visibility can be interpreted as covering rectangle-shaped areas with ellipses while overlapping is allowed. Light intensity within the same ellipse varies depending on the lighting characteristic and the direction of the source LED, thus altering LED directions also changes their extent of contribution to the coverage. Even to the lay mind, LED configuration design for public lighting purposes is obviously not an ordinary covering problem. The complexity implies that there is no hope to handle successfully this type of task with direct and deterministic optimizer methods in reasonable time, therefore we decided to create a suitable genetic algorithm to search for acceptable LED configurations as the application of such heuristic methods proved to be a good strategy in similar situations [13].

To measure the goodness of configurations, we need to evaluate them based on their properties. Beside information already provided by manufacturers such as energy consumption or price, properties related to light quality are only available

if we compute the light pattern generated by the studied configuration. This is the most important component of configuration evaluation as street lights violating the regulations cannot be deployed.

In a nutshell, the two cornerstones of automatic designing of LED street lights are the way how we construct new candidate configurations and how we determine their generated light pattern.

8.2 Light Pattern Computation

Light pattern computation means the determination of light intensity in given points on the surface we light. Regulation prescribes that these points must be the vertices of a grid with 1 m length of side. The height of lamppost and the overhanging of lamp are also necessary to proceed. Without loss of generality, we consider the housing of LEDs as a dimensionless point for simplicity because engineers can house LED sockets in a way that inserted LEDs will be directed through the same point. Lastly, LEDs are described by their lighting characteristic provided as intensities measured in different horizontal and vertical angles in fix distance from the light source following the format of EULUMDAT [1].

Algorithm 1 Compute Light Pattern

```
generate evaluation points
for all evaluation point p do
    for all LED l do
        calculate the direction vector LP pointing to p from l
        calculate the angles of the direction pointing to p and the own direction of l
        interpolate the base intensity towards p
        determine the light intensity in p based on distance
        increase the total intensity in p
    end for
end for
```

The intensity of emitted light decreases quadratically by the distance measured from its source, and the effects of different LEDs simply add up, therefore the whole computation can be considered as repetition of an elementary subtask, calculating the intensity of light cast by a single LED in a single evaluation point.

Light intensities are only available in certain directions, therefore we have to determine which known values are the closest to the value we need. The first step is to calculate the direction from the light source to the evaluation point. Having the angles between this direction and the own direction of the LED, we are able to determine a base intensity towards the evaluation point. We applied bilinear interpolation for this purpose using four known intensity values. The final intensity can be obtained easily based on distance.

Algorithm 1 is a brief step-by-step pseudo code of the computation, but it will not give the correct pattern as it only takes into account the lamp which belongs to the target area, hence further adjustments are needed. Despite the high level control

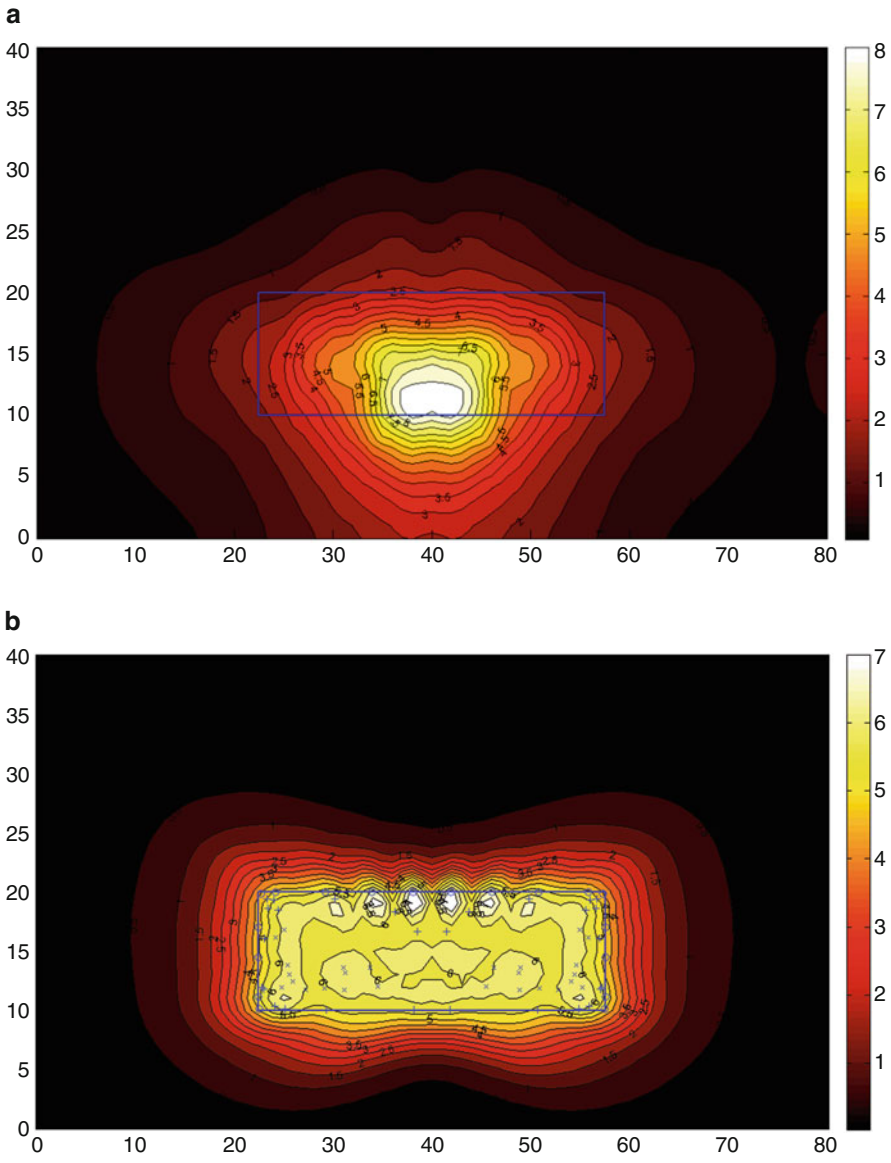


Fig. 8.1 Light patterns of street lights illuminating a *rectangle shaped* street section. *Brighter colours* denote higher intensities measured in LUX. (**a**) Regular incandescent street light. (**b**) Optimized LED street light

over the light of LEDs, we have to include the effects of neighbouring lamps to obtain a valid pattern because intensities do not drop to zero when we leave the borders of the target area as Fig. 8.1 shows. Light patterns also have at least one axis of symmetry, but 2 or 4 are also possible, depending on how the lamp will be deployed compared to the others. Therefore, we do not have to determine the intensity in every evaluation point. Exploiting symmetry, only half of points or even less need to be managed—significantly reducing the runtime.

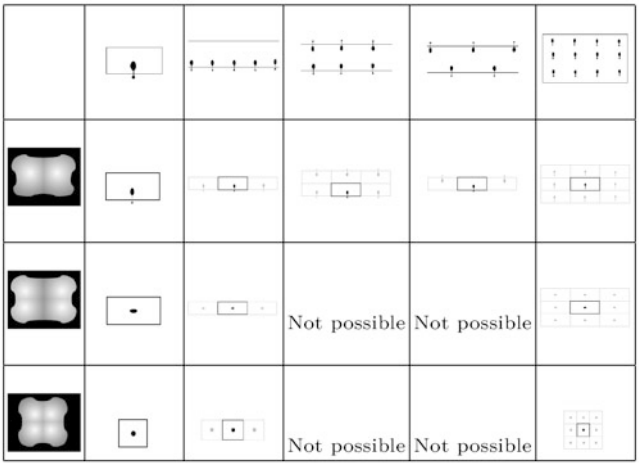
All considered, 11 different light pattern settings are possible depending on lamppost deployment and pattern symmetries, summarized in Table 8.1, which cover most public lighting cases ranging from simple streets to parking lots.

8.3 Global Optimization by Genetic Algorithm

Constrained by complexity, we can only approximate the globally optimal LED configuration in acceptable time. Any designer tool has to be capable of combining different parts of configurations, which are already optimal at some level, to move towards better solutions while it also involves randomness to be able to leave local extremal points. Genetic algorithms seem to offer a suitable approach to handle our problem-type.

Researchers apply genetic algorithms in many areas ranging from optimization to machine learning to solve problems which cannot be handled by other means. The idea of genetic algorithms comes from natural evolution. The basic concept is to model the objects of a problem space as entities, or candidates in other words, of a population and let the rule of "the strong flourish, the weak perish" work out.

Table 8.1 The 11 different light pattern scenarios based on the number of axes of symmetry and the deployment of lampposts



Elaborating more, we repeat the following steps in subsequent iterations until some common stopping criteria are met:

1. Give a fitness value to every candidate based on its properties.
2. Take out some candidates from the population selecting more probably the ones whose fitness is low.
3. Mutate some candidates by slightly altering their properties.
4. Crossover entities mixing their properties somehow in the offspring to replace the ones you took out earlier.

This description may seem very intuitive and it also well shows how we let the principles of evolution help us to find whatever we are looking for represented as the best survivor in the population. The most important concepts are the genetic operators, the way we calculate fitness, the selecting strategy of survivors, and how the objects are distilled into candidates. For an in-depth study of genetic algorithms and applications, see the books [6, 7].

In our case, population naturally consists of different LED configurations. In more detail, each candidate solution contains vertical and horizontal angles, lighting characteristics, and power consumption data for every LED in the configuration. Determining and storing precise positions of LEDs in a lamp is omitted which the engineers designing the final physical product are responsible for.

Choosing the right operators was a more delicate decision. Three mutation operators are used to alter angles, or LED types, or to take out, and put back LEDs into configurations. Angle modification has the least impact on fitness function while the others concern not just the light pattern but every criterion too. Crossover, in contrast of mutation, has a much larger effect on configurations as it is tasked to introduce new approaches of lighting. As above mentioned, our intention was to combine configurations which light considerably well different parts of the target region, see Fig. 8.2. The following steps implement this idea:

1. Choose two parent configurations.
2. Generate a rectangle randomly in the target region with uniform distribution.
3. Select the LEDs pointing in the rectangle from one configuration and the LEDs pointing out of the rectangle from the other configuration. Switch the roles of parent configurations and repeat the process.
4. The two resulted sets of LEDs will compose the child configurations.

The above steps are simple and intuitive. We tested the operator with different parameters of rectangle generation to make it fit best to its intended goal. Finding the maximal and minimal allowed area of rectangles was the key. Too small rectangles result in an insignificant change of configurations while too large ones are likely to include poorly lighted areas as well, again not bringing improvement into the population.

This set of genetic operators provide a fine-grained tool set to create and modify candidate solutions in various levels.

The final element we have not discussed yet is the fitness function, the compass of designing for which we reasoned on behalf of automatic configuration design

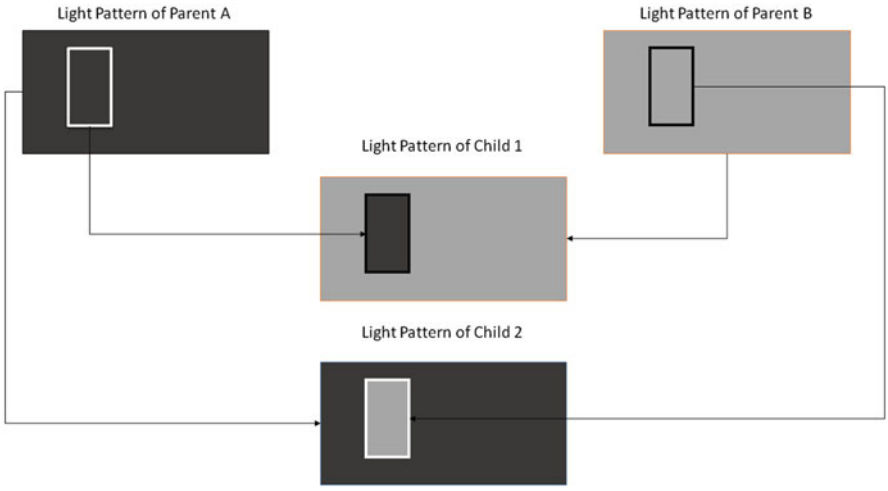


Fig. 8.2 Illustration of recombining two configurations based on light patterns

in the first place. The fitness value unites the goodness of configuration from the economic, energetic, and functional point of view simultaneously. We define fitness as the sum of several penalty terms derived by observing the following properties:

1. difference between the expected average intensity and the current values in evaluation points,
2. difference between the expected and current average of intensity,
3. total energy consumption,
4. the number of LEDs, and
5. difference between the expected and the current variance of intensity.

The first two expressions measure how far configurations are from regulations. If the intensities and the average are not in the allowed range, an additional penalty constant is also applied. Soundly, this forces the genetic algorithm to consider regulations first and everything else second. The user inputs are the strict bounds for term 1 and 2, and the weights expressing the relative importance compared to each other in case of the last three.

8.4 Results

The designing application has two components, a JAVA graphical user interface (GUI) and an optimizer written in c++. The GUI handles typical features as creating, opening, saving, etc. designing projects. When a new project is started, a step-by-step wizard guides the user to set optimization parameters, usable LED types, lamppost settings, and other user defined values. After everything is prepared, the

Table 8.2 Five test cases and their running time of automatic design

No.	Width (m)	Length (m)	LEDs	Intensity (lux)	Total CPU runtime (s)	Optimization runtime (s)
1.	30	10	27	6	299	269
2.	30	10	48	10	431	391
3.	35	20	38	6	705	651
4.	40	20	68	8	1,505	1,410
5.	50	25	100	8	2,849	2,673

GUI starts the optimizer. During optimization, the genetic algorithm frequently sends back the best solutions, whose light pattern and other describing numerical information are visualized in the GUI. The user can stop the optimization whenever he or she decides that the currently shown configuration fulfils the requirements. Otherwise, the process stops when all the characteristics of the best configuration found are within the allowed ranges and it does not change significantly over several iterations.

We implemented the genetic algorithm and the light pattern computation from scratch. Only the GUI relies on third party libraries to read and write XML files and to export LED configurations in PDF format. The program runs on Windows operating systems as this was the platform our industrial partner requested.

Assessing capabilities of the developed methodology was a difficult issue. First, LED street light manufacturers tend to keep their designing processes as well guarded secrets. Academic research groups and companies release improved designs from time to time but never designing tools. Programs available on the market are mainly concerned about visualizing light plans as realistic and fast as possible, but the burden of creating plans is left to the user. Unfortunately, this means that we were unable to compare our software to competitors as they have not presented their results yet.

Our only option was to compete with our industrial partner's engineers. Manual creation of even a single LED configuration takes long hours, therefore we could only ask for a few test cases from which several are shown on Table 8.2. We ran the tests on a simple laptop having an Intel Core I3-370M processor and 3 GB memory. On average, configurations our program found were at least twice better than manually created ones regarding the objective function. The largest difference appeared in the uniformity of light patterns as the automatically designed ones turned out to be much more smoother.

A practical feature that the industrial partner specially asked for is the possibility to add already configured LEDs to configurations in advance whose properties cannot be modified during design. This might seem a little bit odd at first glance. Why would anyone want to force such constraints to the algorithm by adding manually set elements to a lamp? Obviously no one would, but if we create a configuration for a certain lighting scenario, by this feature, we are able to adjust it with also automatically designed LEDs to fit another one. This allows us to produce

Table 8.3 Running time comparison of CPU and GPU based implementations in seconds

#	Total CPU time	Total GPU time	CPU optimization time	GPU optimization time
1.	299	269	144	111
2.	431	391	151	116
3.	705	651	184	134
4.	1,505	1,410	251	156
5.	2,849	2,673	334	181

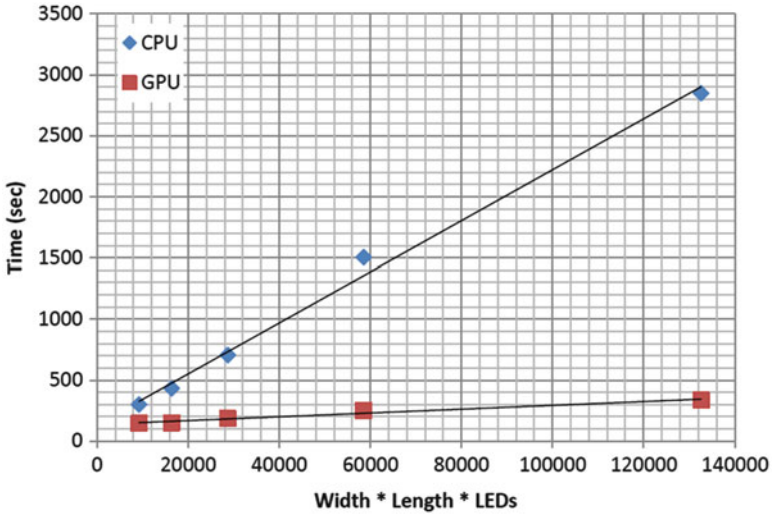


Fig. 8.3 The effect of parallelization

the same housing for different streets or roads. We only need to plug the right LEDs into the right sockets before deployment. This results in less product types to manage saving even more for the companies.

The stopping conditions of optimization were met after 3–4 h for typical design settings, and configurations fulfilling every hard condition already emerged after 20–30 min as Table 8.2 shows. After the first test runs, the industrial partner became interested in the reduction of optimization time assigning a new objective to us.

Profiling the software revealed that 80 % of executed operations are related to light pattern computation. As indicated before, most of these calculations can be executed independently, hence we decided to execute the intensity calculation of different evaluation points in parallel. We based the new implementation on NVIDIA's CUDA technology [11]. We repeated the optimization using the same seeds for random number generation, see Table 8.3. This happened on the same laptop we used earlier with an NVIDIA GeForce GT 335M video card.

As illustrated in Fig. 8.3, we analysed the runtime as a function of problem size, which is the product of the number of applied LEDs and the width and length of the target rectangle. Linear regression resulted in the following coefficients:

$$\begin{aligned} \text{CPUruntime} &= 0.020{,}94\text{problemsize} + 127, \\ \text{GPUruntime} &= 0.001{,}58\text{problemsize} + 134. \end{aligned} \tag{8.1}$$

Dividing the steepness' of (8.1) by each other, we obtain a 13 times speedup limit in runtime. Although this growth in performance can truly be harnessed when larger problems are encountered, parallelization significantly reduces runtime in every case.

8.5 Conclusion

Designing optimal LED configurations for public lighting purposes is far more complex, even for expert engineers, than to be handled manually. This global optimization problem belongs to the classic field of covering problems; however, it can only be approached by stochastic optimization methods due to high dimensionality and special constraints.

We developed a software solution which is capable of designing LED configurations automatically while it considers every relevant factor during the process. Our approach is to handle configuration construction by a genetic algorithm which combines configurations based on partially good light patterns using crossover to obtain better candidate solutions whom mutation refines further. The objective function is the weighted sum of different penalty terms measuring the goodness of energy consumption, quality of lighting, and total cost of applied light sources.

As light pattern related operations put out the bulk of required computation during design, we took advantage of any axial symmetry present in the problems to reduce light intensity evaluation to the most necessary level. After finishing the first prototype, we reimplemented light pattern computation using NVIDIA's CUDA technology to make the optimization even faster by handling effects of different LEDs simultaneously. The result of this effort is a 13 times speedup in limit.

In our experience, automatic design can lead to at least twice better configurations than manual design. The most outstanding difference comes out in the uniformity of light intensities revealing the main strength of our algorithm. The presented test cases prove that the developed optimization technique can truly help the work of engineers reducing designing time and other costs.

References

1. EULUMDAT specification: http://www.helios32.com/Eulumdat.htm (2014). Accessed 12 Nov 2014
2. Falchia, F., Cinzanoa, P., Elvidgeb, C.D., Keithc, D.M., Haimd, A.: Limiting the impact of light pollution on human health, environment and stellar visibility. J. Environ. Manag. **92**, 2714–2722 (2011)
3. Fournier, F., Cassarly, W.: SOFTWARE & COMPUTING: Freeform optics design advances lighting and illumination. Laser focus world. http://www.laserfocusworld.com/articles/print/volume-47/issue-3/columns/software-computing/freeform-optics-design-advances-lighting-and-illumination.html (2011). Accessed 12 Nov 2014
4. Gallaway, T., Olsen, R.N., Mitchell, D.M.: The economics of global light pollution. Ecol. Econ. **69**, 658–665 (2011)
5. Gereffi, G., Dubay, K., Lowe, M.: Manufacturing Climate Solutions Carbon Reducing Technologies and U.S. Jobs. Center on Globalization, Governance & Competitiveness, Duke University, Durham (2008)
6. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Professional, Boston (1989)
7. Lawrence, D.: Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York (1991)
8. Lee, H., et al.: High-performance LED street lighting using microlens arrays. Opt. Express **21**, 10612–10621 (2013)
9. Longcore, T., Rich, C.: Ecological light pollution. Front Ecol. Environ. **2**, 191–198 (2004)
10. Nuttall, D.R., Shuttleworth, R., Routledge, G.: Design of a LED street lighting system. In: 4th IET International Conference on Power Electronics, Machines and Drives, pp. 436–440 (2008)
11. NVIDIA's CUDA homepage: http://www.nvidia.com/object/about-nvidia.html (2014). Accessed 12 Nov 2014
12. Riegel, K.W.: Light pollution: Outdoor lighting is a growing threat to astronomy. Science **179**, 1285–1291 (1973)
13. Szabó, P.G., Csendes, T., Casado, L.G., García, I.: Lower bounds for equal circles packing in a square problem using the TAMSASS-PECS stochastic algorithm. In: Abstracts of GO. '99, Firenze, pp. 122–126 (1999)
14. White, J.J.: No-cost LED street lighting modernization. EATON report (2013)
15. Wu, Y., Shi, C., Zhang, X., Yang, W.: Design of new intelligent street light control system. In: 2010 8th IEEE International Conference on Control and Automation (ICCA), pp. 1423–1427 (2010)