Computing and Informatics, Vol. 36, 2017, 1492–1522, doi: 10.4149/cai_2017_6_1492

USING A FULL SPECTRAL RAYTRACER FOR CALCULATING LIGHT MICROCLIMATE IN FUNCTIONAL-STRUCTURAL PLANT MODELLING

Michael Henke

IRHS, INRA, AGROCAMPUS-Ouest, Université d'Angers
SFR 4207 QUASAV, 42 rue Georges Morel, 49071 Beaucouzé Cedex, France
&
Leibniz Institute of Plant Genetics and Crop Plant Research (IPK Gatersleben)
OT Gatersleben, Corrensstrasse 3, D-06466 Seeland, Germany
e-mail: `mhenke@uni-goettingen.de`

Gerhard H. Buck-Sorlin

IRHS, INRA, AGROCAMPUS-Ouest, Université d'Angers
SFR 4207 QUASAV, 42 rue Georges Morel, 49071 Beaucouzé Cedex, France
e-mail: `gerhard.buck-sorlin@agrocampus-ouest.fr`

Abstract. Raytracers that allow the spatially explicit calculation of the fate of light beams in a 3D scene allow the consideration of shading, reflected and transmitted light in functional-structural plant models (FSPM). However, the spectrum of visible light also has an effect on cellular and growth processes. This recently created the interest to extend this modelling paradigm allowing the representation of detailed spectra instead of monochromatic or white light and to extend existing FSPM platforms accordingly. In this study a raytracer is presented which supports the full spectrum of light and which can be used to compute spectra from arbitrary light sources and their transformation at the organ level by absorption, reflection and transmission in a virtual canopy. The raytracer was implemented as an extension of the FSPM platform GroIMP.

Keywords: Full spectral raytracing, light modelling, FSPM, GPU, photosynthesis, GroIMP

Mathematics Subject Classification 2010: 68-U05

1 INTRODUCTION

Accurate computation of light flux in a plant canopy and thus of its light microclimate should be a prerequisite for every crop model, whether it considers a single plant individual or an entire canopy, since light is the single-most important input parameter for a photosynthesis model, and photosynthetic activity controls plant growth and development. Functional-structural plant modelling (FSPM) refers to a paradigm for the description of a plant by creating a (usually object-oriented) computer model of its structure and selected physiological and physical processes, at different hierarchical levels: organ, plant individual, canopy (a stand of plants), and in which the processes are modulated by the local environment [6]. By better describing the heterogeneity of the micro-environment and considering physiological processes that are modulated by it, FSPM have become increasingly realistic. Correspondingly, on the functional side, implemented processes have become much more complex. Most approaches for light computation have been focusing on the quantity of photosynthetically active radiation (PAR) reaching different parts of a plant. Light quality is as important as quantity, but much harder to estimate quantitatively. Instead of a single ray with only one power value the entire spectral composition of each ray needs to be traced, with reflection, absorption, and transmission being different for each wavelength. Such complex and computationally demanding processes become manageable with the development of highly parallel computing techniques on the graphics card (GPU) [39]. Light quality exerts a significant influence on canopy development [19, 1, 3]. Light quality, via photomorphogenesis, influences shoot architecture and source/sink ratio, and thus indirectly plays a major role for, e.g., fruit quality [5, 20]. Furthermore, reflection and transmission spectra varied considerably among light- and shade-adapted leaves in different apple cultivars [35].

In the past 25 years, several approaches to estimate the light environment have been developed. Greene [21] considered the entire sky as a hemispherical diffuse light source and computed the local light environment within a plant canopy using raycasting. Another early approach was the one by Kanamaru [25]: here, the amount of light reaching a given sampling point was calculated by assuming that it was at the centre of projection, and by subsequently projecting all leaf clusters of a tree onto a hemisphere surrounding this point. The Transrad model by Dauzat [14] simulates multiple scattering of light and returns the complete radiative balance of a canopy. Mech [37] introduced a light environment model based on Monte Carlo (MC) path tracing of photons, with the possibility of interfacing it with virtual plants created using open L-systems [38]. Besides allowing the computation of the absorbed power this approach was also capable of calculating the spectral composition of light. The LIGNUM model implemented two approaches: a raycasting based approach called "mutual shading of segments" [40] and a voxel space method described in [44]. Disney [17] reviewed the use of MC methods in optical canopy reflectance modelling. He predicted a good deal of potential for MC based methods but also adjusted advantages for current analytical methods in cases where speed, invertibility, or a generalised statement of parameter influences are the keys. Estimation of canopy light interception by using the Beer-Lambert law is a simplified method used in many crop models. This method only accounts for leaf area index (LAI) and leaf angle distribution (LAD) without considering the crop's structural heterogeneity in space. Certain modelling approaches that are intermediate between process-based and functional-structural plant models, e.g. GreenLab [13] used this simplified approach. Wang [49] introduced light interception based on photon mapping to replace the Beer-Lambert law in the Qingyuan software, a GreenLab clone. The CARIBU model implemented radiosity for light sampling [11]. CARIBU was subsequently made a part of the OpenAlea software package [42, 10]. AmapStudio, Simeo and AmapSim [34] used the MMR model implemented in the Archimed simulation platform [15, 16]. MMR performs calculations in three steps:

1. MIR calculates the incident radiation intercepted by plant organs;
2. MUSC calculates the scattering of light within the canopy which is divided into horizontal layers and
3. RADBAL combines the previous results according to radiative conditions provided by a meteorological data file.

The model outputs provide the irradiation of plant organs and a map of radiation reaching the ground. The Xplo software used this approach, too [45]. Cieslak [12] used a randomised quasi-Monte Carlo (RQMC) sampling method (QuasiMC) and confirmed that RQMC offers advantages in speed and/or accuracy improvement over MC.

A common work flow for most approaches is to follow a multi-stage process of exporting the 3D scene to a format that can be imported by an external renderer (library/software) and then to reimport the results of the light computation into the core model for further use. Working directly on the generated structure and in this way making the steps of exporting and reimporting redundant would be an obvious way to save computation time, given that the whole work flow is already computationally expensive. The GroIMP platform was among the first model environments that included a Monte-Carlo radiation model [23].

Based on these developments of MCRT methods for FSPM [23, 12], we have published a number of articles [7, 8, 48] describing applications and validations of these existing light modelling methods to concrete cropping situations (rose and tomato production under controlled conditions in the greenhouse), thereby also showing up the gaps and weak points associated with these approaches. The present study describes the latest extension of GroIMP allowing full spectral raytracing powered by parallel computing on the GPU. To our knowledge, in the past seven years, no meaningful progress has been made in the field of light modelling methodology for FSPM. Therefore, the present paper is an attempt to catch up with the needs for progress in light modelling identified from own applications and from enquiring within the community of FSPM modellers.

2 MATERIAL AND METHODS

In order to make full spectral raytracing available for FSPM, and to allow the computation of the spectra from arbitrary emitting light sources and their transformation at organ level by absorption, reflection and transmission in a virtual canopy, a framework that supports the following fundamental aspects is required:

1. a global illumination model (light model),
2. light sources, and
3. a local illumination model (shader) (Figure 1).

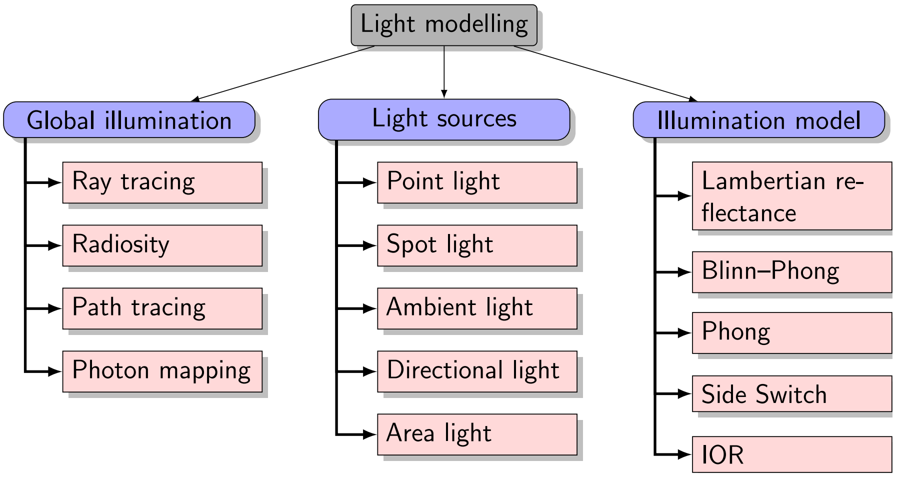


Figure 1. Main computation techniques, light sources, and local illumination models used in computer graphics. The list of examples is not exhaustive.

The features presented in this work have been implemented and integrated in the framework of the modelling software GroIMP [22, 32, 31], with the integrated language XL [29, 30]. The hardware requirement to perform GPU-based raytracing is a programmable graphics card with OpenCL support (SSE > 4.1, [33]).

2.1 Light Model

Figure 2 illustrates the overall work flow of light transport simulation within a 3D scene. The light model acts as the overall control unit: Depending on the method used for light calculation it performs different steps to estimate the light distribution. For standard raytracing a defined number of rays is emitted by one or several light sources. Each ray is traced throughout the scene and in case it hits an object it is treated according to the local optical properties of the hit object, cf. Section 2.3 and Figure 7. For each object in the scene the amount of absorbed light is collected.

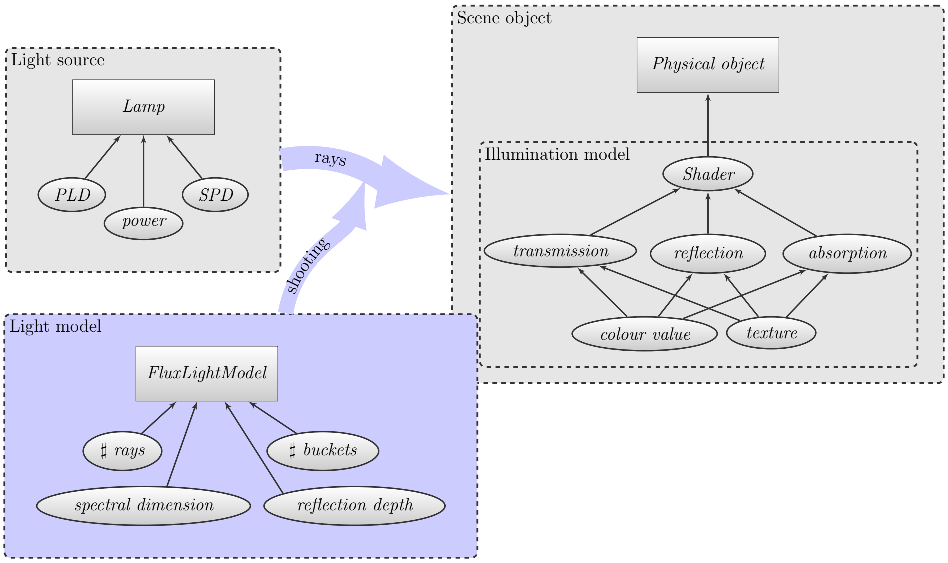


Figure 2. Diagrammatic representation of a common system for light modelling. The light source and the objects are situated in a virtual scene: After invoking the light model all light sources emit virtual rays into the scene, and their paths are traced until one of the cut-off criteria is reached: ray leaving the scene; ray spectral power below a threshold value; maximum number of reflections reached.

GPUFlux, an integrated light model implemented by [47], is a high-performance light model that uses OpenCL [28] to directly access the processor of the graphics card (graphical processing unit – GPU) as well as CPUs that support SSE > 4.1 [33]. Since GPUs are designed to perform highly parallel computation, the computation time can be reduced at least by factor ten and up to more than one hundred times (depending on the compared CPU and GPU). This and the fact that multiple devices, e.g. several GPUs and all threads of a CPU, are supported in parallel considerably speeds up light computation. As a further feature, the full spectrum of light is supported with a minimal optical resolution of 1 nm, over an arbitrary spectrum, but with default values ranging between 300 nm and 800 nm – the range is not limited by the system, however, far outside the visible spectrum it will not produce correct results, as the physical properties of such rays will be too different from those within the visible spectrum. Finally, three types of illumination models are implemented: a path tracer, a bidirectional path tracer and spectral renderer based on a spectral Monte-Carlo light tracer.

Each ray is traced through the scene until one of the following cutoff conditions is triggered:

1. the minimal power of a ray is lower than a predetermined threshold power,
2. the maximal depth of reflections is reached, or

3. the ray leaves the bounding box of the scene.

To control precision of calculation and computation time the cutoff power and maximal recursion depth can be defined by the user.

2.2 Light Sources

An arbitrary light source can be defined by two parameters:

1. physical light distribution (PLD),
2. spectral power distribution (SPD).

The physical light distribution describes the luminous intensity, i.e. the measure of the wavelength-weighted power emitted by a light source in a particular direction per unit solid angle (cf. Figure 3 a)), based on the luminosity function, a standardised model of the sensitivity of the human eye, over the whole sphere (cf. Figure 3 b)). It is usually measured in candela (cd) per steradian beam (st) and measured using a goniophotometer for light sources such as light bulbs. Most manufacturers of light sources provide this information on their websites for public use (see Section 3.1 for the conversion from candela to watt).

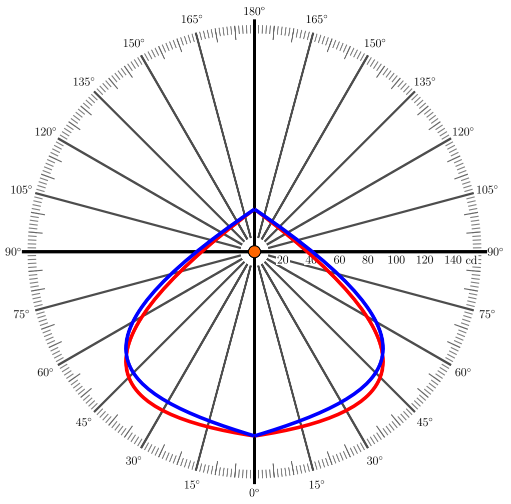
One common file format to describe a PLD is called IES and has been introduced by the Illuminating Engineering Society [24]. Another common format is LUM. Both can be directly imported by GroIMP and both are simple ASCII files that can be converted into each other without problems. Figure 4 shows a GroIMP snapshot of a set of common predefined light sources provided by GroIMP. By turning on an option of these lights the physical light distribution can be visualised through simple lines. The rendered result of a lamp demo model which is a default example included in GroIMP is shown in Figure 5.

The spectral power distribution (SPD) measurement describes the power per unit area per unit wavelength of an illumination. The ratio of spectral concentration (irradiance or exitance) at a given wavelength to the concentration of a reference wavelength provides the relative SPD, as shown in Figure 6 for a high-pressure sodium (SON-T) lamp, which is commonly used as additional growth light source in greenhouses.

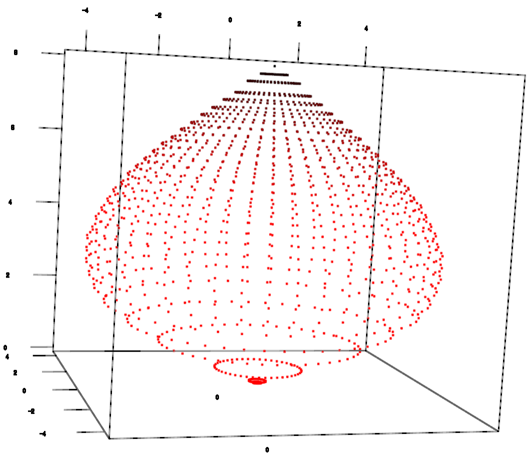
2.3 Illumination Model

An illumination model describes the local illumination, more generally the local optical properties of an object at a certain point on the surface (Figure 7), which includes the three basic properties, absorption, reflection, and transmission.

In computer graphics the optical properties of an object are defined by a shader which is mapped onto the surface of an object. The most common shaders are the Lambertian reflectance [2], which only supports diffuse reflection, and the Phong shader, developed by Phong [41], which is an advanced shader that supports ambient, diffuse and specular reflection (Figure 8).



a) A polar distribution diagram (also called polar curve) showing the luminous intensity values with increasing angles from two imaginary axes of the lamp which is placed in the centre. Red: 0–180° plane, blue 90–270° plane.



b) 3D visualisation of the same light source. The colour of each point (gradient from black to bright red) as well as the distance to the light source both indicate the power emitted by a light source in a particular direction per unit solid angle.

Figure 3. Two visualisations of a physical light distribution of a not further defined light bulb. The 2D case shown in sub-figure a) is the common case usually provided by manufacturers.

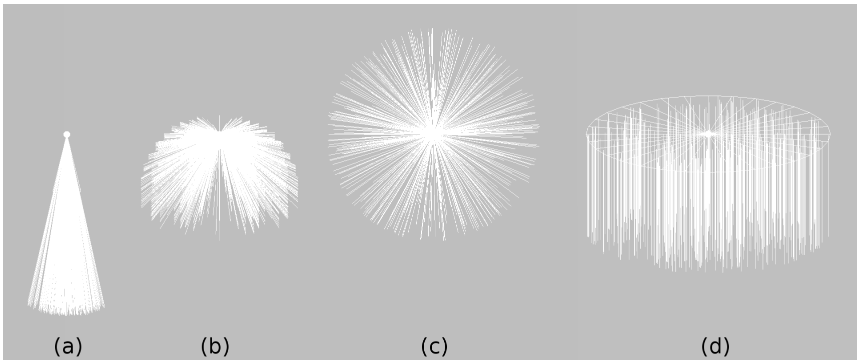


Figure 4. Visualisation of physical light distributions of different light sources: a) spot light, with a defined opening angle; b) user defined distribution; c) point light, equally distributed; d) directional light, equal distribution over an area.

In GroIMP (Code 2.3), each property of each type of shader can be defined separately. The implemented Phong shader allows the definition of values for: shininess, transparency, ambient and specular reflection, emission, diffuse reflection, transparency shininess, and diffuse transparency. Each one of these properties can be defined either as constant for each colour value (graytone) or for each base colour

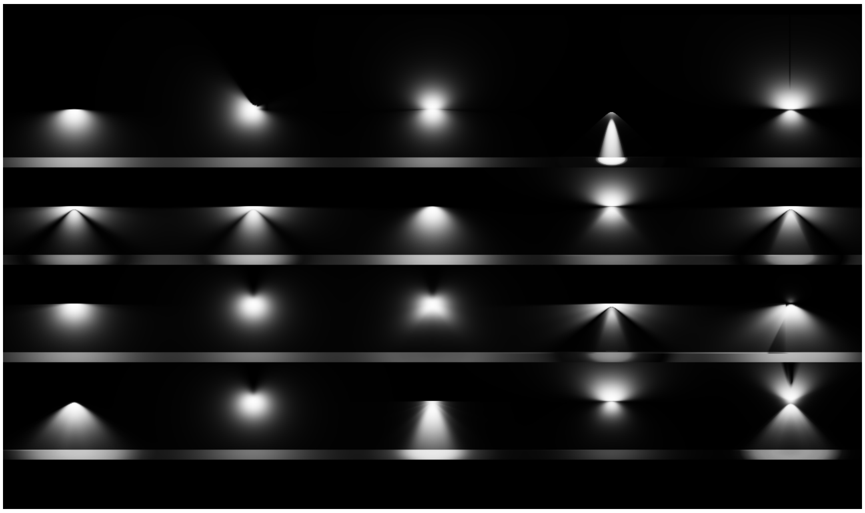


Figure 5. GroIMP snapshot of a lamp demonstration model (rendered image), simulating a set of 20 lamps placed on a wall. The camera is looking from the side showing the distribution of light reflected from the wall and a part of the pattern produced on the ground. Model source is available in the example gallery of GroIMP 1.5.

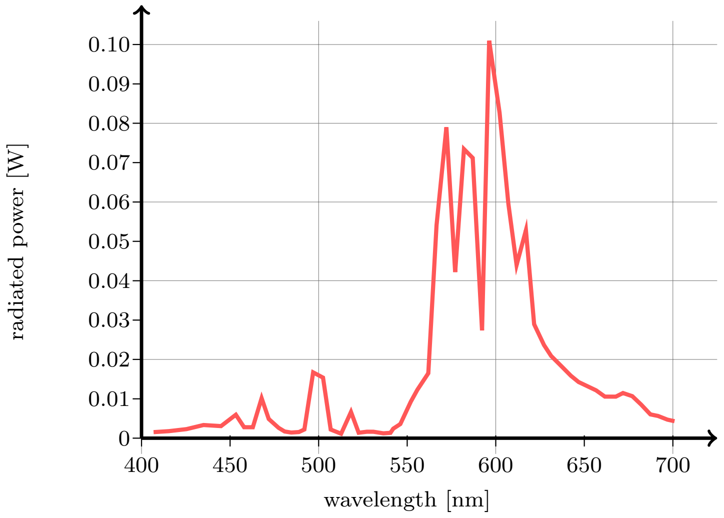


Figure 6. Spectral power distribution of an EYE Lighting – Sunlux® LU400 lamp. The resulting line on the graph is the Spectral Power Distribution (SPD) Curve, and shows the power distribution across the visible spectrum.

independently (RGB colour). Additionally, for spectral raytracing spectral power distributions can be applied.

2.4 Verification

As described above, the three aspects, a global illumination model (light model), light sources, and an illumination model (shader), are needed to perform spectral raycasting for FSPM. Whenever the light climate in a complex system is to be evaluated it is appropriate first to verify each part separately.

Only a few parameters of the light model can be evaluated directly. One thing to validate is whether the total amount of power absorbed by all objects of a scene is equal to the total power output of all light sources. In GroIMP this can be done using what is called an execution rule, identified by the operator "::>". This type of rule will leave the graph structure unchanged (i.e. the topology of nodes and edges), while modifying one or more parameters associated with the node or subgraph; the node, or subgraph, identified by the search pattern on the left-hand side of the rule is thus not replaced by anything on the right-hand side of the rule as is the case in normal L-system rules. In this example the light model is invoked to calculate the amount of absorbed radiation by a specific object before this amount is added to the global counter:

```
float total = 0;
x: ShadedNull ::> { total += LM.getAbsorbedPower(x).integrate(); }
```

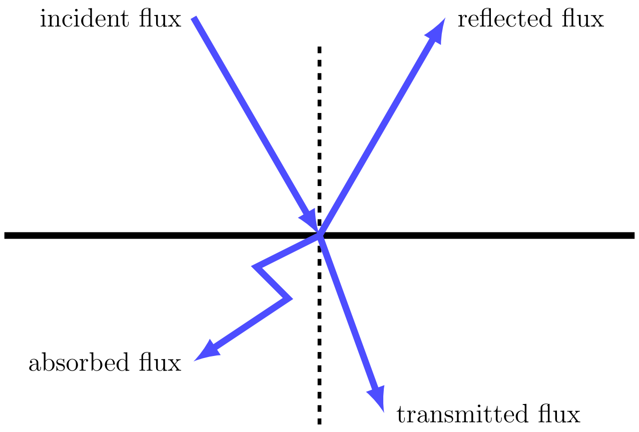


Figure 7. Definition of local illumination. On impact at the surface of a material, the incoming flux is split up into a reflected, absorbed and transmitted flux. Both the transmitted and reflected flux can be further subdivided into a direct and diffuse part.

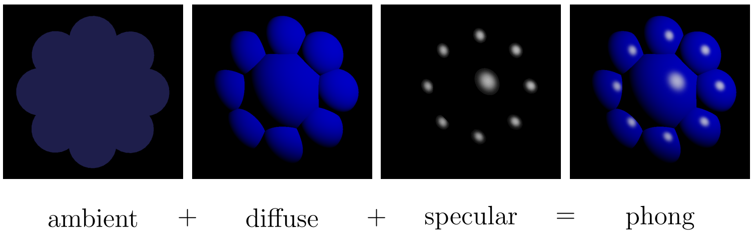


Figure 8. Phong shader, consisting of a combination of ambient, diffuse and specular reflection

For each object found the light model (*LM*) is invoked to obtain the amount of light absorbed. Here *ShadedNull* is a super class of all visible objects in GroIMP. Since the result is returning a spectrum it needs to be integrated before it can be added to the total sum (done with the method *integrate()* of *LM*). An alternative way, assuming the absorbed power has already been calculated by the light model and stored within an attribute *absorbedPower* of the objects, would be the use of graph queries:

```
float total = sum( (* ShadedNull *).absorbedPower );
```

that can directly be called on the GroIMP console. The part "(* *pattern* *)" indicates a graph query searching a specific pattern within the graph (in this case of all nodes of type *ShadedNull*). All parts of the graph matching the pattern will be returned by the query and are available for further actions. Here we query the amount of power absorbed by each object. The results for each object are aggregated by the *sum* function to obtain the final result.

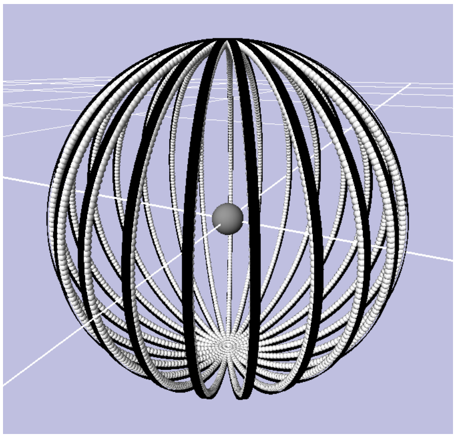


Figure 9. 3D visualisation of a sensor sphere used for verification of the PLD. The light source is placed at the centre of the sphere. The virtual sensor nodes (*SensorNode*) of the model are as many as, and at the same locations as, the real light sensors used to obtain the PLD.

For light sources

1. the spectral power distribution,
2. the physical light distribution as well as
3. the total power output are of interest.

Regarding 1. and 3. the verification is easy: 1. each ray is initialised by the light model per default with the defined spectrum, 3. the total power is equally distributed over all emitted rays. To check this, a sphere with a black shader can be used as a test object, as this will absorb all incident light. The light source is placed in its centre and the light model will be executed once to obtain the total amount of light absorbed by the sphere. The verification of the PLD (2.) requires more effort: since RT is a stochastic process the actual light distribution depends on the number of rays used. It is converging with increasing number of rays to the distribution given by the light source. To get an idea of how many rays are needed to obtain a converging distribution, we implemented a small test environment simulating a goniophotometer with the same number of sensor nodes as defined in the PLD-file, arranged in a sphere around the light source (Figure 9). In this scenario only direct light was registered. For a real 3D scene with all the light interaction a much larger number of rays is needed, making it difficult to give a recommendation. The number of rays is proportional to the complexity of the scene. However, experience showed that increasing the number of rays used improved the results obtained, without any saturation effect being observed.

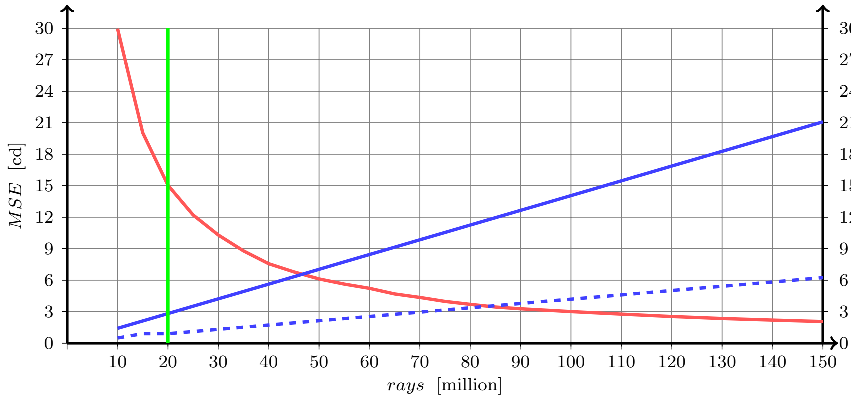


Figure 10. Model accuracy as a function of number of rays used (red line). The objective was to obtain a realistic physical distribution with minimal computational investment. Here, realistic means a minimal mean square error (MSE). The vertical green line represents the recommended minimum number of rays (20 million), whereas 50 million rays will minimize the mean standard error to 6 cd. Error values measured in candela [cd] – 1 cd ≈ 1/683 W at 555 nm, see Section 3.1. Additionally, the computation times needed on an Nvidia Quadro FX 1700M (Date of Announcement: 01.10.2008) (blue line) and on an Nvidia GeForce GTX 880M (Date of Announcement: 12.03.2014) (dashed blue line) are given. Note that for the Quadro card the unit is minutes while for the GeForce it is only maximally six seconds.

The accuracy of a reconstructed light distribution as a function of number of used rays is given in Figure 10. While the increment of computation time is linear, the MSE only decreases proportionally. At 20 million rays the MSE for the light source used in this test is around 15 cd (around 0.022 W), which would be small enough to be negligible for most applications. However, with up-to-date graphics hardware the number of rays can and should be increased to 100 or 200 million rays – which still requires only seconds (depending on the complexity of the scene).

The main factors influencing computation time are listed below:

- hardware: programmable graphics card; optionally a CPU that supports SSE $>$ 4.1, [33],
- complexity of the scene (number of objects, complexity of objects $\Longleftrightarrow$ number of triangles/facets),
- number of light sources (plays a role for rendering not for light modelling),
- resolution and range of investigated spectrum (number of buckets),

- recursion depth,
- optical properties of objects (shader: e.g. transparency, emission).

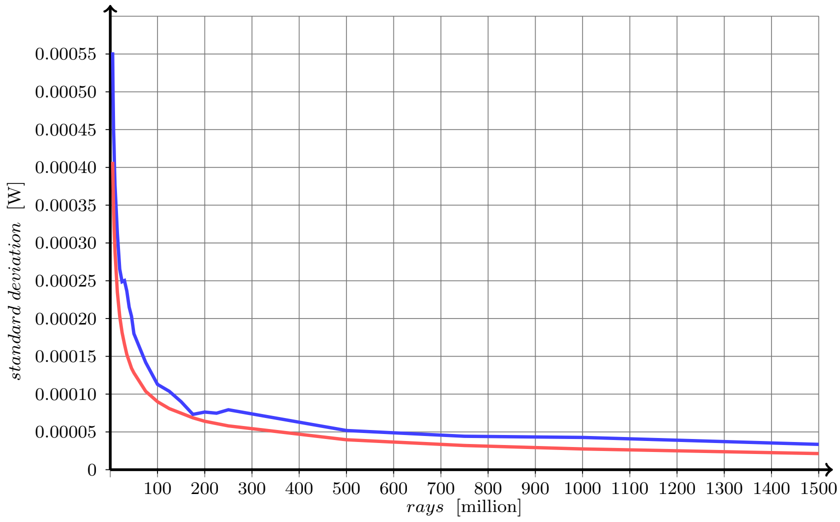


Figure 11. Standard deviation of 25 repeated light model runs for an increasing number of rays (five million to 1.5 billion rays). Blue: simple scene; red: complex scene. (Note that one run with 1.5 billion rays for the complex scene with 2 000 objects took 61 seconds on a Nvidia GeForce GTX880M.)

To test the influence of numbers of rays on the reproducibility of the results of the light calculation a simple sensitivity test was performed using two scenarios, a simple scene with only one object and a complex scene with 2 000 objects randomly distributed in a box with five meter edge length. In both tests one single spotlight with an outer angle of 30 degrees was used as light source. It was placed ten metres above the ground where the test objects were placed. Each test object had an edge length of ten centimetres. Its shader was set to black for scenario 1 and in order to get reflections it was set to 50 per cent black for scenario 2. The recursion depth was set to ten for the complex scenario while for the simple scenario it was set to one. The standard deviation of absorption of the object was calculated for 25 repeated runs of the light model, while for each repetition the light model was initialised with a different random seed, thus resulting in different ray distributions. If the variation of the 25 repetitions is small it can be shown that the distribution is reproducible. However, it does not tell much about the quality of the obtained light level. Therefore, the test needs to be repeated with an increasing number of rays and the obtained mean standard deviation and variance need to be compared. It can be expected that when more rays are used the variance will become smaller and the mean standard deviation will converge. The blue line in Figure 11 shows the standard deviation for 5 million up to 1.5 billion rays for the simple scene. For this very basic test scenario it can be observed that above 500 million rays the standard deviation nearly does not decrease further. In the second scenario, the complex scene, the absorption of each object was measured for 25 repetitions and the standard deviation was calculated. Afterwards, the average standard deviation of these 2 000 standard deviations was calculated (red line in Figure 11). While at first sight the results look similar they were on average 25 per cent better than for scenario 1. This can be explained by the simple fact that the surface area of the objects in the complex scenario is several times larger than in the simple scenario. With an increase in area the possibility of a ray to hit a particular surface increased, too, resulting in an equalisation of the average absorption during repeated tests. To sum up, it was observed that a minimum number of rays is needed to guarantee a satisfactory reproduction of a particular physical and spectral light distribution while a much higher number of rays is needed to obtain a qualitatively good light distribution simulation. With this in mind, a realistic light distribution – a prerequisite for a realistic plant simulation – requires no less than 50 million rays while any number below this is not acceptable. We recommend to use around 200 million rays to obtain a good compromise between computation time and quality of the obtained light distribution. These statements are made for a recursion depth of 10 reflections. With fewer reflections the number of rays needs to be higher.

To check the proper functioning of the shader a simple test environment (Figure 12) was created: this consisted of two cylinder objects to define the boundaries, and a virtual frame fixing the test shader in the middle and dividing the upper and lower cylinder. A spot light with a very small opening angle directly facing the shader was placed at the ground of the lower cylinder.

In the default configuration the power output for a spot light is set to 1 000 W, the spectral resolution it set to 5 nm wide buckets in the range 380 nm to 780 nm, and the number of rays is set to 1 M. An example output of a pure green shader is displayed in Figure 13.

3 ILLUSTRATION

To illustrate the possibilities of the full spectral RT presented here we chose two examples: The first one is a visualisation of a dispersion effect while the second one represents photosynthesis, one of the most important physiological processes. Regarding the latter, a wavelength dependent photosynthesis model will be presented.

3.1 Dispersion

In optics, dispersion is the phenomenon in which the phase velocity of a wave depends on its frequency [4]. One familiar consequence of dispersion is the change in the angle of refraction of different colours of light as commonly illustrated by the spectrum produced by a dispersive prism. The same phenomenon can be observed with crystals as simulated in Figure 14.

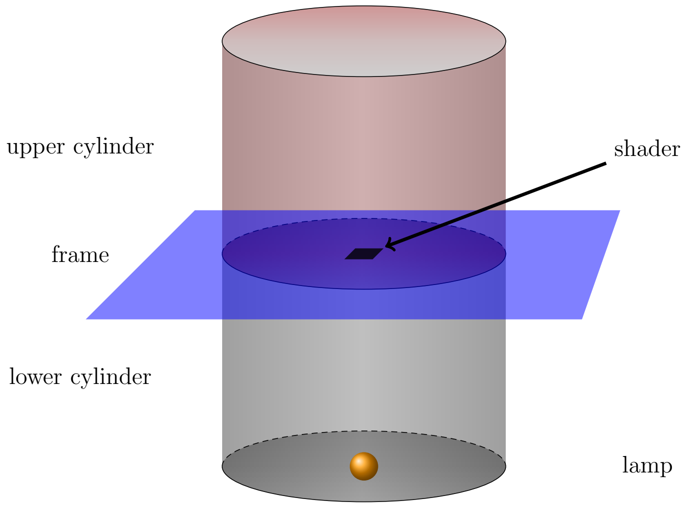


Figure 12. Illustration of the shader test environment. It provides a self-contained system for testing the optical properties of a shader, mainly: absorption, reflection, and transmission.

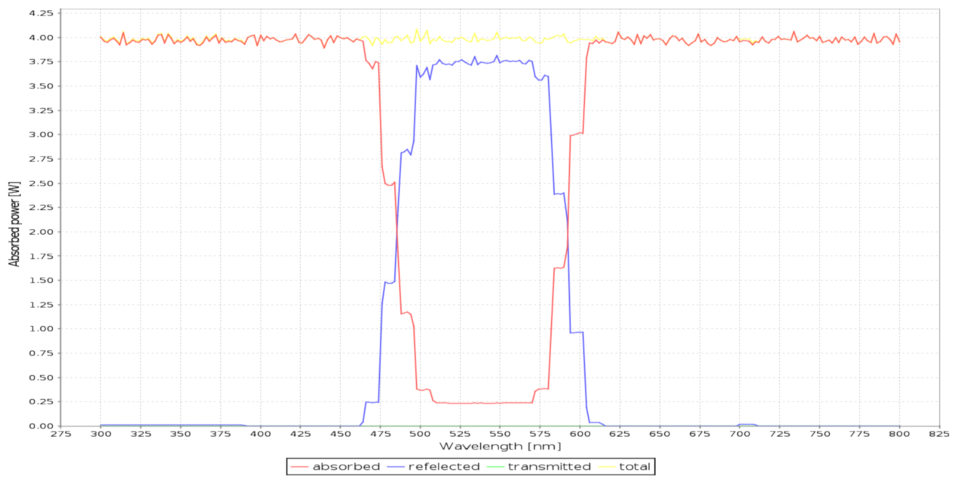
3.2 Spectral Light Reaction – Photosynthesis

In order to model photosynthesis with a high level of accuracy several aspects such as temperature, humidity, CO_2 concentration and light microclimate need to be considered.

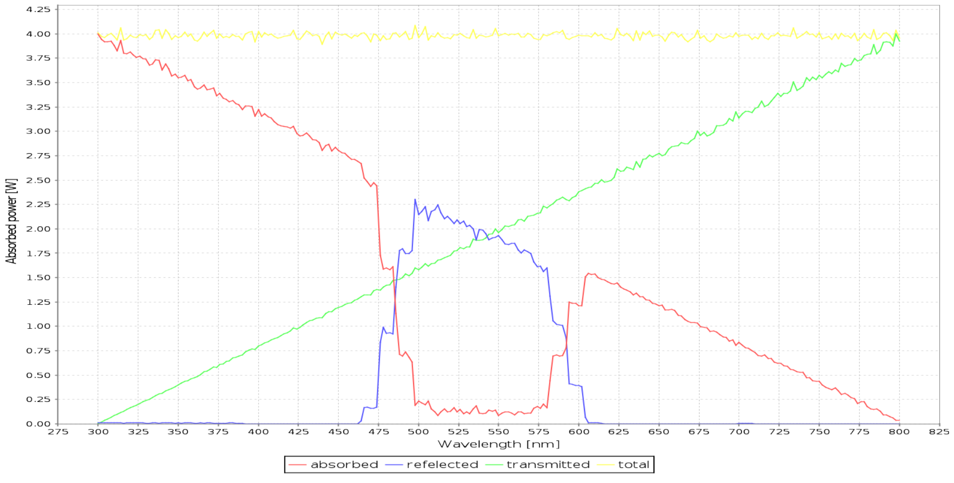
Photosynthesis, more specifically the light reaction, depends upon the absorption of light by pigments in the leaves. The diagram in Figure 15 shows that the percentage of absorbed radiation varies depending upon its wavelength, with the lowest absorption in the green waveband (550 nm), and peaks in the red (650 nm) and blue (450 nm) waveband. The principal pigments responsible for the absorption are chlorophyll *a* and *b*. However, the absorption rate in the green waveband is not zero: about 20 % of it is absorbed due to other leaf pigments, e.g., beta-carotene and chlorophyll *b*. The plot of the absorption spectra of the chlorophylls plus beta carotene correlates well with the observed photosynthetic output, Figure 15.

Figure 16 shows measured values for reflection and transmission of light by a typical soybean leaf [26]. First of all, a shader was parameterised with these values (reflection and transmission) and placed into the shader test environment (Figure 12) to verify the simulated values for absorption. This was repeated for two types of light sources, with a constant spectrum, and simulated daylight. The results are shown in Figure 17 and Table 1.

The McCree Curve (Figure 18) [36] was used to correct the absorbed radiance in order to obtain the usable amount of light per waveband.



a) power absorbed by the shader (red) = 797.87 W; power reflected by the shader and therefore absorbed by the lower cylinder (blue) = 201.25 W; power transmitted by the shader and therefore absorbed by the upper cylinder (green) = 0 W, sum of power absorbed in the scene (yellow) = $a + r + t$ = 999.12 W



b) power absorbed by the shader (red) = 396.326 W; power reflected by the shader and therefore absorbed by the lower cylinder (blue) = 106.56 W; power transmitted by the shader and therefore absorbed by the upper cylinder (green) = 496.23 W, sum of power absorbed in the scene (yellow) = $a + r + t$ = 999.12 W

Figure 13. GroIMP snapshot of a chart showing the absorbed power of one object broken down to 5 nm buckets produced by the shader test environment for a) a pure green shader, b) a pure green shader with a transmission increased linearly over the whole spectrum. Initial power output of the lamp was set to 1 000 W and the number of rays used for the simulation was 100 K.

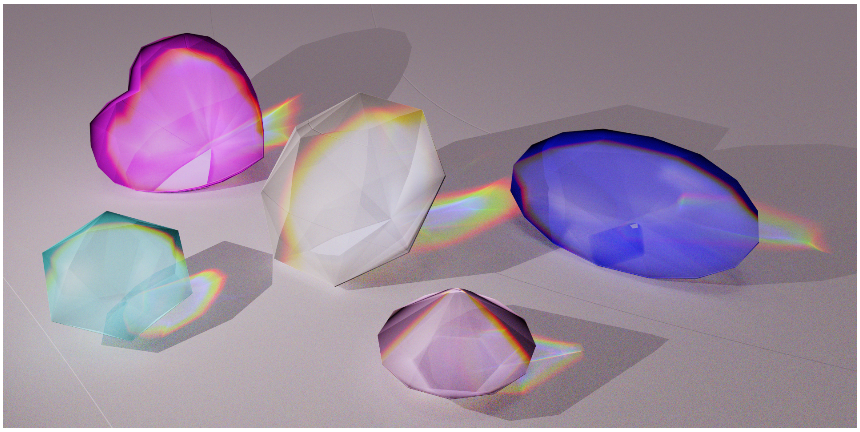


Figure 14. GroIMP snapshot, produced with the Flux renderer, showing the "rainbow" effect produced by the dispersion of white light. The dispersion effect on this image was artificially enhanced for illustration reasons only.

	Original	Case a	Case b	Case a/**Ori**	Case b/**Ori**
Transmission	31.585	30.979	27.653	0.981	0.876
Reflection	25.923	25.724	23.191	0.992	0.895
Absorption	93.714	93.247	99.110	0.995	1.058
Corrected Absorption	72.321	71.863	77.320	0.994	1.070

Table 1. Integrated absolute values for transmission, reflection and absorption of the simulated soybean leaf, Figure 17, for a) a constant spectrum and b) simulated sun light in Watt. The last two columns represent the ratio between simulated and measured/original values.

Up to this point we have estimated and verified the absorbed spectrum of our simulated soybean leaf. In the next step we show how an absorbed spectrum can be used to calculate the corresponding assimilate production.

3.3 Calculation of Photosynthesis

Common photosynthesis models (PSM) use the integral of all absorbed wavelengths as photosynthetically active radiation (PAR) [Wm^{-2}] or photosynthetic photon flux density (PPFD) [µmol photons $\mathrm{m}^{-2}\mathrm{s}^{-1}$] as input to calculate assimilate production. The problem with this integration over all wavelengths is that the different wavelengths cannot be treated as having equal efficiency, since the absorbing pigments (e.g. chlorophyll *a* and *b*, beta-carotene) exhibit clear light spectrum-dependent differences in absorption efficiency (see above).

To our knowledge there is no spectral PSM available in the literature. The idea of a spectral PSM is to calculate the assimilates for each bucket independently.

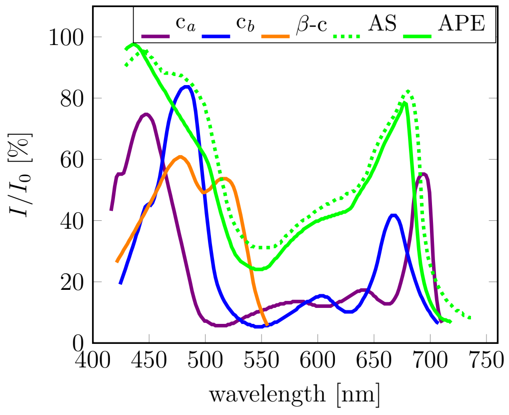


Figure 15. Measured photosynthesis rate as a function of absorbed wavelength. The photochemically active pigments determine the action spectrum and thus photochemical efficiency. Legend: c_a = chlorophyll a, c_b = chlorophyll b, β-c = β-carotene, AS = absorption spectrum = relative light absorption, APE = actual photochemical efficiency = action spectrum. I/I_0 refers to the radiation absorbed, transmitted or reflected relative to incident radiation at the same wavelength. Data taken from [27].

The actual photochemical efficiency (APE) from Figure 15 is normalised so that the integral is one. This provides the wavelength efficiency distribution. Then a common PSM is used to calculate assimilate production for the given temperature, leaf age and integral of absorbed power. The result of this calculation is used to scale the normalised APE, the latter then yielding a distribution that follows the

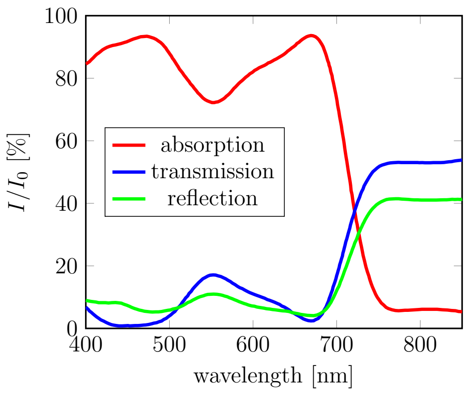


Figure 16. Absorption, reflection and transmission of light by a typical soybean leaf. I/I_0 refers to the radiation absorbed, transmitted or reflected relative to incident radiation at the same wavelength. Data taken from [26].

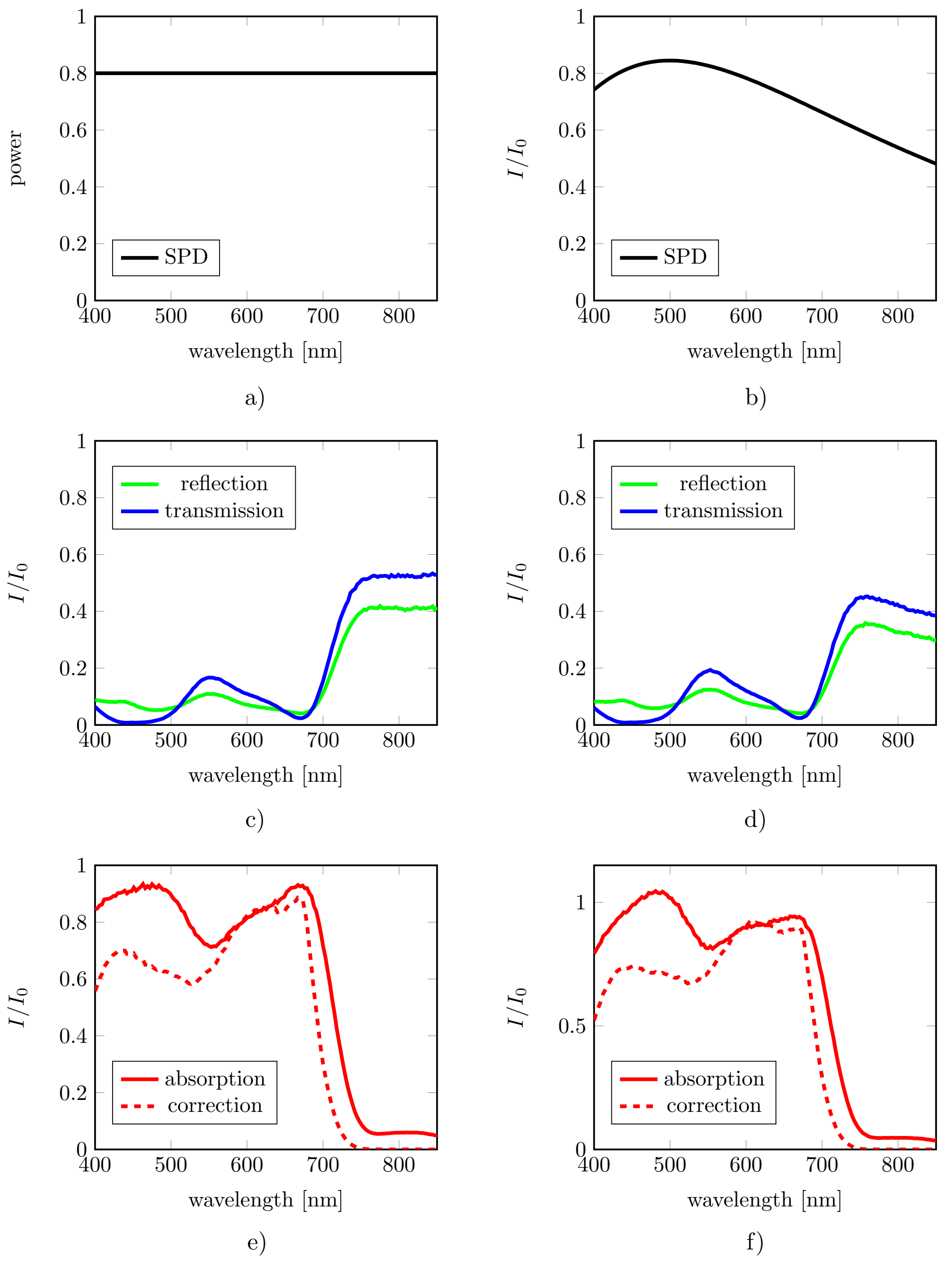


Figure 17. Direct comparison of two different input light spectra and their behaviour in the presence of the soybean leaf shader described above: a) constant spectrum (case a), b) simulation of sunlight (case b); c) and d) reflected and transmitted spectrum; e) and f) absorbed spectrum and McCree corrected spectrum.

APE. Finally, the intersection of the scaled APE and the absorbed spectrum are calculated. The intersection describes the maximal possible assimilate production at the absorbed spectrum according to the estimated assimilate production that was calculated by a common PSM. Figure 19 provides a schematic overview of the whole calculation.

In GroIMP, each object within the 3D scene can be "queried" for the amount of light it has absorbed at a given time step. According to the light model used and its parameterisation the results differ. For this example the spectral RT, called GPUFlux, was used, with 5 million rays emitted by a single light source. The observed spectrum had 80 buckets, ranging from 300 nm to 700 nm in 5 nm steps.

The amount of absorbed power A_p of an object x of the 3D scene is determined using a library method *getAbsorbedPowerMeasurement* of the light model (Code 2.1), which returns the spectrum as an array of absorbed radiation divided into buckets in $\mathrm{Wobj}^{-1}_{\mathrm{surface\_area}}\mathrm{bucket}^{-1}$.

For the calculation of the net photosynthesis A_n the model of Thornley [46] was used. As input parameter the integral of absorbed light converted into yielded photon flux (YPF) was used; a default temperature *temp* of 25 degrees C and a leaf age *age* of 30 days were assumed, the latter corresponding to full functionality.

Before the absorbed power could be used for further calculation it was necessary to consider photochemical efficiency. This required the distinction between different wavelength/buckets. The McCree Curve was used to weight PAR values according to the photosynthetic response at each wavelength and thus YPF approximated [36, 3]. The McCree Curve, also known as the Plant Sensitivity Curve, describes the action spectrum for an average plant. Figure 18 shows the graph of the McCree Curve in the range 325 nm to 775 nm.

4 DISCUSSION

In this work we demonstrated how at present commonly available computer hardware can be used to drastically reduce computation time for calculation of light fate in a 3D scene using inverted Monte Carlo raytracing. Furthermore, not only performance can be increased with our implementation but also the quality of light computation can be raised to a higher level by including information about the light spectrum. The simulation of natural light, including spectral information, opens up new possibilities for application in the domain of FSPM, such as realistic simulations of additional light sources, e.g. in greenhouses, climate chambers, or indoor farming, the latter in the context of urban horticulture, which will be playing an important role for local food production in the near future. Furthermore, the presented approach enables the modelling of an arbitrary diversity of specific light sources, e.g. High Pressure Sodium, Mercury Vapor, and LED lamps, in combination with commercially available reflectors and panels, e.g. double-ended wing, or raptor. The effect of certain wavebands on physiological processes is another domain in which the present model could be used: processes affecting plant growth or fruit quality

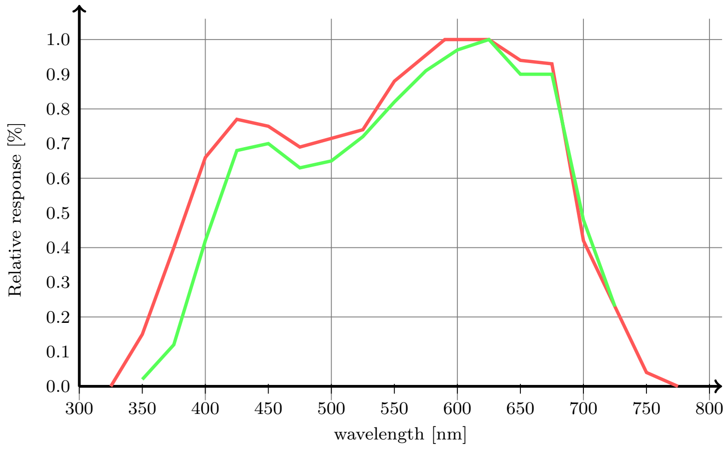


Figure 18. The relative quantum efficiency curve, also known as the McCree Curve, as determined by the average plant response for photosynthesis rate; Red: `http://en.wikipedia.org/wiki/Photosynthetically_active_radiation`. Green: Mean relative quantum yield of field plant species [36], Table IV.

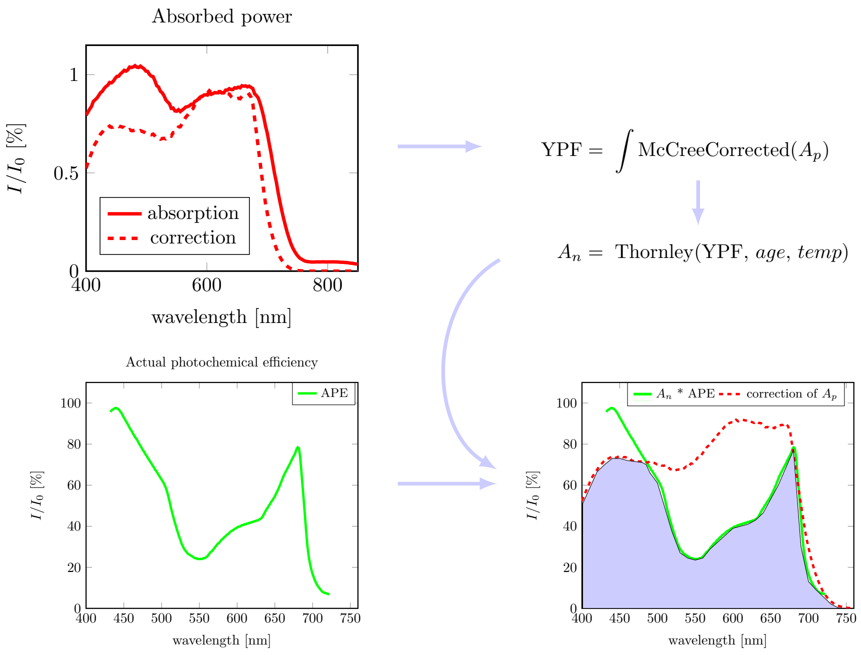

Absorbed power
I/I_0 [%]
absorption
correction
wavelength [nm]
$\mathrm{YPF} = \int \mathrm{McCreeCorrected}(A_p)$
$A_n = \mathrm{Thornley}(\mathrm{YPF}, age, temp)$
Actual photochemical efficiency
APE
A_n * APE
correction of A_p

Figure 19. Schematic overview of the spectral photosynthesis model. The grey-blue area of the lower right chart represents estimated assimilate production.

traits are often affected by the strength of certain light signals; the most common example is the ratio of the red to far-red waveband which is instantly translated in the plant into a ratio of phytochromes. This in turn can initiate complex processes such as etiolation or germination via transcription factors [9]. On a more practical note, differences and changes in light quality within a canopy can be influential for product quality, notably homogeneity or yield stability, e.g. the red colour of apple fruit which is due to sunlight induced anthocyan formation in the skin of the apple [18]. In crops that grow in height or in different layers the lower layers receive much less light, due to shading by upper structures. The lower parts benefit from a higher percentage of diffuse light, as this allows light rays to penetrate deeper into the canopy than direct light. However, the process of light scattering also goes along with a shift in the spectrum which might have an effect on the lower leaves yet which cannot be considered in models that neglect spectral information.

Acknowledgements

The authors thank Dietger van Antwerpen for the implementation of the GPU raytracer. The work presented in this paper has been partially funded by the "Pathfinder" project "LED – Realizing carbon footprint reduction in food and non-food chain by new productions systems based on LED lighting" in the frame of the Climate-KIC (Knowledge and Innovation Community). The financial support is gratefully acknowledged.

A LIST OF SYMBOLS

Abbreviation	Description	Unit
APE	actual photochemical efficiency	
AS	absorbed spectrum	
CPU	central processing unit	
FSPM	functional-structural plant model	
GPU	graphics processing unit	
LAD	leaf angle distribution	
LAI	leaf area index	
LED	light-emitting diode	
LM	light model	
LPI	leaf position index	
MC	Monte-Carlo	
MCRT	Monte-Carlo raytracer	
MSE	mean square error	
PAR	photosynthetically active radiation	$\mathrm{Wm^{-2}}$
PLD	physical light distribution	
PPFD	photosynthetic photon flux density	µmol photon $\mathrm{m^{-2}s^{-1}}$
PS	photosynthesis	
PSM	photosynthesis model	

RQMC	randomised quasi-Monte Carlo	
RT	raytracing / raytracer	
SPD	spectral power distribution	
YPF	yield photon flux	Wm^{-2}

B CODE

2.1 Light Model

Code 1: Setup of the Flux Light Model including the following steps:

```
import de.grogra.gpuflux.tracer.FluxLightModelTracer
    .MeasureMode;
import de.grogra.gpuflux.scene.experiment.
    Measurement;


const int RAYS = 10000000;
const int DEPTH = 10;
const FluxLightModel LM = new FluxLightModel(RAYS,
    DEPTH);


protected void init () {
  LM.setMeasureMode(MeasureMode.FULL_SPECTRUM);
  LM.setSpectralBuckets(81);
  LM.setSpectralDomain(380, 780);
}
```

1. import needed classes
2. initialisation: with 10 million rays and a recursion depth of 10
3. parameterisation: 400 nm divided into 80 buckets of 5 nm

Code 2: Run the light model and determine the amount of sensed radiation or of absorbed power for an object-type. Steps:

```
public void run () [
  {
    LM.compute();
  }

  x:SensorNode ::> {
    Measurement spectrum = LM.
        getSensedIrradianceMeasurement(x);
    float absorbedPower = spectrum.integrate();
    ...
  }

  x:Box ::> {
    Measurement spectrum = LM.
        getAbsorbedPowerMeasurement(x);
    float absorbedPower = spectrum.integrate();
    ...
  }
]
```

1. computation: runs the light model
2. evaluation of sensor objects
3. evaluation of objects of type *Box*

2-3 and integrate the whole absorbed spectrum

Code 3: Demonstration of the method for the determination of the amount of absorbed power per bucket or integration over a certain spectral range.

```
  Measurement spectrum = LM.
      getAbsorbedPowerMeasurement(x);

  //absorbed power for the first bucket: 380-385 nm
  float ap380_385 = spectrum.data[0];

  //accumulate absorbed power for the first four 50
        nm buckets
  float b0 = 0, b1 = 0, b2 = 0, b3 = 0;
  for (int i:(0:10)) {
     b0 += spectrum.data[i];
     b1 += spectrum.data[i + 10];
     b2 += spectrum.data[i + 20];
     b3 += spectrum.data[i + 30];
  }

  //integrate the whole spectrum
  float ap = spectrum.integrate();
```

1. obtain the absorbed spectrum
2. store the first bucket in a variable
3. build four integrals, each of 50 nm (10 buckets of 5 nm) and sum up the first 40 buckets
4. calculate the integral over the whole spectrum

2.2 Light Sources

Code 4: Parameterisable light node – with explicit definition of physical light and spectral power distribution by simple arrays.

```
import de.grogra.imp3d.spectral.
    IrregularSpectralCurve;

const double[] [] DISTRIBUTION = {
  {131.25, 131.67, 132.37, ...},
  {131.36, 131.81, ...},
  ...
};

static const float[] WAVELENGTHS = {380,385, ...};
static const float[] AMPLITUDES = {0.000967,
    0.000980, ...};

module MyLamp extends LightNode() {
  {
    setLight(new SpectralLight(
      new IrregularSpectralCurve(WAVELENGTHS,
          AMPLITUDES)).(
        setPower(10), //[W]
        setLight(new PhysicalLight(DISTRIBUTION)));
  }
}
```

1. import of required classes
2. definition of the physical light distribution
3. definition of the spectral power distribution
4. definition of a lamp using the specified parameters

Code 5: Parameterizable light node – using file references for physical and spectral power distribution.

```
const LightDistributionRef DISTRIBUTION = light("
    distribution1");
const SpectrumRef SPECTRUM = spectrum("equal");

module MyLamp1 extends LightNode {
 {
   setLight(new SpectralLight(new PhysicalLight(
       DISTRIBUTION), SPECTRUM, 10)); // 10 W
 }
}

module MyLamp2 extends LightNode {
  {
    setLight(
      new SpectralLight().(
        setPower(10), //[W]
        setLight(new PhysicalLight(DISTRIBUTION)),
        setSpectrum(SPECTRUM)));
  }
}
```

1. Definition of a file reference. This file can be included or linked to a project.
2. This reference can be applied to a concrete lamp via the constructor,
3. or by using the set-methods of the *LightNode* class.

2.3 Illumination Model

Code 6: Definition of a Phong shader by textures, by colours and by a user-defined spectral power distribution.

```
  Phong myShader = new Phong();
  ImageMap image = new ImageMap();
  image.setImageAdapter(new FixedImageAdapter(image(
      "leaf").toImageAdapter().getBufferedImage()));
  myShader.setDiffuse(image);

  Phong myShader = new Phong();
  myShader.setDiffuse(new RGBColor(0,1,0));
  //myShader.setSpecular(new Graytone(0.5));
  //myShader.setShininess(new Graytone(0.5));
  myShader.setDiffuseTransparency(new RGBColor
      (0.5,0,0));
  //myShader.setAmbient(new Graytone(0.5));
  //myShader.setEmissive(new Graytone(0.5));

   ChannelSPD MySPD = new ChannelSPD(new
       IrregularSpectralCurve(
     new float[] {400,410, ....,740,750} //
         WAVELENGTHS,
     new float[] {0.1,0, .... ,0.4,0.25} //
         AMPLITUDES
   ));
  Phong myShader = new Phong();
  myShader.setDiffuse(MySPD);
```

1. Set an image as texture: Values for reflection depend on the colour of the texture at each pixel of the image.
2. Set specific properties: At this configuration green will be reflected to 100 % and red will be transmitted to 50 % for the whole surface of the object.
3. A user-defined SPD is used to define the diffuse colour.

C CALCULATIONS

3.1 Conversion: Candela $\Longleftrightarrow$ Watt

Candela (cd) indicates the brightness of a light in a given direction. It is defined as lumen lm per steradian sr. A steradian is the standard unit solid angle in three dimensions. At a given wavelength of 555 nm one candela can be approximated by 1/683 watt. For an arbitrary wavelength λ the luminous intensity $I_v(\lambda)$ can be obtained by Equation (1):

$$I_v(\lambda) = 683.002 * \bar{y}(\lambda) * I_e(\lambda) \tag{1}$$

where $\bar{y}(\lambda)$ is the luminous intensity function by Sharpe [43], which describes the visual perception of brightness by the human eye (Figure 20) between 390 and 830 nm.

To estimate the total power over a defined range of wavelengths $[w_{lower}, w_{upper}]$, the luminous intensity function needs to be integrated over the whole spectrum of wavelengths.

$$\int_{\lambda=w_{lower}}^{w_{upper}} I_v(\lambda) \tag{2}$$

with the approximation Equation (3) using n nm buckets:

$$\sum_{\lambda=w_{lower}}^{w_{upper}} I_v(\lambda), \quad \lambda \equiv 0 \pmod{n} \tag{3}$$

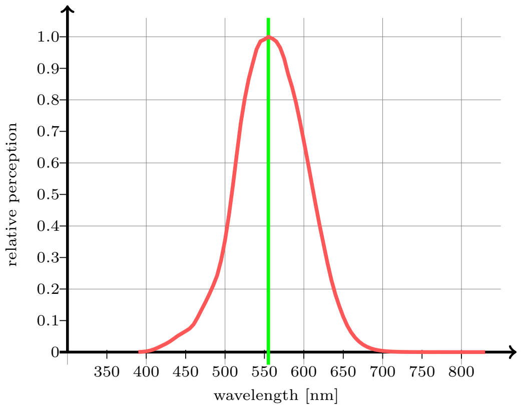


Figure 20. Luminous intensity function by Sharpe [43], which describes the visual perception of brightness by a human eye. The human eye is most sensitive to green light at a wavelength of 555 nm.

The values for $I_e(\lambda)$ can be obtained directly for each object *obj* by calling the spectral raytracer (getAbsorbedPowerMeasurement(*obj*)).

REFERENCES

[1] Alabadí, D.—Gil, J.—Blázquez, M. A.—García-Martínez, J. L.: Gibberellins Repress Photomorphogenesis in Darkness. Plant Physiology, Vol. 134, 2004, No. 3, pp. 1050–1057, doi: 10.1104/pp.103.035451.

[2] Angel, E.: Interactive Computer Graphics – A Top-Down Approach Using OpenGL. 4th ed. Addison-Wesley, 2006.

[3] Barnes, C.—Tibbitts, T.—Sager, J.—Deitzer, G.—Bubenheim, D.—Koerner, G.—Bugbee, B.: Accuracy of Quantum Sensors Measuring Yield Photon Flux and Photosynthetic Photon Flux. HortScience, Vol. 12, 1993, No. 28, pp. 1197–1200.

[4] Born, M.—Wolf, E.: Principles of Optics. Cambridge University Press, Cambridge, 1999, pp. 14–24.

[5] Brown, C. S.—Schuerger, A. C.—Sager, J. C.: Growth and Photomorphogenesis of Pepper Plants under Red Light-Emitting Diodes with Supplemental Blue or Far-Red Lighting. Journal of the American Society for Horticultural Science, Vol. 120, 1995, No. 5, pp. 808–813.

[6] Buck-Sorlin, G. H.: Functional-Structural Plant Modeling. In: Dubitzky, W., Wolkenhauer, O., Cho, K.-H., Yokota, H. (Eds.): Encyclopedia of Systems Biology. Springer New York, 2013, pp. 778–781, doi: 10.1007/978-1-4419-9863-7_1479.

[7] Buck-Sorlin, G. H.—de Visser, P. H. B.—Henke, M.—Sarlikioti, V.—van der Heijden, G. W. A. M.—Marcelis, L. F. M.—Vos, J.: Towards a Functional-Structural Plant Model of Cut-Rose: Simulation of Light Environment, Light Absorption, Photosynthesis and Interference with the Plant Structure. Annals of Botany, Vol. 108, 2011, No. 6, pp. 1121–1134, doi: 10.1093/aob/mcr190.

[8] Buck-Sorlin, G. H.—Hemmerling, R.—Vos, J.—de Visser, P. H. B.: Modelling of Spatial Light Distribution in the Greenhouse: Description of the Model. In: Li, B. et al. (Eds.): Proceedings of the Third International Symposium on Plant Growth Modeling, Simulation, Visualization and Applications (PMA '09), Beijing, China, November 2009. IEEE, Los Alamitos 2010, pp. 79–86.

[9] Casal, J. J.—Candia, A. N.—Sellaro, R.: Light Perception and Signalling by Phytochrome A. Journal of Experimental Botany, Vol. 65, 2014, pp. 2835–2845, doi: 10.1093/jxb/ert379.

[10] Chelle, M.—Hanan, J. S.—Autret, H.: Lighting Virtual Crops: The CARIBU Solution for Open L-Systems. In: Godin, C. et al. (Eds.): Proceedings of the Fourth International Workshop on Functional-Structural Plant Models (FSPM '04), Montpellier, France, June 07–11, 2004. UMR AMAP, Montpellier, 2004, pp. 194–194.

[11] Chelle, M.—Andrieu, B.: The Nested Radiosity Model for the Distribution of Light within Plant Canopies. Ecological Modelling, Vol. 111, 1998, No. 1, pp. 75–91.

[12] Cieslak, M.—Lemieux, C.—Hanan, J. S.—Prusinkiewicz, P.: Quasi-Monte Carlo Simulation of the Light Environment of Plants. Functional Plant Biology, Vol. 35, 2008, No. 10, pp. 837–849, doi: 10.1071/FP08082.

[13] Cournède, P.-H.—Mathieu, A.—Houllier, F.—Barthélémy, D.—de Reffye, P.: Computing Competition for Light in the GreenLab Model of

Plant Growth: A Contribution to the Study of the Effects of Density on Resource Acquisition and Architectural Development. Annals of Botany, Vol. 101, 2008, No. 8, pp. 1207–1219.

[14] Dauzat, J.: Radiative Transfer Simulation on Computer Models of Elaeis Guineensis. Oléagineux (Paris), Vol. 49, 1994, No. 3, pp. 81–90.

[15] Dauzat, J.—Franck, N.—Rapidel, B.—Luquet, D.—Vaast, P.: Simulation of Ecophysiological Processes on 3D Virtual Stands with the Archimed Simulation Platform. Second International Symposium on Plant Growth Modeling and Applications (PMA '06), November 2006, pp. 101–108, doi: 10.1109/PMA.2006.52.

[16] Dauzat, J.—Clouvel, P.—Luquet, D.—Martin, P.: Using Virtual Plants to Analyse the Light-Foraging Efficiency of a Low-Density Cotton Crop. Annals of Botany, Vol. 101, 2008, No. 8, pp. 1153–1166.

[17] Disney, M. I.—Lewis, P.—North, P. R. J.: Monte Carlo Ray Tracing in Optical Canopy Reflectance Modelling. Remote Sensing Reviews, Vol. 18, 2000, No. 2-4, pp. 163–196, doi: 10.1080/02757250009532389.

[18] Feng, F. J.—Li, M. J.—Ma, F. W.: The Effects of Bagging and Debagging on External Fruit Quality, Metabolites, and the Expression of Anthocyanin Biosynthetic Genes in 'Jonagold' Apple (Malus domestica Borkh.). Scientia Horticulturae, Vol. 165, 2014, pp. 123–131, doi: 10.1016/j.scienta.2013.11.008.

[19] Furuya, M.—Schäfer, E.: Photoperception and Signalling of Induction Reactions by Different Phytochromes. Trends in Plant Science, Vol. 1, 1996, No. 9, pp. 301–307.

[20] Goins, G. D.—Yorio, N. C.—Sanwo, M. M.—Brown, C. S.: Photomorphogenesis, Photosynthesis, and Seed Yield of Wheat Plants Grown under Red Light-Emitting Diodes (LEDs) with and without Supplemental Blue Lighting. Journal of Experimental Botany, Vol. 48, 1997, No. 7, pp. 1407–1413.

[21] Greene, N.: Voxel Space Automata: Modeling with Stochastic Growth Processes in Voxel Space. ACM SIGGRAPH Computer Graphics – Special Issue: Proceedings of the 1989 ACM SIGGRAPH Conference, Vol. 23, 1989, No. 3, pp. 175–184.

[22] GroIMP Developer Group, Web site. Available on: `http://www.grogra.de/`, 2015.

[23] Hemmerling, R.—Kniemeyer, O.—Lanwert, D.—Kurth, W.—Buck-Sorlin, G. H.: The Rule-Based Language XL and the Modelling Environment GroIMP Illustrated with Simulated Tree Competition. Functional Plant Biology, Vol. 35, 2008, pp. 739–750, doi: 10.1071/FP08052.

[24] Illuminating Engineering Society, ANSI/IESNA LM-63-02. IES-Specification, Standard File Format for the Electronic Transfer of Photometric Data. Available on: `http://www.cn-hopu.com/upload/file/IES.pdf`, 2002.

[25] Kanamaru, N.—Chiba, N.—Takahashi, K.—Saito, N.: CG Simulation of Natural Shapes of Botanical Trees Based on Heliotropism. The Transactions of the Institute of Electronics, Information, and Communication Engineers J75-D-II, 1992, pp. 76–85 (in Japanese).

[26] Kasperbauer, M. J.: Far-Red Light Reflection from Green Leaves and Effects on Phytochrome-Mediated Assimilate Partitioning under Field Conditions. Plant Physiology, Vol. 85, 1987, No. 2, pp. 350–354, doi: 10.1104/pp.85.2.350.

[27] Karp, G.: Cell and Molecular Biology: Concepts and Experiments. 7th ed. John Wiley and Sons, 2013.

[28] Khronos Group – OpenCL, Web site. Available on: `https://www.khronos.org/opencl/`, 2015.

[29] Kniemeyer, O.: Rule-Based Modelling with the XL/GroIMP Software. In: Schaub, H., Detje, F., Brüggemann, U. (Eds.): The Logic of Artificial Life, Proceedings of 6th GWAL. AKA Akademische Verlagsges. Berlin, 2004, pp. 56–65.

[30] Kniemeyer, O.: Design and Implementation of a Graph Grammar Based Language for Functional-Structural Plant Modelling. Ph.D. thesis, BTU Cottbus. Available on: `http://opus.kobv.de/btu/volltexte/2009/593/`, 2008.

[31] Kurth, W.—Buck-Sorlin, G. H.—Kniemeyer, O.: Relationale Wachstumsgrammatiken: Ein Formalismus zur Spezifikation multiskalierter Struktur-Funktions-Modelle von Pflanzen. In: Modellierung Pflanzlicher Systeme aus Historischer und Aktueller Sicht. Symposium zu Ehren von Prof. Dr. Dr. h.c. Eilhard Alfred Mitscherlich, Schriftenreihe des Landesamtes für Verbraucherschutz, Landwirtschaft und Flurneuordnung Brandenburg, Reihe Landwirtschaft, Band 7, 2006, pp. 36–45 (in German).

[32] Kurth, W.: Specification of Morphological Models with L-Systems and Relational Growth Grammars. Computational Visualistics and Picture Morphology (Themenheft zu IMAGE 5), Vol. 5, 2007, No. 1, pp. 50–79.

[33] Le, Y. G.: Schema Validation with Intel® Streaming SIMD Extensions 4 (Intel® SSE4), White Paper, December 16, 2008.

[34] Leroy, C.—Sabatier, S.—Wahyuni, N.—Barczi, J.-F.—Dauzat, J.—Laurans, M.—Auclair, D.: Virtual Trees and Light Capture: A Method for Optimizing Agroforestry Stand Design. Agroforestry Systems, Vol. 77, 2009, No. 1, pp. 37–47.

[35] Massonnet, C.: Functional and Architectural Variability of Vegetative System in Apple Tree (Malus Domestica): Comparison Between Four Cultivars by a Functional-Structural Modelling Approach. Thesis, Ecole Nationale Superieure Agronomique de Montpellier – AGRO M, December 2004. Available on: `https://tel.archives-ouvertes.fr/tel-00010748`.

[36] McCree, K. J.: The Action Spectrum, Absorbance and Quantum Yield of Photosynthesis in Crop Plants. Agricultural Meteorology, Vol. 9, 1971/72, pp. 191–216.

[37] Měch, R.: Modeling and Simulation of the Interaction of Plants with the Environment Using L-Systems and Their Extensions. Ph.D. thesis, University of Calgary, Calgary, Alberta, Canada, 1998.

[38] Měch, R.—Prusinkiewicz, P.: Visual Models of Plants Interacting with Their Environment. Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '96), New York, USA, 1996, pp. 397–410, doi: 10.1145/237170.237279.

[39] Owens, J. D.—Houston, M.—Luebke, D.—Green, S.—Stone, J. E.—Phillips, J. C.: GPU Computing. Proceedings of the IEEE, Vol. 96, 2008, No. 5, pp. 879–899.

[40] Perttunen, J.—Sievänen, R.—Nikinmaa, E.: LIGNUM: A Model Combining the Structure and the Functioning of Trees. Ecological Modelling, Vol. 108, 1998, No. 1–3, pp. 189–198, doi: 10.1016/S0304-3800(98)00028-3.

[41] Phong, B. T.: Illumination for Computer Generated Pictures. Communications of the ACM, Vol. 18, 1975, No. 6, pp. 311–317.

[42] Pradal, C.—Dufour-Kowalski, C.—Boudon, F.—Fournier, C.—Godin, C.: OpenAlea: A Visual Programming and Component-Based Software Platform for Plant Modelling. Functional Plant Biology, Vol. 35, 2008, No. 9/10, pp. 751–760, doi: 10.1071/FP08084.

[43] Sharpe, L. T.—Stockman, A.—Jagla, W.—Jägle, H.: A Luminous Efficiency Function, $v * (\lambda)$, for Daylight Adaptation. Journal of Vision, Vol. 5, 2005, No. 11, pp. 948–968.

[44] Sievänen, R.—Perttunen, J.—Nikinmaa, E.—Kaitaniemi, P.: Toward Extension of a Single Tree Functional-Structural Model of Scots Pine to Stand Level: Effect of the Canopy of Randomly Distributed, Identical Trees on Development of Tree Structure. Functional Plant Biology, Vol. 35, No. 10, 2008, pp. 964–975, doi: 10.1071/FP08077.

[45] Taugourdeau, O.—Dauzat, J.—Griffon, S.—de Coligny, F.—Sabatier, S.—Caraglio, Y.—Barthélémy, D.: Retrospective Analysis of Fir Sapling Growth vs. Light Interception. In: DeJong, T., Da Silva, D. (Eds.): Proceedings of the 6th International Workshop on Functional-Structural Plant Models (FSPM '10), University of California, Davis, USA, November 12–17, 2010, pp. 93–95.

[46] Thornley, J. H. M.: A Model to Describe the Partitioning of Photosynthate During Vegetative Plant Growth. Annals of Botany, Vol. 36, 1972, pp. 419–430, doi: 10.1093/oxfordjournals.aob.a084601.

[47] van Antwerpen, D. G.: Unbiased Physically Based Rendering on the GPU. M.Sc. thesis, TU Delft, 2011. Available on: `http://graphics.tudelft.nl/~dietger/thesis/index.htm`.

[48] Vos, J.—Evers, J. B.—Buck-Sorlin, G. H.—Andrieu, B.—Chelle, M.—de Visser, P. H. B.: Functional-Structural Plant Modelling: A New Versatile Tool in Crop Science. Journal of Experimental Botany, Vol. 61, 2010, No. 8, pp. 2101–2115.

[49] Wang, H. Y.—Kang, M. Z.—Hua, J: Simulating Plant Plasticity under Light Environment: A Source-Sink Approach. Proceedings of the Fourth International Symposium on Plant Growth Modeling, Simulation, Visualization and Applications (PMA '12), Shanghai, China, October 31–November 3, 2012, IEEE, pp. 431–438, doi: 10.1109/PMA.2012.6524869.

Michael Henke – Diploma degree in computer science from the Brandenburg University of Technology Cottbus — 2009–2010 Visiting Scholar at Zhejiang University, Hangzhou, China — 2013 Researcher at French National Institute for Agricultural Research — 2014–2016 Researcher at Wageningen UR, The Netherlands — 2017 Ph.D. in computer science at the University of Göttingen, Department Ecoinformatics, Biometrics and Forest Growth — since 2017 Postdoctoral scientist at the Institute of Plant Genetics and Crop Plant Research in Gatersleben, Germany — Assistant Lecturer at Brandenburg University of Technology Cottbus and University of Göttingen — GroIMP developer — research fields: functional-structural plant modelling, simulation, biometrics: data acquisition, image analysis.

Gerhard H. Buck-Sorlin – Diploma degree in biology at the University of Göttingen and Ph.D. in biology at the University of Wales in Bangor, UK (1997), subsequently junior postdoctoral scientist at the Institute of Plant Genetics and Crop Plant Research in Gatersleben, Germany — 2002–2007 Senior Scientist and Lecturer at the University of Technology in Cottbus — 2005–2009 Guest Professor at the Zhejiang University in Hangzhou — 2007–2011 Senior Scientist and Lecturer at Wageningen University and Research Centre — since 2011 Professor for fruit tree culture and modelling at Agrocampus Ouest, Centre d'Angers, France — research fields: ecophysiology of crop plants, functional-structural plant modelling.