TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

Implementation of simulation-based virtual sensors using Radiance

MASTER THESIS

Vienna University of Technology

Institute of Building Physics and Building Ecology

Univ. Prof. Dr. Ardeshir Mahdavi

WS 2013

Vaibhav Jain

Matriculation Number: 1129826

Master Building Science and Technology

Vienna, June 2014

Table of Contents

List of figures:

List of tables:

IMPLEMENTATION OF SIMULATION-BASED VIRTUAL SENSORS USING RADIANCE

1 ABSTRACT

Building performance simulation is traditionally used to support the building design process; however, they can also be used during building operation phase by providing simulation output as control parameter for building control and/or monitoring systems. Such a simulation output from a virtual (or computational) model of a building are hereafter referred as output from virtual sensor.

Theoretically, virtual sensors, if reliably and effectively incorporated, can expand the reach of real sensors (Zach, Glawischnig, & Mahdavi). A continuous supply of output from such sensors can support real-time (or near real-time) monitoring of simple environmental conditions such as air temperature, relative humidity etc. as well as complex performance indicators such as thermal comfort, visual glare etc. This information, especially in big multi-zone buildings, will otherwise be very costly to obtain from real sensors due to high initial costs of the sensors, their installation, networking, and maintenance. Virtual sensors can easily increase the resolution of sensory information, since they have no monetary or spatial limitations. For example, a simulation model can have as many costly glare sensors which can be placed where ever desired.

However, practically the accuracy of virtual sensors poses a challenge, especially in cases where high uncertainty is involved, such as daylight illuminance calculation. For accurate results (which mean being reasonably close to measured values from real sensors) well calibrated models of the building as well as its boundary conditions are required.

This study will (a) describe the concept of virtual sensors, (b) implement this concept by developing a software application that automates the process of continuously producing real-time output from illuminance type virtual sensors, (c) create a calibrated model of a building situated in Vienna using *Radiance* lighting simulation program to test the developed software application, (d) discuss results, and challenges pertaining to the implementation.

A comparison of measured and simulated output from the current version of developed application (*Radiance* based) was performed, and was found to give reliable results on clear sky days, but moderately accurate results on intermediate sunny/cloudy days. Results for artificial lighting case

had good accuracy. The use of sky-scanner is recommended so that the simulation sky has the luminance distribution of the actual. Due to the modular architecture of the developed application, data from precise measurement instruments such as sky-scanner can be included in the future version of the application.

2 INTRODUCTION

2.1 Simulation based control

One of the earlier studies on simulation based control strategies encouraged the use of multi-aspect simulation model of the building, which exchanges information with the management system of the real building, while running parallel to it, in order to capture the dynamic behavior of the building. Simulation-based control strategies, along with numerous benefits, allow continuous monitoring of different performance indicators pertaining to indoor environment with reduced number of real sensors. Moreover, it can also reduce the quantity of required sensor units (Mahdavi, 2001).

2.2 Virtual sensors

The idea to implement the concept of simulation-based virtual sensors was conceived during the MOST research project (Zach R. , Glawischnig, Hönisch, Appel, & Mahdavi, 2012). In this project a Java based open-source, vendor and technology independent toolkit was developed for building monitoring, data preprocessing, and visualization. This toolkit required to obtain information about simple environmental conditions such as air temperature, relative humidity etc., as well as complex performance indicators such as thermal comfort, visual glare etc. for the building. It was soon realized that due to various reasons related to technical constraints and cost considerations, there are limitation to the deployment of real sensors. These limitations may due to their number, types, quality, locations, installation, networking, etc. On the other hand, simulation-based virtual sensors are free from such limitations. Virtual sensors if reliably and effectively incorporated, can not only expand the reach of real sensors but could also support the pervasive and continuous monitoring of complex performance indicators that could be otherwise very expensive to monitor (Zach, Glawischnig, & Mahdavi).

A real building has both static characteristics, and dynamic behaviors such as construction, material properties, and solar load, occupant's behavior respectively. If a simulation model, after calibration of static characteristics, can effectively incorporate the dynamic behavior of buildings then it could behave reliably close to real building (Tahmasebi & Mahdavi, 2012). For example, once the simulation model has been calibrated, using existing real sensor data, to match building's construction and material properties, then the state (On/Off) of individual lights, which is a constituent of the dynamic behavior, can be automatically passed from building automation system to the simulation model without any human intervention. Similar concepts have been proposed in a few studies (Mahdavi et.al, 2001), (Mahdavi, 2001), however implementation and testing of this concept is done in this study. As a proof of concept, a supervisory software application is developed in Java programming language which (1) inputs a calibrated Radiance model (2) updates the state of indoor artificial lights in the model using state information from building management system (BMS), (3) updates the sky model using current onsite irradiance data from weather station, (4) combines building and sky model to create a scene, (5) runs Radiance simulation, (6) displays output illuminance values on monitoring screen (part of application's GUI), (7) as well as stores them in database for the utilization of building control system, if any. This is performed in a repetitive manner to produce a continuous supply of near real-time illuminance values.

Before proceeding further it is important to describe the user of developed application, and API. The user of the developed application could be a person related to building facility management who is versed with the sensor network and database technology within the building. The user of the developed API could be a person related to the development of software modules for building monitoring and control services, or a researcher. In any of the case, it is beneficial for the user to have basic understanding of *Radiance* modeling and simulation (particularly ambient parameter optimization for inter-reflections) for fault detection in modeling, and minimization of computational cost during simulation.

2.3 Radiance simulation engine

Radiance is currently a mature and popular ray tracing program package that performs accurate and reliable daylighting and lighting simulations. It is widely established in the research community and has been extensively validated in many projects such as (Mardaljevic, 1999), (Compagnon, 1997). It uses backward ray tracing algorithm to calculate inter-reflections by utilizing a hybrid deterministic/stochastic sampling approach which makes it reasonably accurate and also computationally efficient (Ward G. J., 1994). *Radiance* offers great flexibility which allows user to specify almost all the input parameter. Due to this it has a steep learning curve, and any validation study depends more upon the experience of the user with the program. Therefore, it is suitable and important to point out the famous computer science analogy of garbage in, garbage out (GIGO). The developed application, *RealTimeRadiance*, uses *Radiance* simulation engine to perform daylight as well as artificial lighting simulations.

3 METHODOLOGY

3.1 Application development and its architecture

The developed application, *RealTimeRadiance*, is intended to implement different kinds of *Radiance* based virtual sensors such as illuminance, luminance, glare, etc. However, as a proof of concept only illuminance type virtual sensors have been implemented in this study. Since the application may include more types of virtual sensors in the future, therefore its architecture needed to be modular, where each type of virtual sensor may represent a module (a class or collection of classes). Java is a class-based, object-oriented programming language that incorporates Modularity software design technique which allows separating the logical concern of the application code into separate independent modules, thus, making the application more testable, reusable, and maintainable (Blewitt, 2009). Different modules are typically incorporated in the program through implementing Java interfaces. This means that, as and when required more types of virtual sensors (i.e. their modules) can be developed and incorporated into *RealTimeRadiance*.

3.1.1 Software requirements

Figure 1 is a context diagram, which shows an abstract and high-level representation of application's interaction with external entities. The arrowheads show the direction of information flow. Whenever the user requests illuminance values, the application, interacts with the external systems to update the calibrated building model, so that it closely represents the real world building and its current boundary conditions, and then returns back the illuminance values at specified locations as output from the simulation.

Figure 2 shows the requirements of the application as different use-cases in UML notation. The first use-case is to show illuminance values, calculated at different locations, on the GUI (graphical user interface) for user's supervision; the second use-case is to store these calculated illuminance values in the database so that they can be used by any other service such as building monitoring or control; the third use-case is to visualize the interior of the model so that any possible modeling error can be identified and hence corrected. All the three use-cases require a combined simulation model (hereafter referred as updated scene) which constitutes calibrated building model and its boundary conditions such as current state of light and sky luminance distribution (or sky model). The current real time state of light is obtained from BMS, and current sky luminance distribution is calculated using the Perez All Sky model (Perez, Seals, & Michalsky, 1993) that takes input of solar and sky irradiance values which are obtained from on-site weather station. The development of calibrated building and sky model is discussed in 4.2.

To understand this whole process in more detail, let us look at Figure 2 which shows the use-case diagram of the application. It is clear from this diagram that updated calibrated building and sky model is required for all the three use cases. The simulation model needs to be continuously updated to represent the current real time boundary conditions.

The application, firstly, imports the provided calibrated building simulation file, secondly, it checks the current state of internal artificial lights within the real-building by communicating with the BMS, thirdly, it creates a sky luminance distribution model using the current irradiance values obtained from the on-site weather station, and fourthly, it runs the simulation and calculates illuminance, and lastly it displays and stores the results on UI and database respectively.

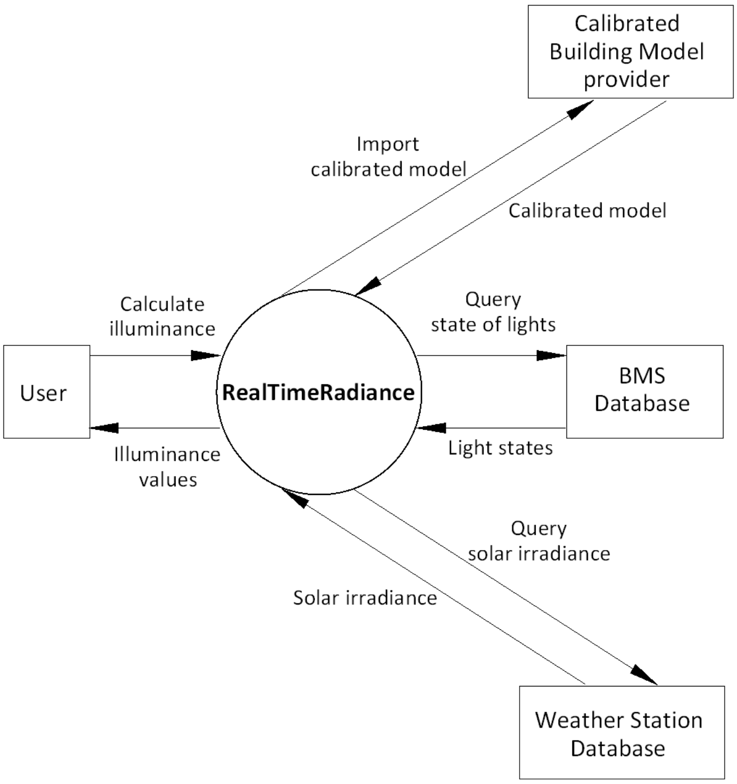


Figure 1: Context diagram showing data flow between the application and external entities

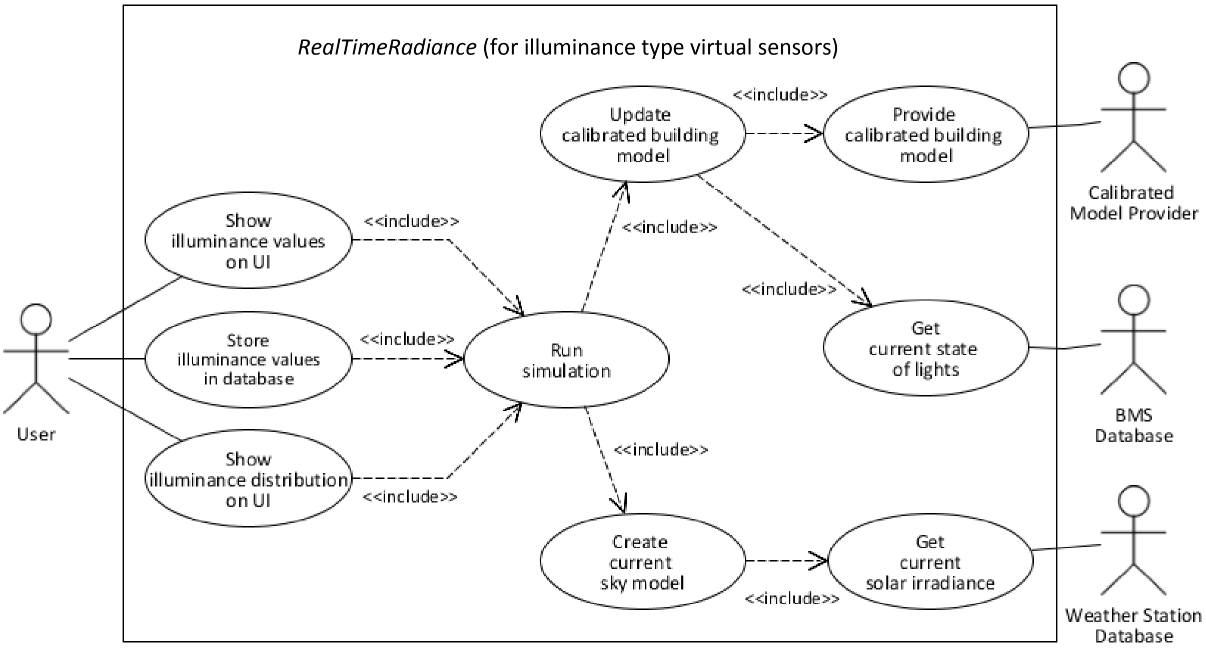

RealTimeRadiance (for illuminance type virtual sensors)
Update calibrated building model
<<include>>
Provide calibrated building model
Calibrated Model Provider
Show illuminance values on UI
<<include>>
<<include>>
<<include>>
Get current state of lights
Store illuminance values in database
<<include>>
Run simulation
BMS Database
User
Show illuminance distribution on UI
<<include>>
<<include>>
Create current sky model
Get current solar irradiance
<<include>>
Weather Station Database

Figure 2: UML Use-Case diagram of the application

To calculate real-time illuminance, *RealTimeRadiance* utilizes Java's multi-threading (concurrency) capability and simultaneously executes three methods (a Java notation for processes/sub-processes). In the first method, a calibrated *Radiance* model of the building is imported into the "temp" (temporary) folder/directory of the application. Simultaneously, the second method reads the state (On/Off) of lights in the building management system's (BMS) database, and accordingly updates the state of corresponding lights in the *Radiance* model. The third method, firstly, reads the diffuse and global irradiance from the weather station's database, and then calculates direct normal irradiance using method described in 4.2.3.1. The solar zenith angle is obtained from a publicly available Java library (Brunner, 2013), which utilizes the algorithm described in (Blancol, Alarcón, López, & Lara, 2001). The resulting solar zenith angle from this Java library were compared and checked with other online calculators, and were found to be accurate and reliable.

3.2 Application Programming Interface (API) development

3.2.1 Class-diagram of the developed application

An application-programming interface (API) is a particular set of programming instructions and standards for accessing a software application or library. It is a software-to-software interface, not a user interface (UI). With APIs, applications or programs communicate with each other without any user intervention (Orenstein, 2000)

As this study does not intend to be a treatise on Java programing; therefore, the reader is encouraged to read (Metsker & Wake, 2006), (Larman, 2004), (Arnold, Gosling, & Holmes, 2006) to have a better understanding on the concepts of object-oriented programming. However, it will attempt to describe the flow of information between the classes, as shown in the class diagram Figure 3, in order to aid the user of the developed API.

Figure 3 shows the UML Class diagram of the application which shows the relationship between the classes as well as the signature of the class fields (attributes and methods). All the data required to create the updated scene for simulation is obtained from the AllInputDataImplementation.java class which implements AllInputData.java interface. Once all the required input data is ready, the user needs to (1) create an instance of MainClass.java using its constructor and pass the input data as its parameter, and (2) execute the runIllumSimulation() method to run *Radiance* and calculate illuminance using so called virtual sensors. The class constructor initializes the different data structures (List) which are declared as private class fields to store the input data. This class is called MainClass since it is the most important class as it holds instances of all the data structure which hold the input data. This becomes convenient and manageable because only a single instance of MainClass.java class needs to be passed to all the other classes that execute and interface with external simulation engine, which is *Radiance* in this case. Singleton design pattern is used to allow only one instantiation of MainClass.

The next section describes the development of the GUI using JavaFX. However, the developed API can also be used separately to develop GUI in other Java based architectural style/platform such as RESTful API. It should be noted that the developed API is intended to be used as a module for *Radiance* based virtual sensors in the MOST project (Zach, Glawischnig, & Mahdavi).

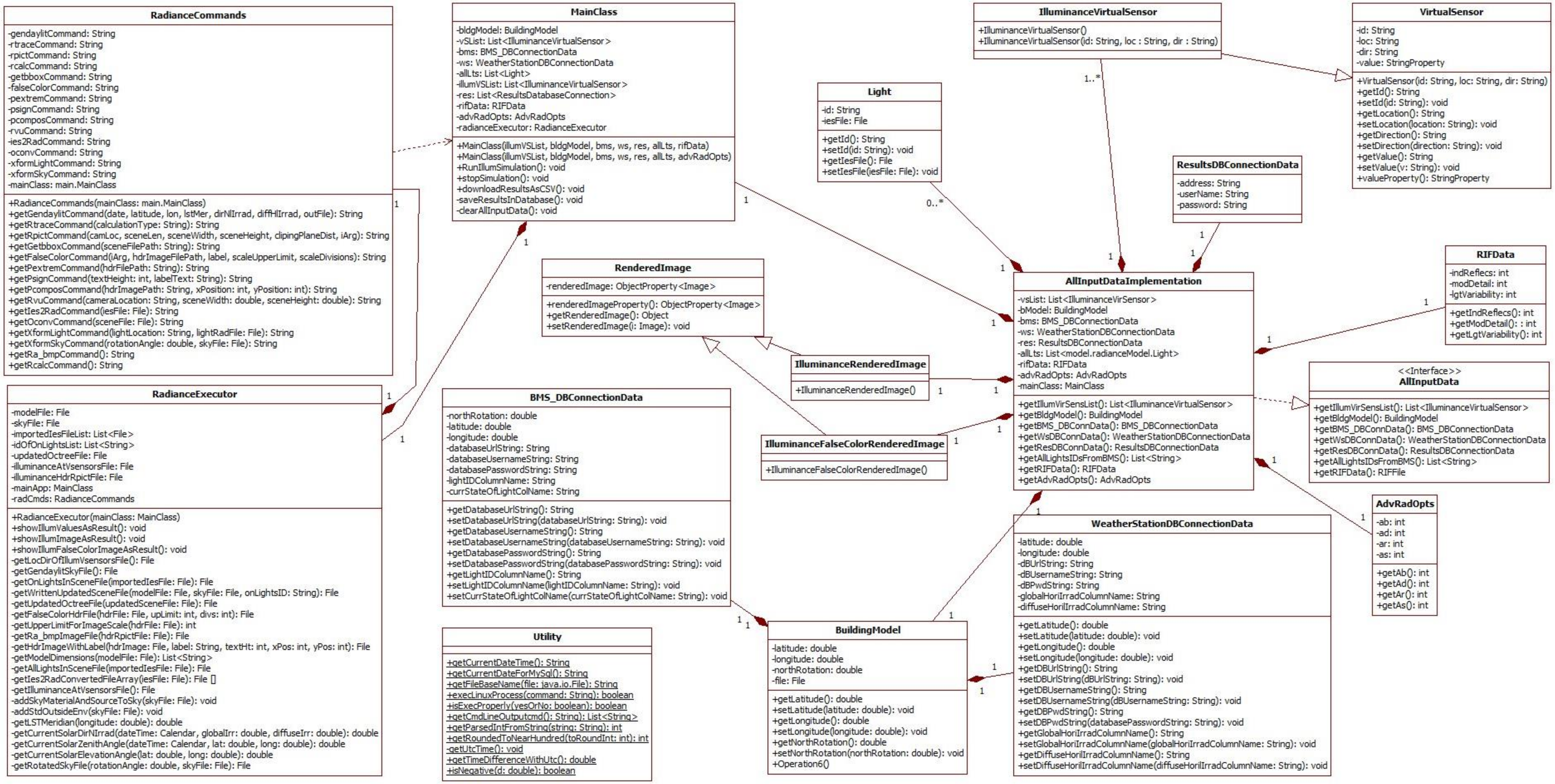

RadianceCommands
-gendaylitCommand: String
-rtraceCommand: String
-rpictCommand: String
-rcalcCommand: String
-getbboxCommand: String
-falseColorCommand: String
-pextremCommand: String
-psignCommand: String
-pcomposCommand: String
-rvuCommand: String
-ies2RadCommand: String
-oconvCommand: String
-xformLightCommand: String
-xformSkyCommand: String
-mainClass: main.MainClass
+RadianceCommands(mainClass: main.MainClass)
+getGendaylitCommand(date, latitude, lon, lstMer, dirNIrrad, diffHIrrad, outFile): String
+getRtraceCommand(calculationType: String): String
+getRpictCommand(camLoc, sceneLen, sceneWidth, sceneHeight, clipingPlaneDist, iArg): String
+getGetbboxCommand(sceneFilePath: String): String
+getFalseColorCommand(iArg, hdrImageFilePath, label, scaleUpperLimit, scaleDivisions): String
+getPextremCommand(hdrFilePath: String): String
+getPsignCommand(textHeight: int, labelText: String): String
+getPcomposCommand(hdrImagePath: String, xPosition: int, yPosition: int): String
+getRvuCommand(cameraLocation: String, sceneWidth: double, sceneHeight: double): String
+getIes2RadCommand(iesFile: File): String
+getOconvCommand(sceneFile: File): String
+getXformLightCommand(lightLocation: String, lightRadFile: File): String
+getXformSkyCommand(rotationAngle: double, skyFile: File): String
+getRa_bmpCommand(): String
+getRcalcCommand(): String
MainClass
-bldgModel: BuildingModel
-vSList: List<IlluminanceVirtualSensor>
-bms: BMS_DBConnectionData
-ws: WeatherStationDBConnectionData
-allLts: List<Light>
-illumVSList: List<IlluminanceVirtualSensor>
-res: List<ResultsDatabaseConnection>
-rifData: RIFData
-advRadOpts: AdvRadOpts
-radianceExecutor: RadianceExecutor
+MainClass(illumVSList, bldgModel, bms, ws, res, allLts, rifData)
+MainClass(illumVSList, bldgModel, bms, ws, res, allLts, advRadOpts)
+RunIllumSimulation(): void
+stopSimulation(): void
+downloadResultsAsCSV(): void
-saveResultsInDatabase(): void
-clearAllInputData(): void
IlluminanceVirtualSensor
+IlluminanceVirtualSensor()
+IlluminanceVirtualSensor(id: String, loc : String, dir : String)
1..*
VirtualSensor
-id: String
-loc: String
-dir: String
-value: StringProperty
+VirtualSensor(id: String, loc: String, dir: String)
+getId(): String
+setId(id: String): void
+getLocation(): String
+setLocation(location: String): void
+getDirection(): String
+setDirection(direction: String): void
+getValue(): String
+setValue(v: String): void
+valueProperty(): StringProperty
Light
-id: String
-iesFile: File
+getId(): String
+setId(id: String): void
+getIesFile(): File
+setIesFile(iesFile: File): void
0..*
ResultsDBConnectionData
-address: String
-userName: String
-password: String
1
1
1
1
1
1
1
RenderedImage
-renderedImage: ObjectProperty<Image>
+renderedImageProperty(): ObjectProperty<Image>
+getRenderedImage(): Object
+setRenderedImage(i: Image): void
AllInputDataImplementation
-vsList: List<IlluminanceVirSensor>
-bModel: BuildingModel
-bms: BMS_DBConnectionData
-ws: WeatherStationDBConnectionData
-res: ResultsDBConnectionData
-allLts: List<model.radianceModel.Light>
-rifData: RIFData
-advRadOpts: AdvRadOpts
-mainClass: MainClass
+getIllumVirSensList(): List<IlluminanceVirtualSensor>
+getBldgModel(): BuildingModel
+getBMS_DBConnData(): BMS_DBConnectionData
+getWsDBConnData(): WeatherStationDBConnectionData
+getResDBConnData(): ResultsDBConnectionData
+getAllLightsIDsFromBMS(): List<String>
+getRIFData(): RIFData
+getAdvRadOpts(): AdvRadOpts
RIFData
-indReflecs: int
-modDetail: int
-lgtVariability: int
+getIndReflecs(): int
+getModDetail(): : int
+getLgtVariability(): int
1
1
<<Interface>>
AllInputData
+getIllumVirSensList(): List<IlluminanceVirtualSensor>
+getBldgModel(): BuildingModel
+getBMS_DBConnData(): BMS_DBConnectionData
+getWsDBConnData(): WeatherStationDBConnectionData
+getResDBConnData(): ResultsDBConnectionData
+getAllLightsIDsFromBMS(): List<String>
+getRIFData(): RIFFile
IlluminanceRenderedImage
+IlluminanceRenderedImage()
1
1
IlluminanceFalseColorRenderedImage
+IlluminanceFalseColorRenderedImage()
1
1
1
RadianceExecutor
-modelFile: File
-skyFile: File
-importedIesFileList: List<File>
-idOfOnLightsList: List<String>
-updatedOctreeFile: File
-illuminanceAtVsensorsFile: File
-illuminanceHdrRpictFile: File
-mainApp: MainClass
-radCmds: RadianceCommands
+RadianceExecutor(mainClass: MainClass)
+showIllumValuesAsResult(): void
+showIllumImageAsResult(): void
+showIllumFalseColorImageAsResult(): void
-getLocDirOfIllumVsensorsFile(): File
-getGendaylitSkyFile(): File
-getOnLightsInSceneFile(importedIesFile: File): File
-getWrittenUpdatedSceneFile(modelFile: File, skyFile: File, onLightsID: String): File
-getUpdatedOctreeFile(updatedSceneFile: File): File
-getFalseColorHdrFile(hdrFile: File, upLimit: int, divs: int): File
-getUpperLimitForImageScale(hdrFile: File): int
-getRa_bmpImageFile(hdrRpictFile: File): File
-getHdrImageWithLabel(hdrImage: File, label: String, textHt: int, xPos: int, yPos: int): File
-getModelDimensions(modelFile: File): List<String>
-getAllLightsInSceneFile(importedIesFile: File): File
-getIes2RadConvertedFileArray(iesFile: File): File []
-getIlluminanceAtVsensorsFile(): File
-addSkyMaterialAndSourceToSky(skyFile: File): void
-addStdOutsideEnv(skyFile: File): void
-getLSTMeridian(longitude: double): double
-getCurrentSolarDirNIrrad(dateTime: Calendar, globalIrr: double, diffuseIrr: double): double
-getCurrentSolarZenithAngle(dateTime: Calendar, lat: double, long: double): double
-getCurrentSolarElevationAngle(lat: double, long: double): double
-getRotatedSkyFile(rotationAngle: double, skyFile: File): File
1
1
1
1
BMS_DBConnectionData
-northRotation: double
-latitude: double
-longitude: double
-databaseUrlString: String
-databaseUsernameString: String
-databasePasswordString: String
-lightIDColumnName: String
-currStateOfLightColName: String
+getDatabaseUrlString(): String
+setDatabaseUrlString(databaseUrlString: String): void
+getDatabaseUsernameString(): String
+setDatabaseUsernameString(databaseUsernameString: String): void
+getDatabasePasswordString(): String
+setDatabasePasswordString(databasePasswordString: String): void
+getLightIDColumnName(): String
+setLightIDColumnName(lightIDColumnName: String): void
+setCurrStateOfLightColName(currStateOfLightColName: String): void
AdvRadOpts
-ab: int
-ad: int
-ar: int
-as: int
+getAb(): int
+getAd(): int
+getAr(): int
+getAs(): int
1
1
1
Utility
+getCurrentDateTime(): String
+getCurrentDateForMySql(): String
+getFileBaseName(file: java.io.File): String
+execLinuxProcess(command: String): boolean
+isExecProperly(yesOrNo: boolean): boolean
+getCmdLineOutputcmd(): String): List<String>
+getParsedIntFromString(string: String): int
+getRoundedToNearHundred(toRoundInt: int): int
-getUtcTime(): void
+getTimeDifferenceWithUtc(): double
+isNegative(d: double): boolean
1
1
BuildingModel
-latitude: double
-longitude: double
-northRotation: double
-file: File
+getLatitude(): double
+setLatitude(latitude: double): void
+getLongitude(): double
+setLongitude(longitude: double): void
+getNorthRotation(): double
+setNorthRotation(northRotation: double): void
+Operation6()
1
1
1
WeatherStationDBConnectionData
-latitude: double
-longitude: double
-dBUrlString: String
-dBUsernameString: String
-dBPwdString: String
-globalHoriIrradColumnName: String
-diffuseHoriIIrradColumnName: String
+getLatitude(): double
+setLatitude(latitude: double): void
+getLongitude(): double
+setLongitude(longitude: double): void
+getDBUrlString(): String
+setDBUrlString(dBUrlString: String): void
+getDBUsernameString(): String
+setDBUsernameString(dBUsernameString: String): void
+getDBPwdString(): String
+setDBPwdString(databasePasswordString: String): void
+getGlobalHoriIrradColumnName(): String
+setGlobalHoriIrradColumnName(globalHoriIrradColumnName: String): void
+getDiffuseHoriIIrradColumnName(): String
+setDiffuseHoriIIrradColumnName(diffuseHoriIIrradColumnName: String): void

Figure 3: UML Class diagram of the application

3.3 Graphical User Interface (GUI) development

Graphical User Interface (GUI) is a type of user interface that allows interaction between human and machine (or software) using graphical icons and visual indicators. GUI for *RealTimeRadiance* has been developed in JavaFX framework by following the Model View Controller (MVC) software design pattern. MVC design pattern divides the software into three interconnected components (or packages) such as Model, View, and Controller which helps in separating the different logical concerns of the application. As shown in Figure 4, Model represents the classes related to business logic behind the application; View represents the classes related to GUI elements such as Button, TextField, Drop-down menu, etc.; Controller represents the classes related to interface between View, and Model. For example: whenever a button (part of a View) is clicked, the method written in its Controller class is executed which implements or exchanges data between other classes in the Model (Metsker & Wake, 2006).

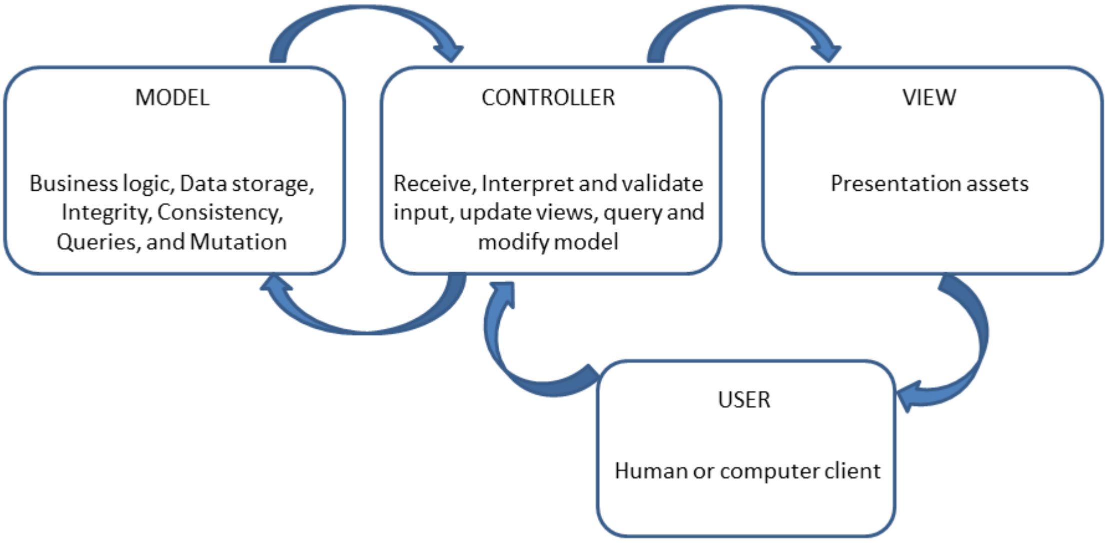

MODEL
Business logic, Data storage, Integrity, Consistency, Queries, and Mutation
CONTROLLER
Receive, Interpret and validate input, update views, query and modify model
VIEW
Presentation assets
USER
Human or computer client

Figure 4: Diagram describes the interaction between the components of Model-View-Controller design pattern

Another important design pattern used in the application is the Observer design pattern. JavaFX API has Property listeners (javafx.beans.property) and Binding (javafx.beans.binding) features which allows storing data properties (read-only values) into observable data structures (javafx.collections) such as ObservableList. ObservableList allows listeners (inbuilt programs) to track changes when they occur (JavaFX 2.2 API Docs), which means that if a data property, stored in an ObservableList, is bonded (using JavaFX Binding API) with, for example, a table cell (GUI element), then whenever the value of this property changes, the change gets immediately reflected in the table cell. This feature allowed binding illuminance value property (and also illuminance image property) with the table cells in the results window, as shown in Figure 10, which resulted in automatic and immediate update of simulated results on the GUI after every simulation run.

In summary the developed GUI provides the following functionality:

Visualize *Radiance* building model

Define location and direction of virtual sensors, and save user input in serialized XML

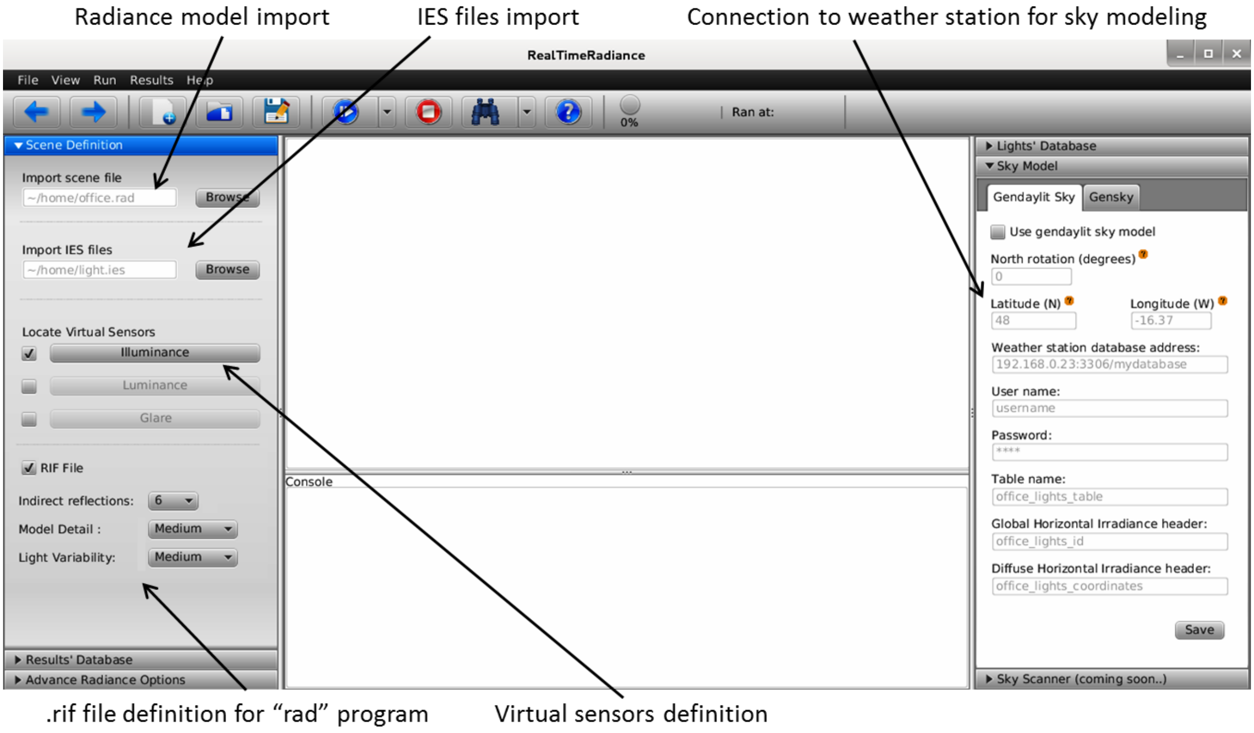
Define qualitative input for automation in ambient parameters selection (RIF file)

Define individual ambient parameter explicitly, for experienced users

Continuously exchange data with databases of BMS and weather station

Continuously store simulation output in the database for future use

Visualize current simulation results and background processes (commands)



Radiance model import

IES files import

Connection to weather station for sky modeling

.rif file definition for “rad” program

Virtual sensors definition

Import scene file: ~/home/office.rad

Import IES files: ~/home/light.ies

☑ Illuminance

□ Luminance

□ Glare

☑ RIF File

Indirect reflections: 6

Model Detail : Medium

Light Variability: Medium

□ Use gendaylit sky model

North rotation (degrees): 0

Latitude (N): 48

Longitude (W): -16.37

Weather station database address: 192.168.0.23:3306/mydatabase

User name: username

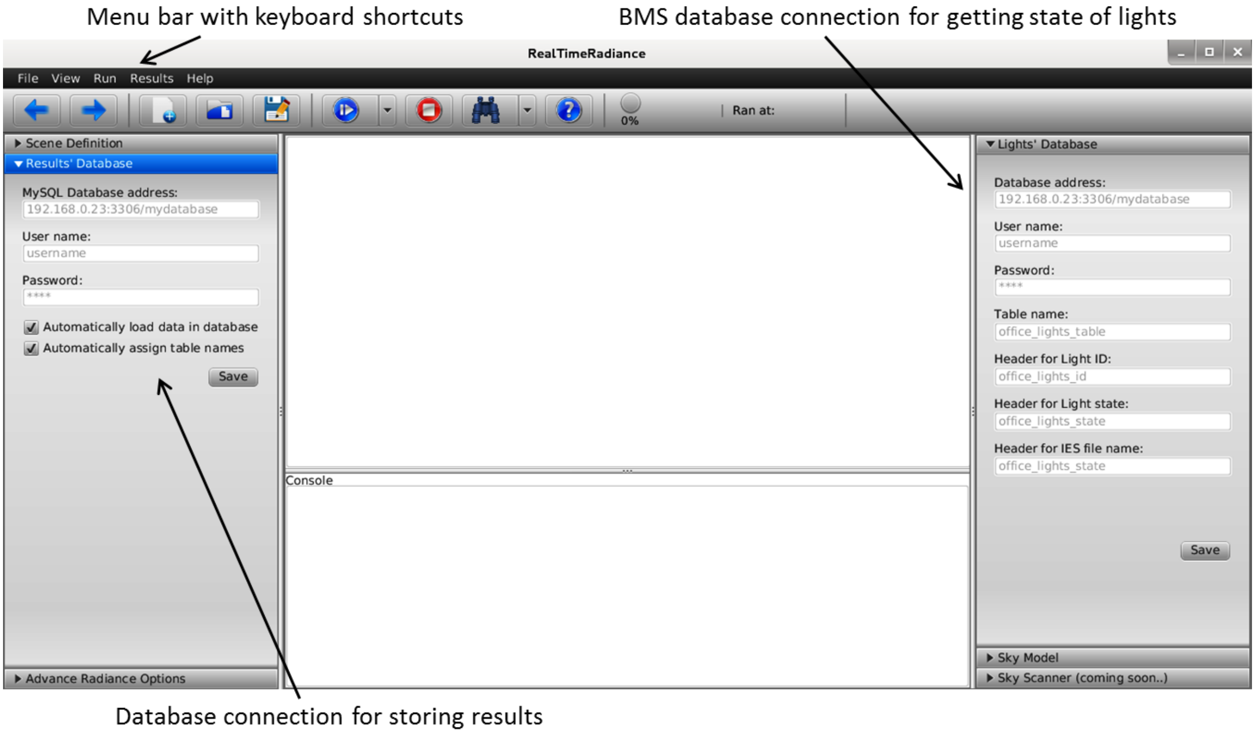
Password: ****

Table name: office_lights_table

Global Horizontal Irradiance header: office_lights_id

Diffuse Horizontal Irradiance header: office_lights_coordinates

Figure 5: GUI of the developed application showing tabs for the *Radiance* model (left) and sky model definition (right)



Menu bar with keyboard shortcuts

BMS database connection for getting state of lights

Database connection for storing results

MySQL Database address: 192.168.0.23:3306/mydatabase

User name: username

Password: ****

☑ Automatically load data in database

☑ Automatically assign table names

Database address: 192.168.0.23:3306/mydatabase

User name: username

Password: ****

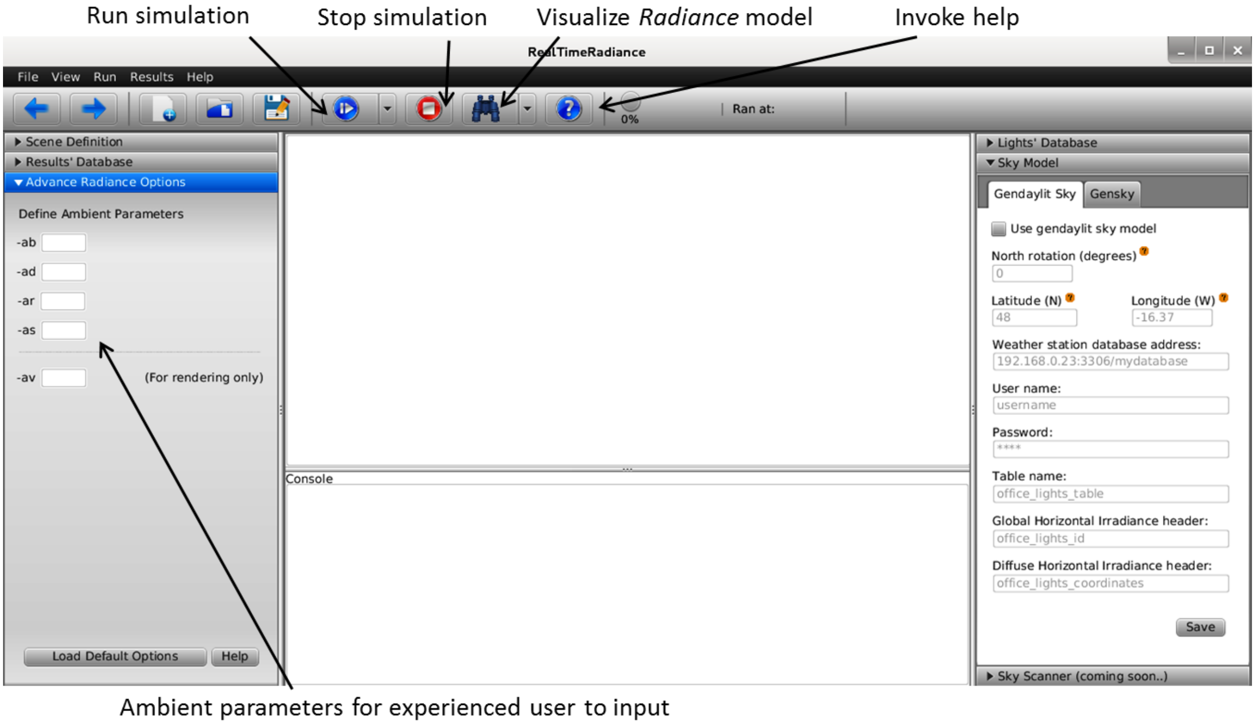
Table name: office_lights_table

Header for Light ID: office_lights_id

Header for Light state: office_lights_state

Header for IES file name: office_lights_state

Figure 6: GUI of the developed application showing tabs for the results (left) and BMS database connection (right)



Run simulation

Stop simulation

Visualize *Radiance* model

Invoke help

Ambient parameters for experienced user to input

Define Ambient Parameters

-ab

-ad

-ar

-as

-av (For rendering only)

□ Use gendaylit sky model

North rotation (degrees): 0

Latitude (N): 48

Longitude (W): -16.37

Weather station database address: 192.168.0.23:3306/mydatabase

User name: username

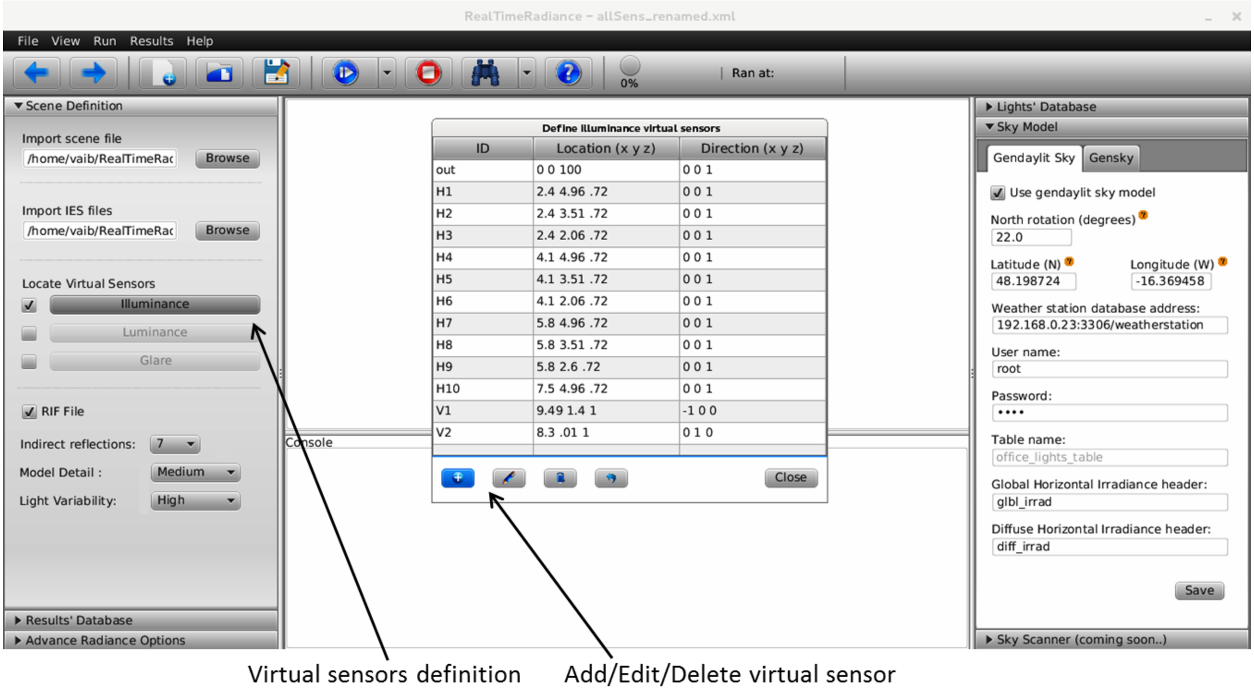
Password: ****

Table name: office_lights_table

Global Horizontal Irradiance header: office_lights_id

Diffuse Horizontal Irradiance header: office_lights_coordinates

Figure 7: GUI of the developed application showing tabs for ambient parameter definition (left) and auxiliary features (middle)



Import scene file: /home/vaib/RealTimeRad

Import IES files: /home/vaib/RealTimeRad

Locate Virtual Sensors

☑ Illuminance

□ Luminance

□ Glare

☑ RIF File

Indirect reflections: 7

Model Detail : Medium

Light Variability: High

Define illuminance virtual sensors

ID	Location (x y z)	Direction (x y z)
out	0 0 100	0 0 1
H1	2.4 4.96 .72	0 0 1
H2	2.4 3.51 .72	0 0 1
H3	2.4 2.06 .72	0 0 1
H4	4.1 4.96 .72	0 0 1
H5	4.1 3.51 .72	0 0 1
H6	4.1 2.06 .72	0 0 1
H7	5.8 4.96 .72	0 0 1
H8	5.8 3.51 .72	0 0 1
H9	5.8 2.6 .72	0 0 1
H10	7.5 4.96 .72	0 0 1
V1	9.49 1.4 1	-1 0 0
V2	8.3 .01 1	0 1 0

☑ Use gendaylit sky model

North rotation (degrees): 22.0

Latitude (N): 48.198724

Longitude (W): -16.369458

Weather station database address: 192.168.0.23:3306/weatherstation

User name: root

Password: ••••

Table name: office_lights_table

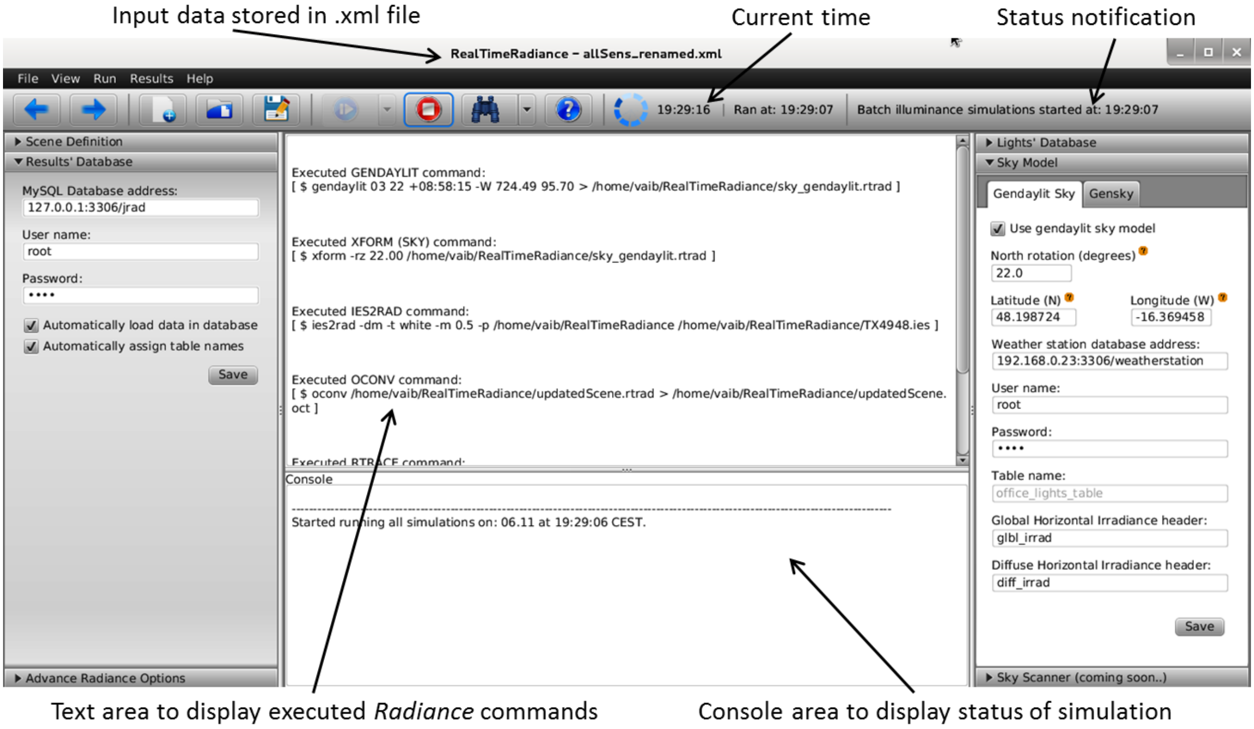
Global Horizontal Irradiance header: glbl_irrad

Diffuse Horizontal Irradiance header: diff_irrad

Virtual sensors definition

Add/Edit/Delete virtual sensor

Figure 8: GUI of the developed application showing the panel for virtual sensor data input



Input data stored in .xml file

Current time

Status notification

RealTimeRadiance – allSens_renamed.xml

19:29:16 | Ran at: 19:29:07 | Batch illuminance simulations started at: 19:29:07

MySQL Database address: 127.0.0.1:3306/jrad

User name: root

Password: ••••

☑ Automatically load data in database

☑ Automatically assign table names

```
Executed GENDAYLIT command:
[ $ gendaylit 03 22 +08:58:15 -W 724.49 95.70 > /home/vaib/RealTimeRadiance/sky_gendaylit.rtrad ]


Executed XFORM (SKY) command:
[ $ xform -rz 22.00 /home/vaib/RealTimeRadiance/sky_gendaylit.rtrad ]


Executed IES2RAD command:
[ $ ies2rad -dm -t white -m 0.5 -p /home/vaib/RealTimeRadiance /home/vaib/RealTimeRadiance/TX4948.ies ]


Executed OCONV command:
[ $ oconv /home/vaib/RealTimeRadiance/updatedScene.rtrad > /home/vaib/RealTimeRadiance/updatedScene.
oct ]


Executed RTRACE command:
```

```
--------------------------------------------------------------------------------------------
Started running all simulations on: 06.11 at 19:29:06 CEST.
```

☑ Use gendaylit sky model

North rotation (degrees): 22.0

Latitude (N): 48.198724

Longitude (W): -16.369458

Weather station database address: 192.168.0.23:3306/weatherstation

User name: root

Password: ••••

Table name: office_lights_table

Global Horizontal Irradiance header: glbl_irrad

Diffuse Horizontal Irradiance header: diff_irrad

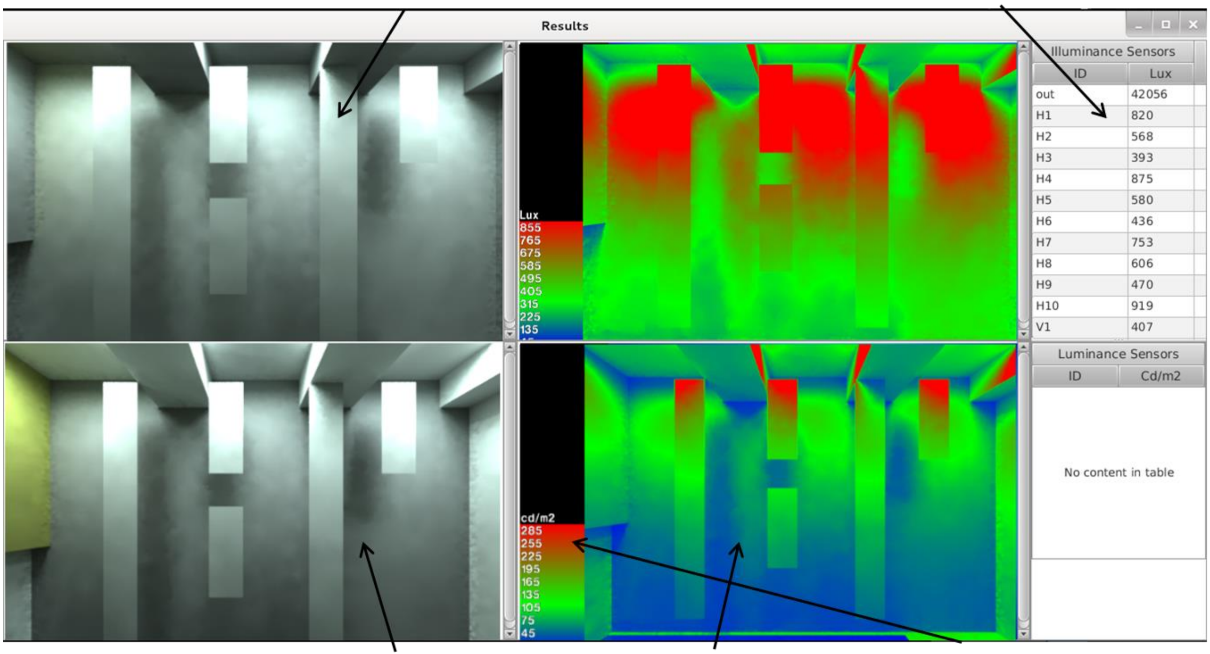
Text area to display executed *Radiance* commands

Console area to display status of simulation

Figure 9: GUI of the developed application showing input data storage, diagnostic, and notification features

Illuminance image for monitoring/fault detection

Simulated illuminance values from virtual sensors



Illuminance Sensors

ID	Lux
out	42056
H1	820
H2	568
H3	393
H4	875
H5	580
H6	436
H7	753
H8	606
H9	470
H10	919
V1	407

Luminance Sensors

ID	Cd/m2

No content in table

Luminance image for monitoring/fault detection

False color renderings, with dynamic scales, for analysis

Figure 10: GUI of the developed application showing panel for resulting illuminance values (right), and auxiliary rendering for model diagnostics

4 VALIDATION

4.1 Experimental setup

The experiment was setup in the seminar hall (about 60 sq. m) in the Institute of Building Physics and Building Ecology at Vienna University of Technology, Vienna, Austria. Figure 11, to Figure 13 shows the location of the seminar hall and its immediate surroundings. As shown in these images, the weather station is taller than the surrounding buildings, in other words, it gets an un-obstructed view of the sky. Table 1 and Figure 17 describe the sensors from the weather station which were used in sky modeling. Section 4.2.3 describes sky modeling in more detail.

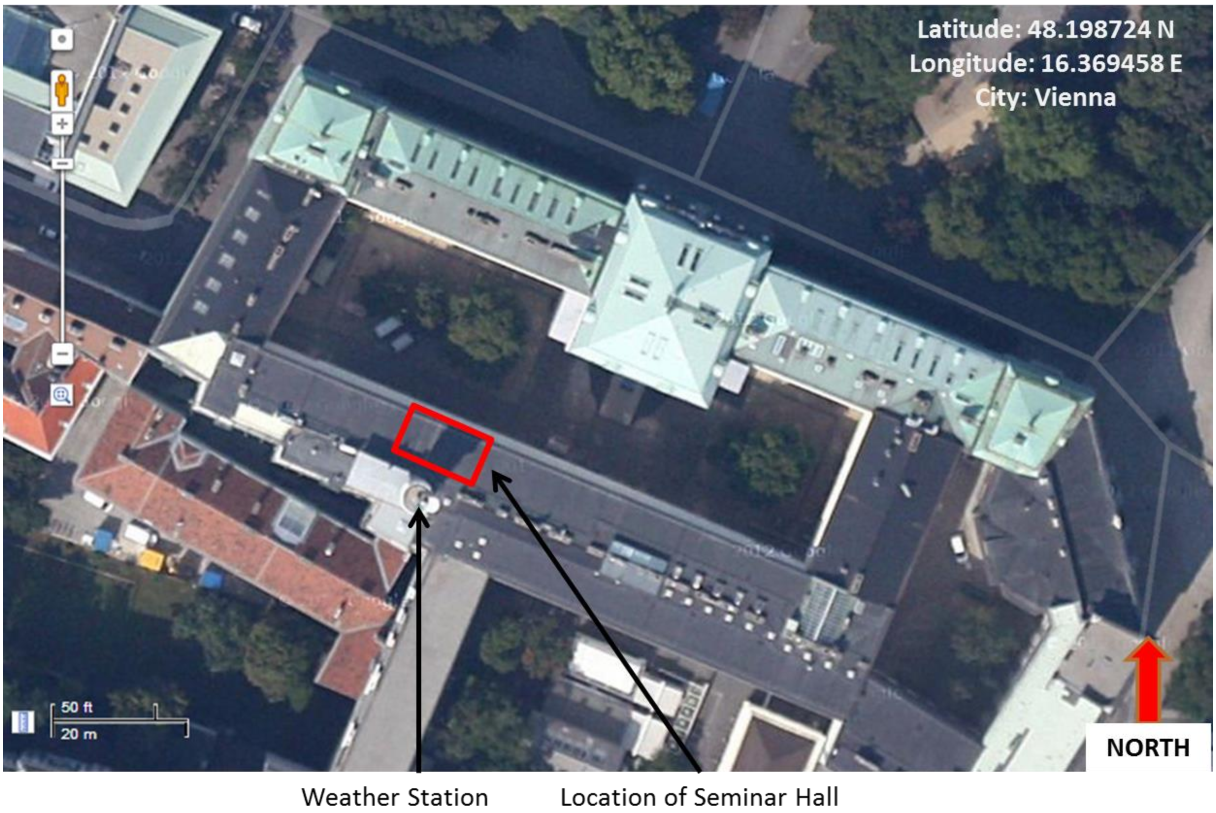


Figure 11: Aerial view showing location of seminar hall and its surroundings from top

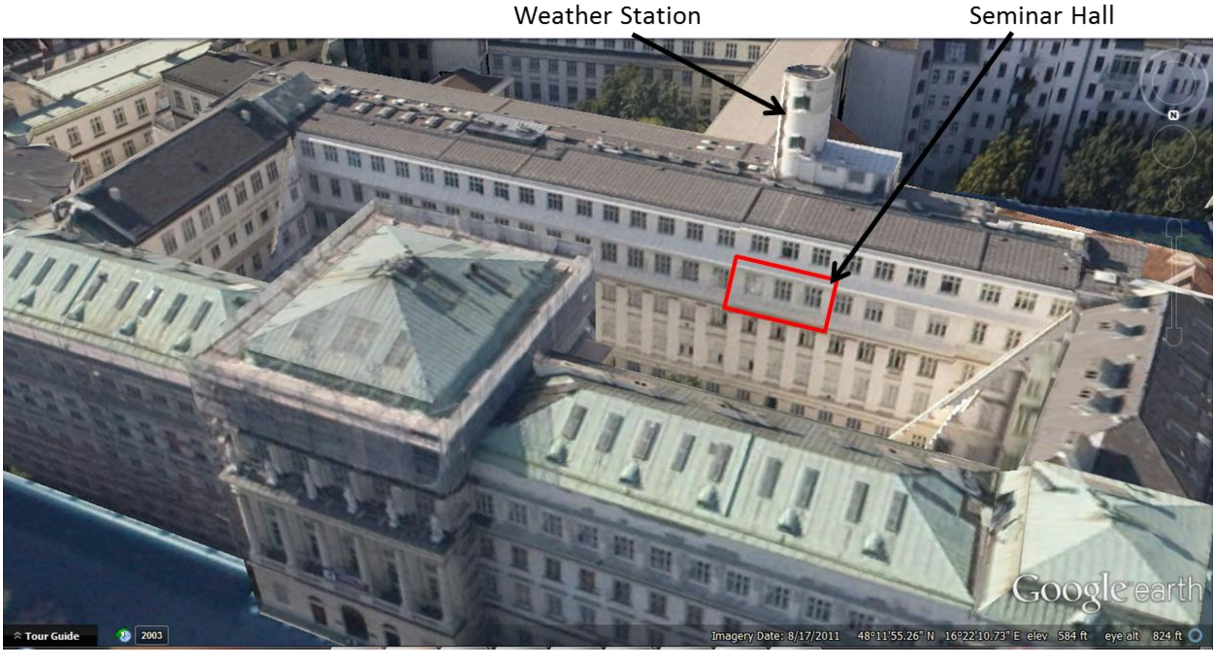

Weather Station
Seminar Hall
N
Google earth
Tour Guide
2003
Imagery Date: 8/17/2011 48°11'55.26" N 16°22'10.73" E elev 584 ft eye alt 824 ft

Figure 12: Aerial view showing location of seminar hall and its surroundings from North

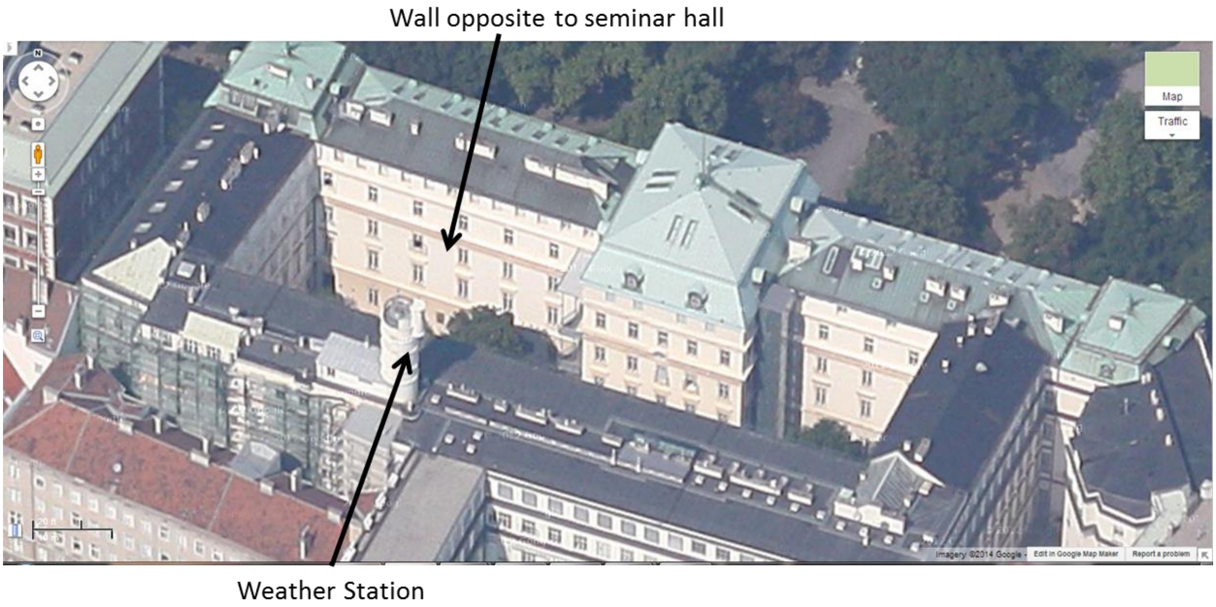


Figure 13: Aerial view showing location of weather station and its surrounding from South

The experiment was conducted on a weekend, 22nd and 23rd March 2014, when the institute was un-occupied. It was important to prevent human movement or intervention during the experiment, as the illuminance measurement is very sensitive towards shadows. Indoor illuminance was measured at twelve different locations. Ten out of twelve sensors were placed on tables, at work plane height of 0.72 m, to measure horizontal illuminance. The remaining two sensors were placed on the two different adjacent walls, at the height of 1m above ground, to measure vertical illuminance. Figure 14 and Figure 15 show the interior of the seminar hall. These images qualitatively describe different elements of the room, their surface properties. For example wall and ceiling are painted white, floor is wooden and light brown in color, and partition door is made up of reflective translucent glass. It is important to note here that, although the windows are facing North-East direction, there is still reflected light entering the hall and falling on the sensors, and subsequently shadow of the window frame (mullion) is seen falling on a sensor. Another point worth noting is that, all the chairs and white boards were removed from the seminar hall during the experiment in order to minimize major sources of confounds.

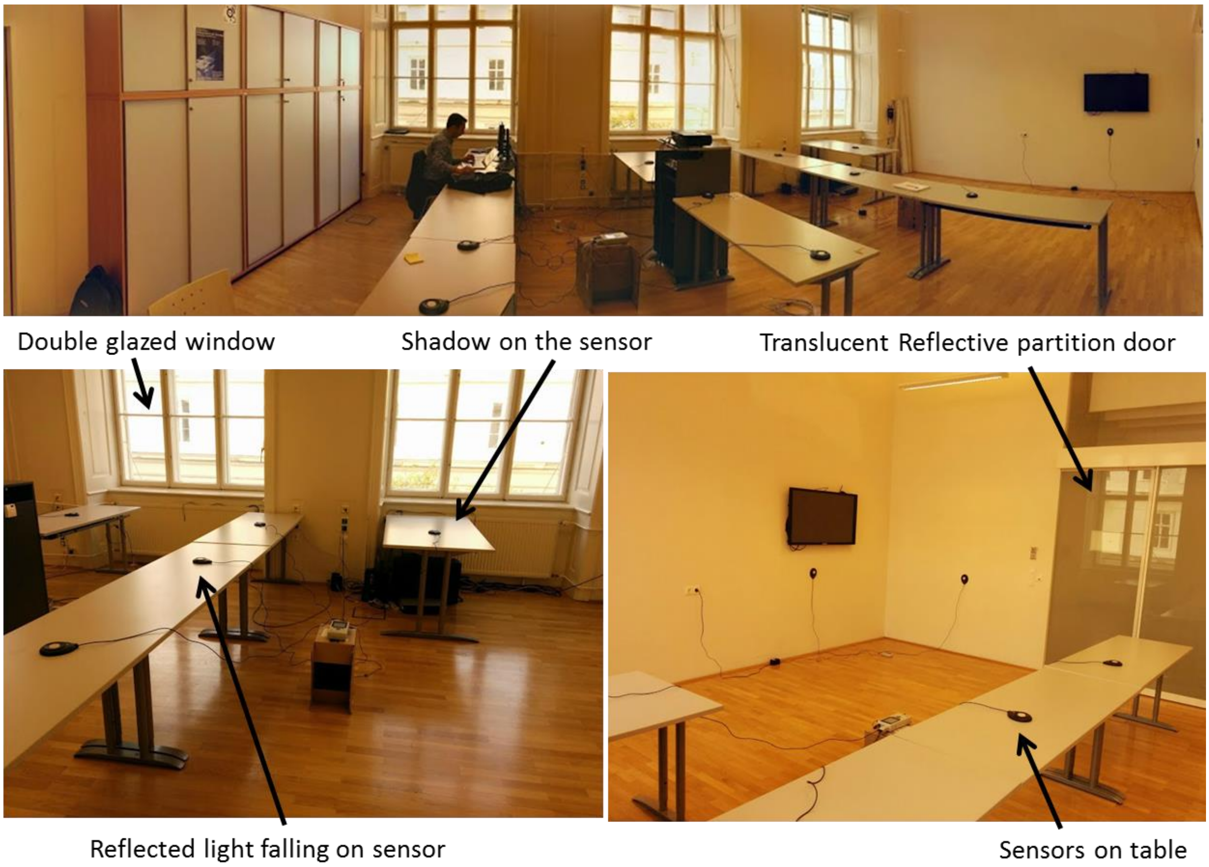

Double glazed window
Shadow on the sensor
Translucent Reflective partition door
Reflected light falling on sensor
Sensors on table

Figure 14: Photograph showing interior of the seminar hall

Figure 15 is the plan of the seminar hall which shows the location of indoor illuminance sensors. Figure 16 is a representative section showing the distance of the three major rows of sensors from the window, which relates to the area of visible sky. Figure 18 shows the indoor sensor used in the experiment. Figure 21 shows the view from these virtual sensors, which can be helpful in analyzing their performance. Table 4 shows the area (%) of sky visible from these sensors.

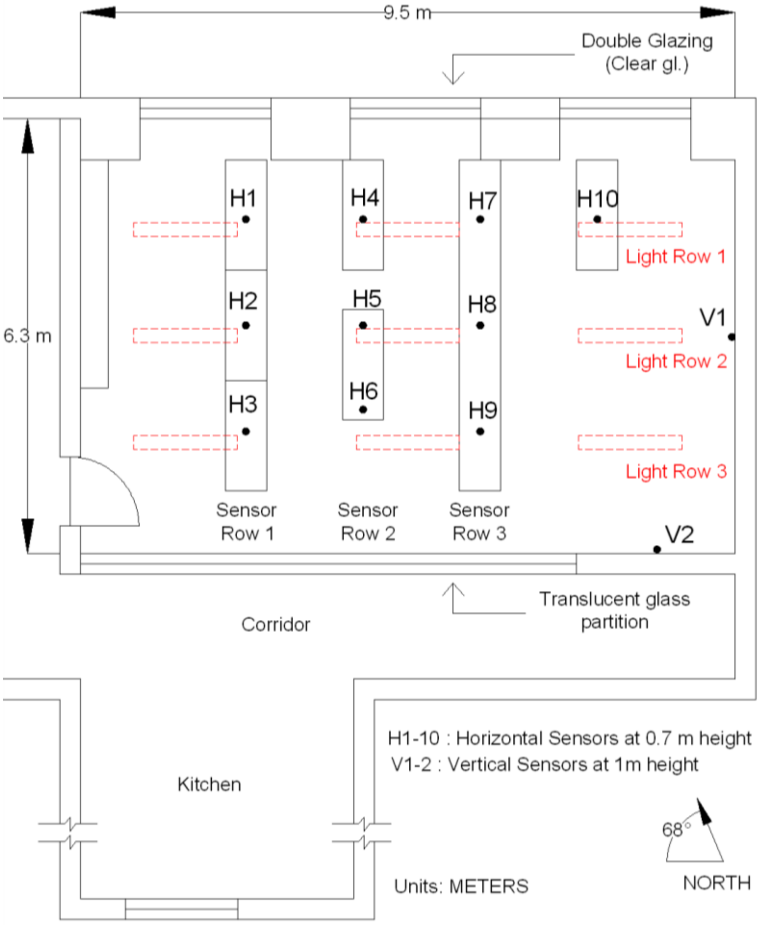


Figure 15: Plan of seminar hall showing location of sensors (also sensor rows) and lights (also light rows)

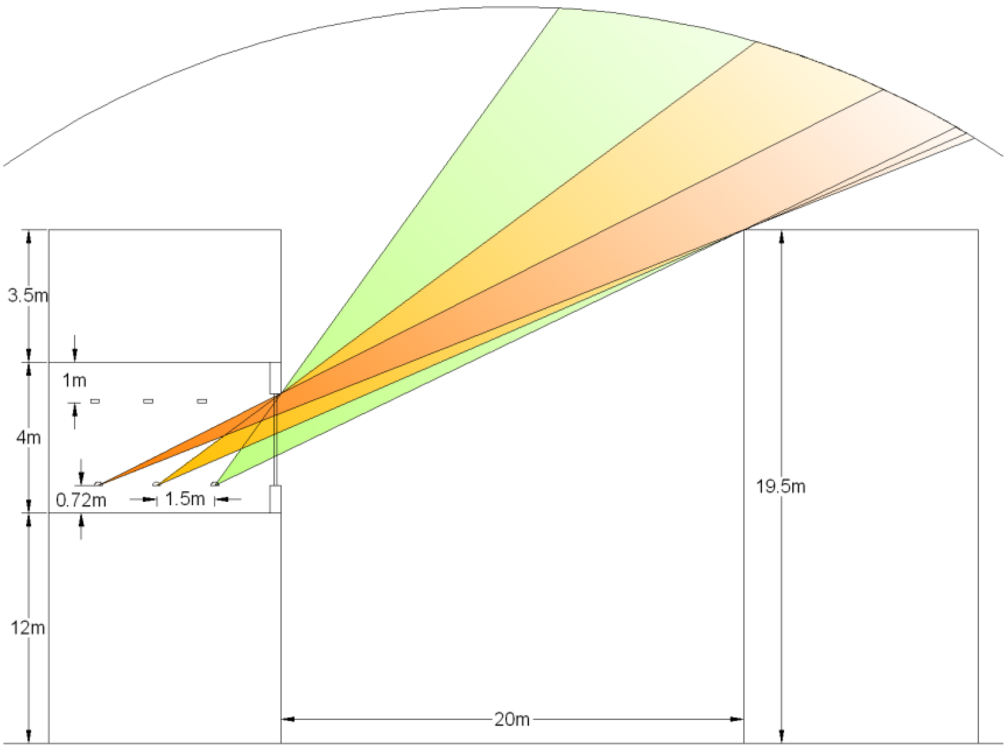


Figure 16: Section of seminar hall adjacent building and sky view from different sensor locations

Table 1: Table describing the different kinds of sensors used in the experiment

Sensor type	Measured quantity	Units	Model no.	Manufacturer
Sunshine pyranometer	Diffuse horizontal irradiance	W/m2	SPN1	Delta-T
Pyranometer	Global horizontal irradiance	W/m2	CM3	Kipp and Zonen
Illuminance sensor	Indoor illuminance	Lux	CA 808	Chauvin Arnoux
Illuminance sensor	Outdoor illuminance	Lux		

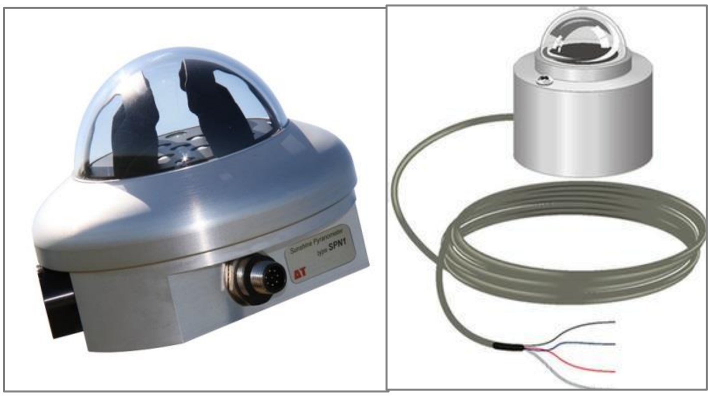


Figure 17: Sunshine pyranometer (left) for diffuse irradiance and pyranometer (right) for global irradiance measurement

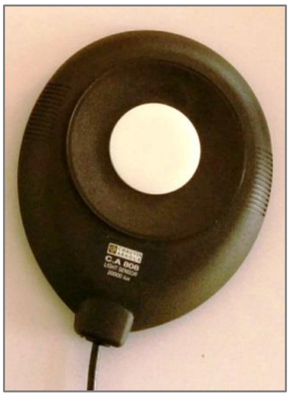


Figure 18: Indoor illuminance meter used in the experiment to measure work plane illuminance

4.2 Simulation model

4.2.1 Building model

The simulation model of the seminar hall and surrounding buildings was geometrically made very close to the reality. The dimensions of the hall, tables, and window frame were measured by hand to an accuracy of ~1 cm, ~0.2 cm, and ~0.2 cm respectively. Tables were modelled as discrete horizontal planes at 0.72m above ground to represent the work plane. Table legs were not modelled for the reason of simplification, and reducing the computational overhead (Ward & Shakespeare, 1998); also for the same reason, horizontal and vertical elements of the window frame were modelled as planes with no thickness. In reality, the exterior windows are double glazed with clear glass of light transmittance of 0.8 each, however windows have been modelled as single glazed with clear glass but light transmittance is reduced to 0.72 as recommended by (Reinhart, 2011). During calibration of the building model it was observed that modeling two panes of window glazing gives unreliable results. This observation was also supported by best practices described in (Reinhart, 2011). The illuminance type virtual sensors were not modelled; rather horizontal/vertical illuminance at that point was calculated. Exterior buildings (obstructions) and ground plane were also modelled. Figure 19 shows interior and exterior view of the building model.

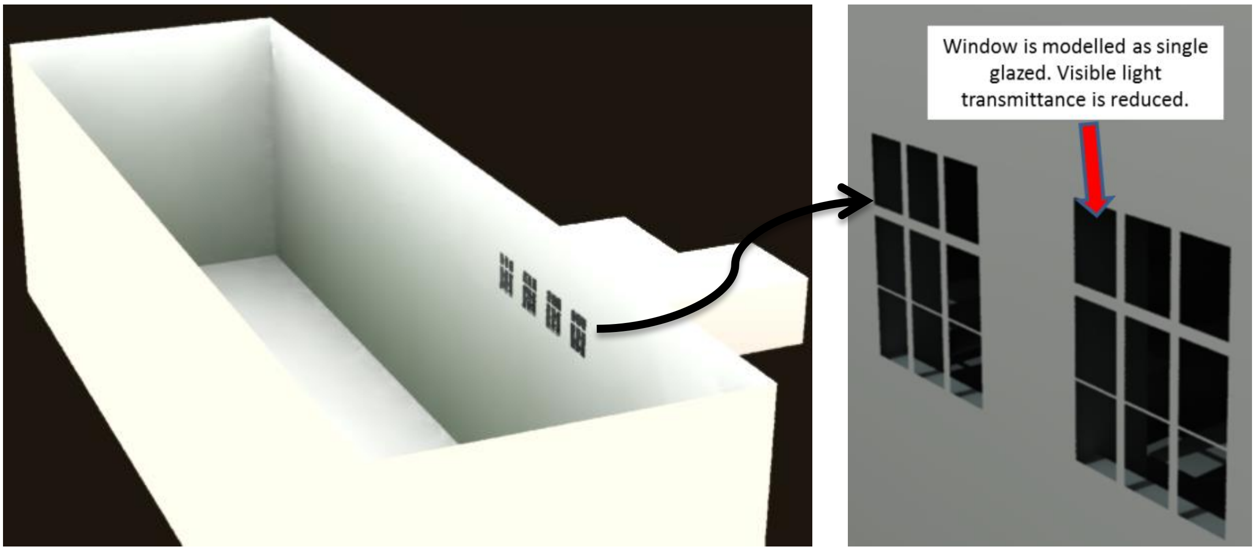

Window is modelled as single glazed. Visible light transmittance is reduced.

Figure 19: Rendering showing the exterior model view (left) and close up of modeled windows from outside (right)

In order to calculate the reflectance of the indoor opaque surfaces, luminance (Cd/m2) and illuminance (lux) at a particular spot on each of the surfaces was measured. Luminance was measured using Minolta Luminance Meter LS-100, and illuminance was measured using Minolta T-10A illuminance meter as described in Table 2 and Figure 20. An average of three measurements for both luminance and illuminance, taken at an interval of 10 minutes each, was considered for the calculation of reflectance. Reflectance (unit less) of the surfaces was calculated using the following equation (Hiscocks, 2011):

$$\rho = \frac{L\,\pi}{E} \tag{1}$$

Where,

ρ is the reflectance of the surface

L is the average luminance at a particular spot on the surface

E is the average illuminance at the same spot on the surface

Table 3 shows the calculated reflectance of all the surfaces in the seminar hall.

Table 2 Table describing the sensors use in surface reflectance measurement

Sensor type	Measured quantity	Units	Model no.	Manufacturer
Luminance meter	Luminance	Cd/m2	LS-100	Minolta
Illuminance meter	Illuminance	Lux	T-10A	Minolta

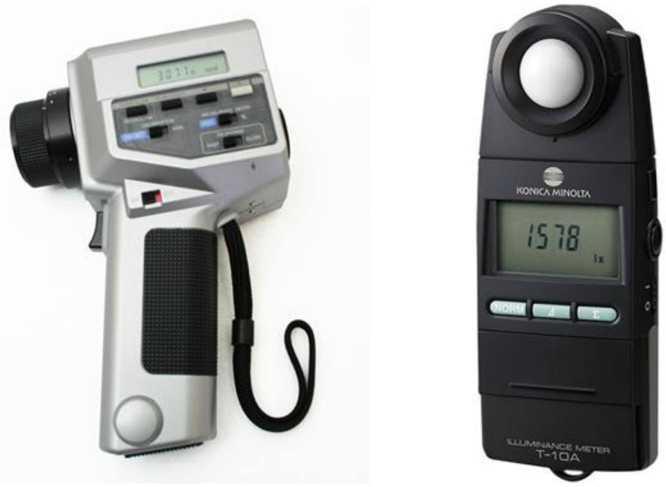


Figure 20: Luminance meter (left) and hand-held illuminance meter (right) used for reflectance measurement

Table 3: Table showing calculated reflectance of different surfaces and their definition in *Radiance*

Surface material properties (measured)											
	Table Top	**Interior Wall / Ceiling / Window sill**	**Double glazed window**	**Partition Glass Door**	**Floor**	**Cabinet Front**	**Cabinet Side**	**Door**	**Acoustic Tiles**	**Exterior wall (white wall+glass) avg.**	**External Ground**
Real Material	Sunmica (Satin finish)	Paint (Matt)	Clear glass	Glass (Translucent)	Wood (Polished)	Sunmica (Satin finish)	Sunmica (Satin finish)	Sunmica (Satin finish)	Polystyrene	-	-
Real Color	Light Grey	White	transparent	Grey	Light Brown	Light Grey	Reddish Brown	White	White	-	-
Avg. Reflectivity (Measured)	0.471	0.787	-	-	0.376	0.468	0.433	0.820	0.702	0.5	0.2
Visible light transmittance	-	-	0.720	0.717	-	-	-	-	-	-	-
Radiance input											
Radiance Material	plastic	plastic	glass	trans	plastic	plastic	plastic	plastic	plastic	plastic	plastic
Radiance definition	void plastic TableTop_BPI 0 0 5 0.471 0.471 0.471 0 0	void plastic Wall_BPI 0 0 5 0.788 0.788 0.788 0 0	void glass DblGlz_BPI 0 0 3 0.72 0.72 0.72	void trans TranslucentGlass_BPI 0 0 7 0.7 0.7 0.7 .07 0 0.872 0.1	void plastic Floor_BPI 0 0 5 0.376 0.376 0.376 0 0	void plastic CabinetFront 0 0 5 0.468 0.468 0.468 0 0	void plastic CabinetSide 0 0 5 0.433 0.433 0.433 0 0	void plastic Door_BPI 0 0 5 0.820 0.820 0.820 0 0	void plastic Plaster_Insulation_Suspended 0 0 5 0.82 0.82 0.82 0 0	void plastic ExternalWall_BPI 0 0 5 0.5 0.5 0.5 0 0	void plastic ExternalPaving 0 0 5 0.2 0.2 0.2 0 0

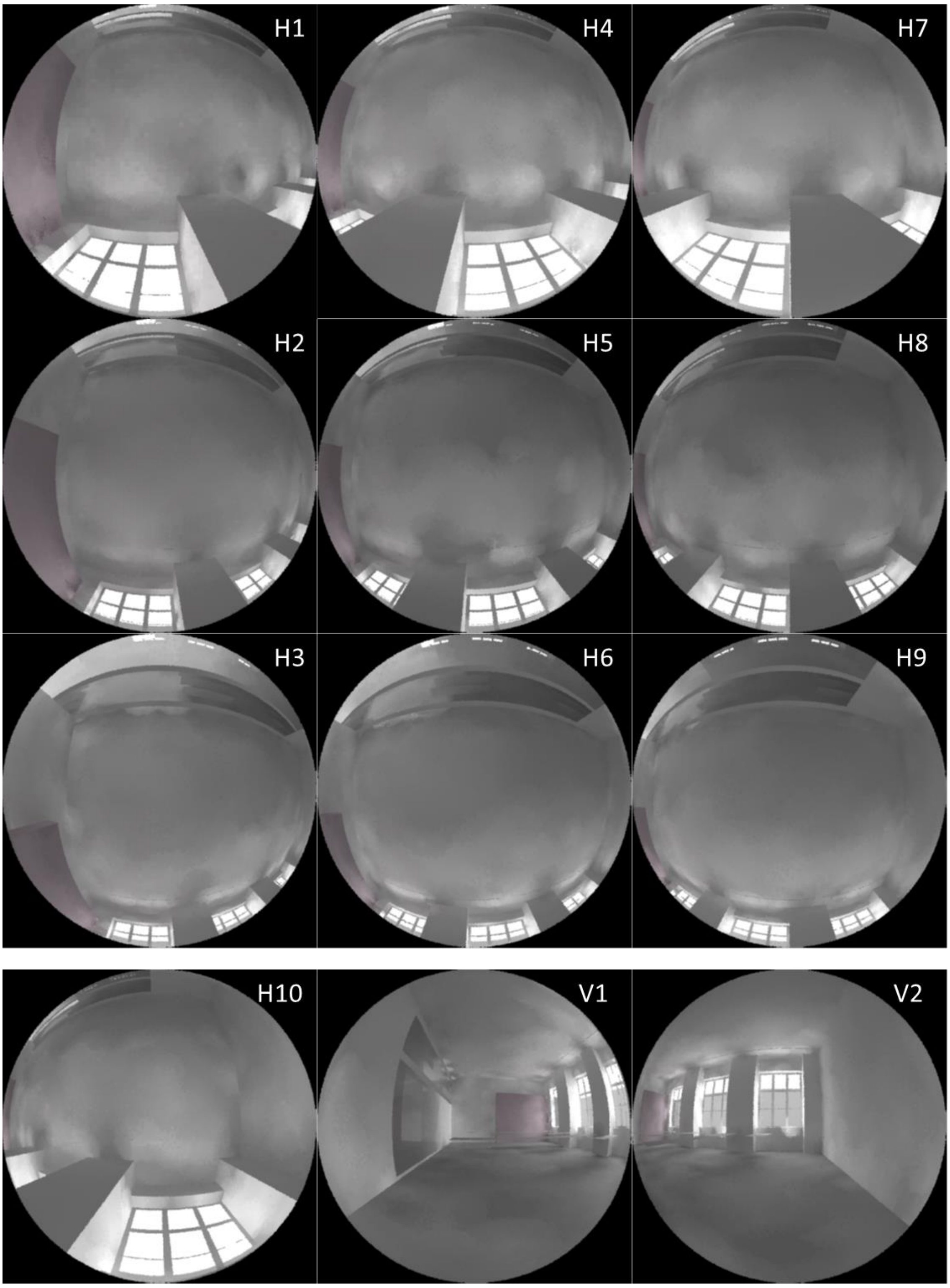


Figure 21: Figure showing the point of view of different virtual sensors

Table 4: Table showing the visible sky area (%) from different virtual sensors

Visible sky area from sensor (% of total visible area)											
Row 1			Row 2			Row 3					
H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	V1	V2
6.81	2.75	1.21	6.49	2.93	1.34	6.16	2.88	1.36	7.20	0.48	0.97

4.2.2 Artificial lighting

The artificial lights that are present in the seminar hall are suspended fluorescent light fixture (refer APPENDIX for specification) with direct-indirect light distribution. Each light fixture houses two T5 luminaires (each 53W), with total lumen output of 4300 lumens. Figure 15 shows the position of the light rows (three fixtures in one row) inside the room, and Figure 22 shows the light distribution of the light fixture that has been obtained from the photometric data file of the fixture (IES file). Left most image in this figure represents the photometric data directly from the IES file of the fixture; the middle image shows the light cut-off and distribution for both transverse and lateral plane; the third and the right most image shows the light distribution when the fixture is placed at the centroid of a cube that represents a room. The latter two images have been generated from the converted photometric file (.rad format) from the *ies2rad* conversion program of the *Radiance* suite using ltview.pl and ltview.pl Perl scripts respectively which were obtained from (Jacobs, 2014). These scripts help in verifying the converted photometric file (output of ies2rad program) that will be used in the *Radiance* model. A light loss factor of 0.5 was calculated using the procedure defined in (Zhu, 2010). Table 5 shows the lighting schedule used during the experiment for artificial light case. Typical command used to convert photometric file (.ies) to *Radiance* format was: **$ ies2rad -dm -t white -m 0.5 TX4948.ies**

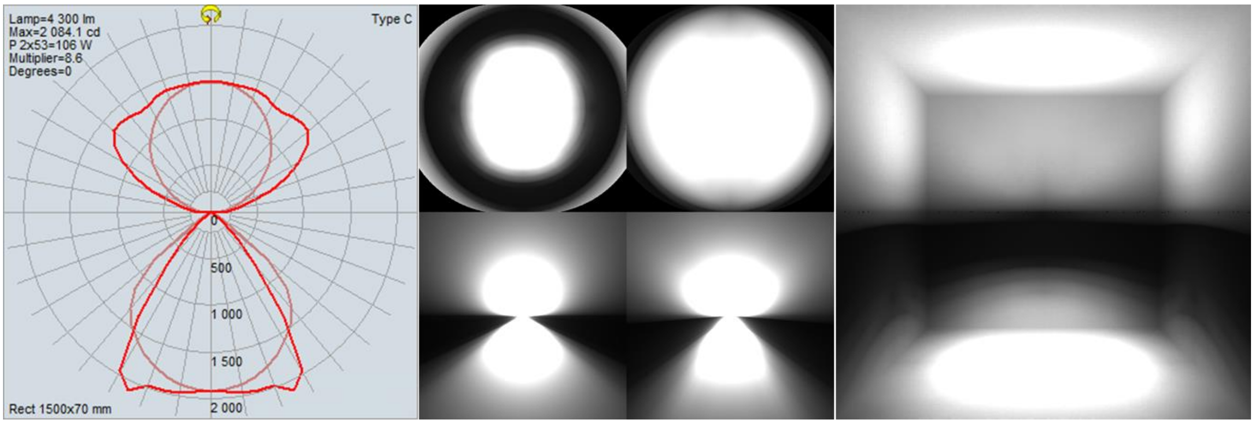


Figure 22: Figure showing photometric diagram (left), light distribution in both planes (middle), and inside a box (right)

Table 5: Table showing the scenario considered to switch state (On/Off) of artificial lights during the experiment

Light Row	Time	Status
All rows	18:30 to 19:00	On
Only Row 1	19:00 to 19:15	On
Only Row 2	19:15 to 19:30	On
Only Row 3	19:30 to 19:45	On
All rows	19:45 to 20:00	Off

4.2.3 Sun and sky model

4.2.3.1 Monitoring of sky luminance distribution

In Radiance, sky model can be generated using gensky or gendaylit programs. *gendaylit* produces a sky brightness distribution based on the Perez All-Weather model (Perez, Seals, & Michalsky, 1993). Perez sky model is widely used in research and development as it accommodates the full range of naturally occurring sky conditions within a single theoretical scheme. The appropriate sky type is automatically generated from a few basic inputs such as direct normal and diffuse horizontal irradiance/illuminance to generate a continuous sky luminance distribution (Ward & Shakespeare, 1998). This is the sky model used in this study.

In this study, the on-site weather station measured global horizontal irradiance (hereafter referred as GHI) and diffuse horizontal irradiance (hereafter referred as DHI). These two irradiance quantities are linearly related with direct normal irradiance (hereafter referred as DNI), which is required by gendaylit program as input, by equation 2.

$$Direct\ Normal\ Irradiance = \frac{Global\ Horizontal\ Irradiance\ - Diffuse\ Horizontal\ Irradiance}{Cosine(Solar\ Zenith\ Angle)} \qquad (2)$$

Figure 23 shows the three irradiance quantities, at discrete time steps of 15 minutes interval, for both the experiment days together. The first day i.e. 22nd March 2014 was observed to be **clear sunny day**, and the second day i.e. 23rd March 2014 was observed to be **partly cloudy day** in Vienna. The difference in the irradiance levels supports this observation (refer Figure 23). For better observation, Figure 24, and Figure 25 shows irradiance levels for the two days separately. Typical command used to generate sky model was (Delaunay J.):

```
$ gendaylit 03 22 12:40:00 -a 48.1987 -o -16.3695 -m -15 -W 825.11 148.30 > sky.rad.
```

Date and local time are given at the starting of the command; latitude, longitude and site meridian are given suffixing -a, -o, and –m options respectively. These inputs are used to calculate the position of sun in the sky. Diffuse horizontal and direct normal irradiance are given after suffixing -W option (Delaunay J.). An example of the generated sky model is included in the APPENDIX.

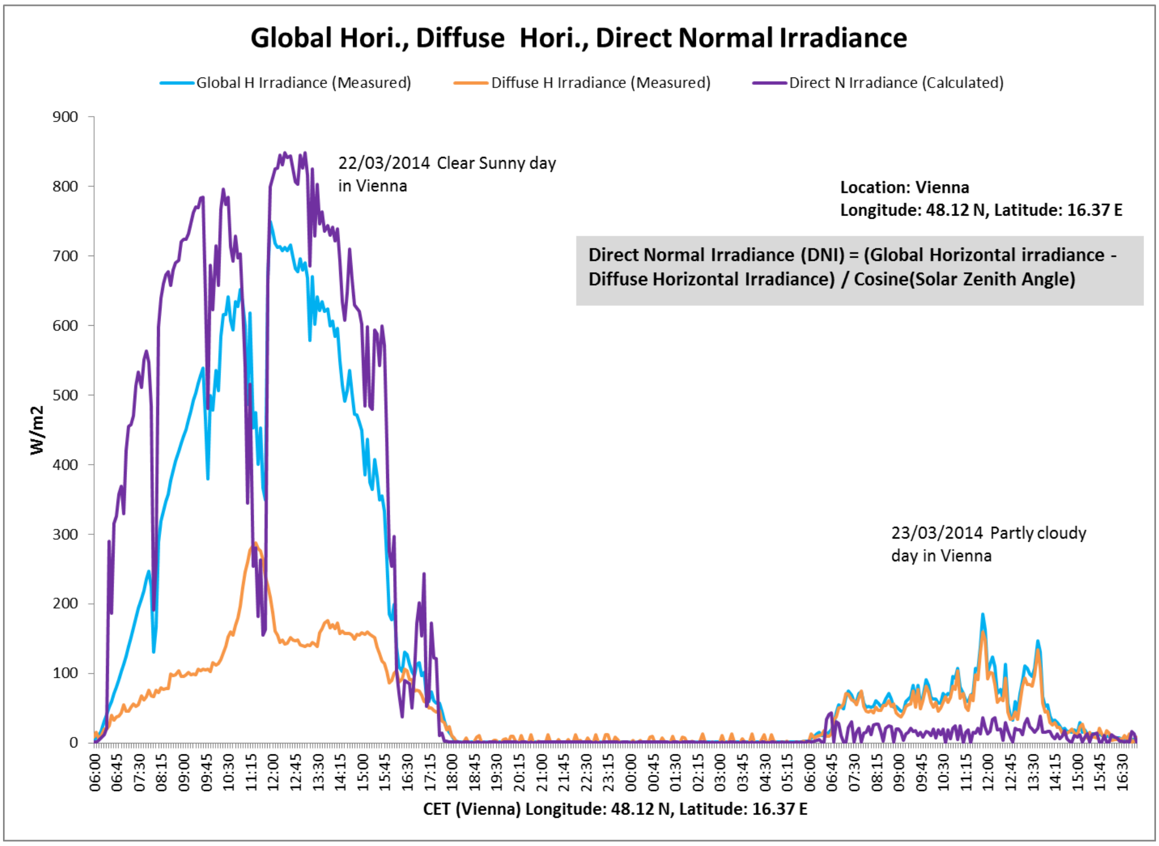


Figure 23: Figure showing measured DHI, GHI, and calculated DNI for both experiment days

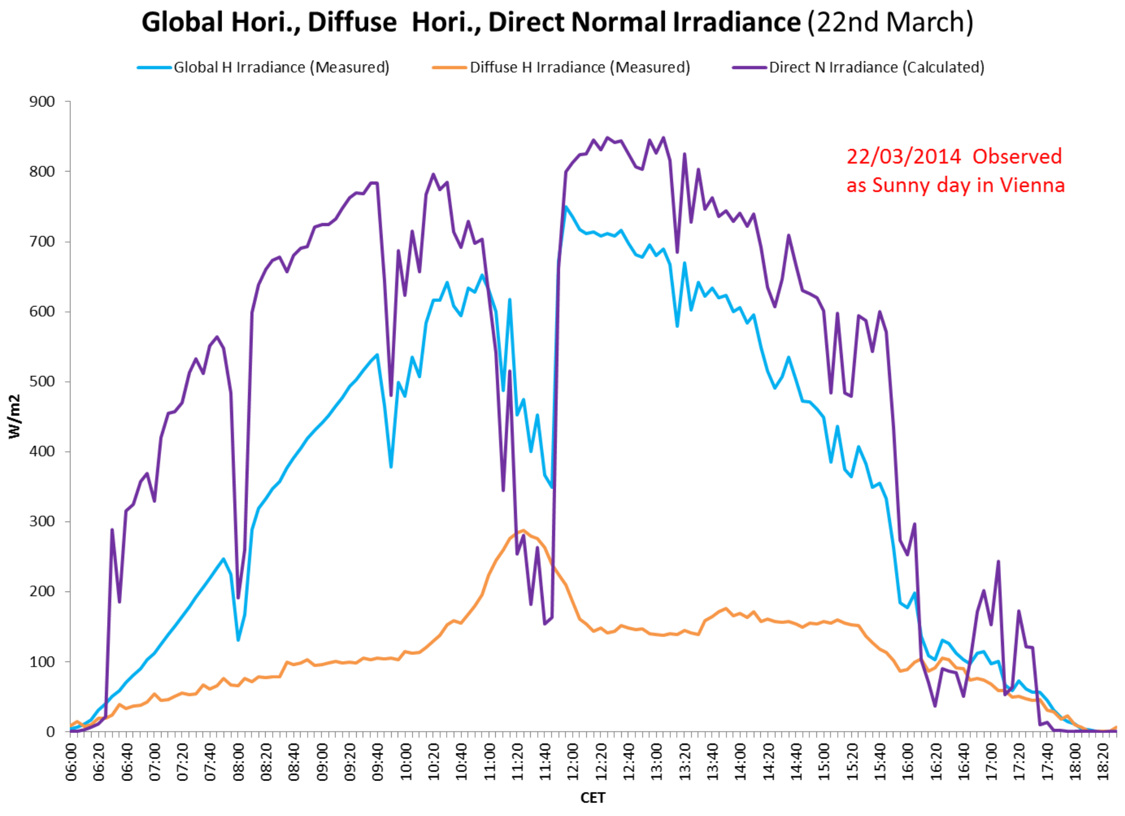
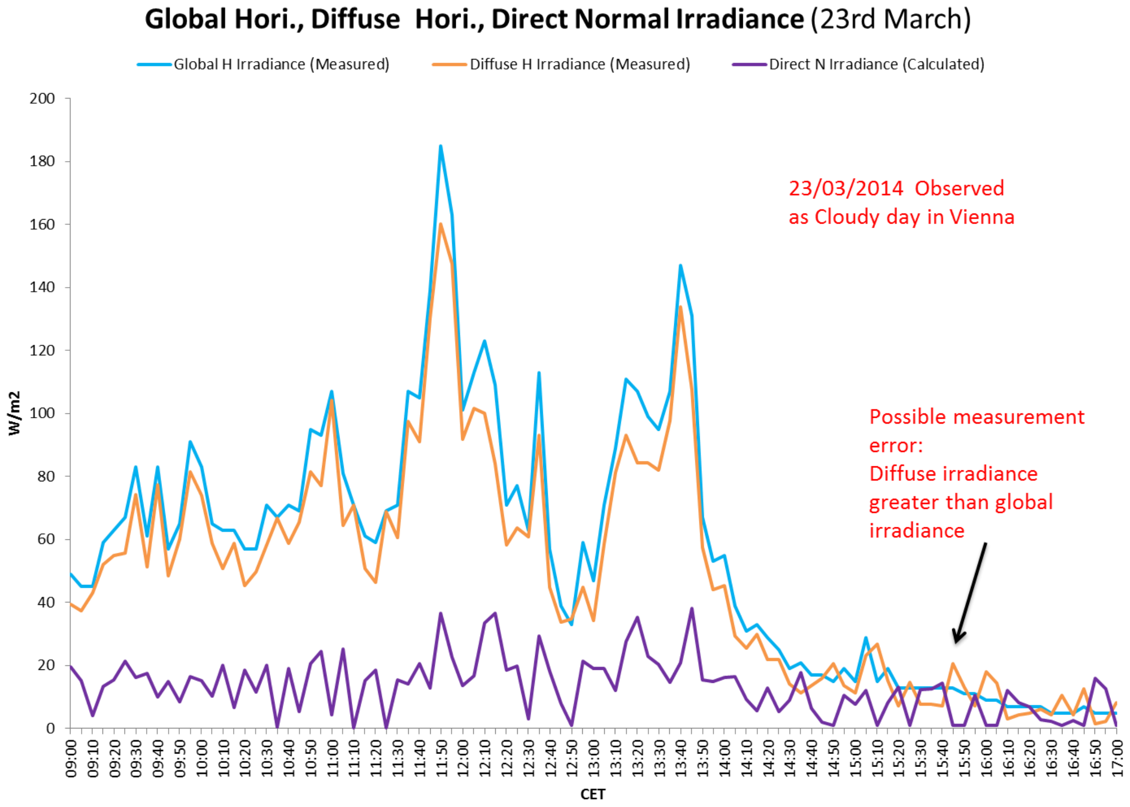


Figure 24: Figure showing measured DHI and GHI, and calculated DNI for 22nd March 2014



23/03/2014 Observed as Cloudy day in Vienna

Possible measurement error: Diffuse irradiance greater than global irradiance

Figure 25: Figure showing measured DHI and GHI, and calculated DNI for 23rd March 2014

Figure 26 qualitatively supplements Figure 23 which shows the difference in intensity of solar irradiance for the two days. The false color image at top of the image (of 22nd March) shows bright distinct profile of sun and high brightness. A more subtle sun profile is seen at the lower bottom image (23rd March). A typical command used to generate these fish-eye images is (Mardaljevic, 1999):

```
$ rpict -vta -vp 0 0 0 -vd 0 0 1 -vu 0 1 0 -vh 180 -vv 180 sunnySky.oct | falsecolor -s 32000 -l cd/m2 > skyImage.hdr
```

These images are of a model that contained only sky (no building), with view origin at the origin of the model (-vp 0 0 0), view direction looking upwards towards the sky (-vd 0 0 1). The upper limits of the luminance scale, in the false color images, were obtained by manually tracing the apparently brightest pixel in the produced HDR image using the *rvu* command (e.g. -s 32000 in the above command).

Possible measurement error can be observed from Figure 25 (above), as the measured diffuse horizontal irradiance is greater than measured global irradiance, which normally should not be the case. This may be because of the sensitivity of the sensors in monitoring station at low solar irradiance level. This causes sky model to produce erroneous sky luminance distribution which will affect resulting daylight illuminance. Due to this reason, time steps when global irradiance is less than 50 W/m2 were excluded from the analysis of results. (Refer 5.1).

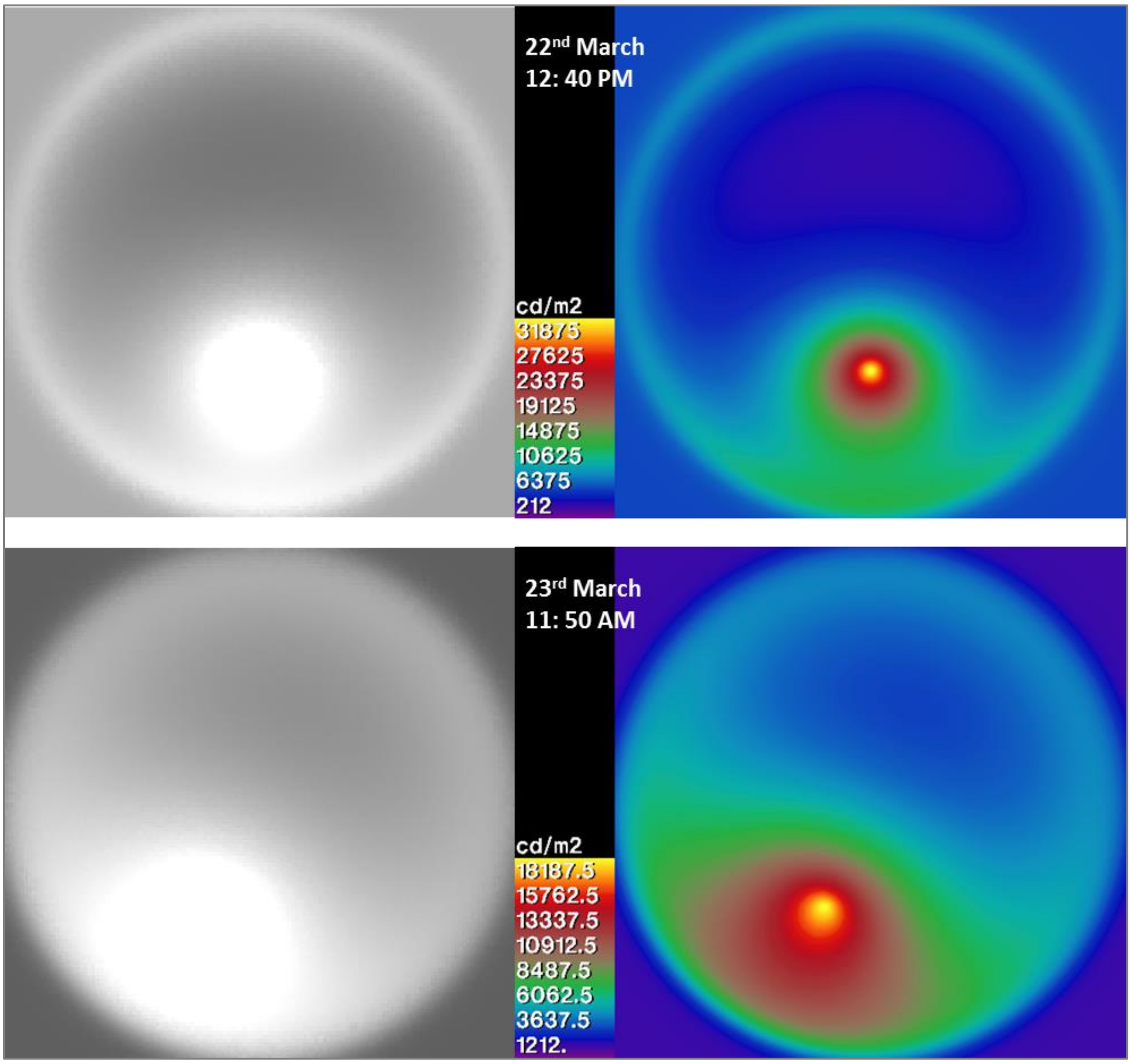


Figure 26: Figure showing the sky luminance distribution generated from *gendaylit* for both experiment days

4.3 Ambient parameter optimization

Radiance has an executive program named *rad* that allows users to qualitatively describe the desired accuracy. It then intelligently calculates ambient parameters (Ward G. L.). *RealTimeRadiance* then uses these optimized ambient parameters as argument in the *rtrace* command to calculate illuminance. The *rad* program screens the user from intricacies of finding out the adequate values of ambient parameters; however, an experienced *Radiance* user can still fine-tune the parameters to further reduce the computational cost and enhance accuracy (Mardaljevic, 1999). *RealTimeRadiance* allows such users to input customized set of ambient parameters, as shown in Figure 10. Ambient value (-av) is recommended to be set to zero for precise illuminance prediction (Mardaljevic, 1999).

For this study, initial ambient parameters obtained from *rad* program were modified, based on the learning obtained from (Mardaljevic, 1999). No significant change in results was observed after modifying the parameter, as the model converged at –ad 1536, however, significant computation cost (about 60%) was reduced. Table 6 shows the ambient parameter used in the study. Table 7 shows the three different scenarios that are considered in this study to show comparison between measured and simulated illuminance.

Table 6: Table showing the ambient parameter considered for illuminance simulation

Ambient parameter	Value
-ab	7
-ad	1536
-ar	16
-as	128

Table 7: Table describing simulation scenarios

Exp./Sim.	Date	Time	Observation
Daylight	22nd March	09:00 to 17:00	Clear sunny day
Daylight	23rd March	09:00 to 17:00	Cloudy day
Artificial light	22nd March	18:30 to 20:00	Dark outside

4.4 Error statistics

Comparison of only measured and simulated values is not sufficient to test accuracy. Mean bias error (MBE) and root mean square error (RMSE) are two statistics that are widely used in research field to analyze the accuracy of simulated data. MAE is the indication of average deviation (error) of simulated values from measured values, in both negative and positive direction. RMSE is a measure of average magnitude of error, weighted according to the square of error. It provides the average magnitude of simulated errors but does not give its direction. Because it is a squared quantity, RMSE is influenced more strongly by large errors than by small errors. This study will focus mainly on relative error (RER), MAE and RMSE, as they can be compared with other similar research studies;

however, other statistics are also included for reference of the reader. The equations for these statistics are as follows (Mardaljevic, 1999), (Tahmasebi & Mahdavi, 2012).

Relative Error (RER):

$$RER = 100 \times \left(\frac{Simulated - Measured}{Measured}\right)$$

Mean Absolute Error (MAE):

$$MAE = \frac{1}{N}\sum_{i=0}^{N} |Simulaed_i - Measured_i|$$

Mean Bias Error (MBE)

$$MBE = 100 \times \frac{1}{N}\sum_{i=0}^{N}\left(\frac{Simulated_i - Measured_i}{Measured_i}\right)$$

Root Mean Square Error (RMSE)

$$RMSE = 100 \times \sqrt{(\frac{1}{N})\sum_{i=0}^{N}\left(\frac{Simulated_i - Measured_i}{Measured_i}\right)^2}$$

Coefficient of variation of the Root Mean Square Deviation (CV(RMSD))

$$CV(RMSD) = \frac{\sqrt{\frac{\sum_{i=1}^{N}(Measured_i - Simulated_i)^2}{N}}}{\overline{Measured}}$$

The most influential statistic in determining the accuracy of a model is RMSE (Baharuddin, 2013), however, there is no range of RMSE to ascertain more or less accuracy, as it mostly depends upon the context. For this reason, MAE of 10% and RMSE of 20% will be used to determine the accuracy (Mardaljevic, 1999). Table 8 gives a subjective meaning to different ranges of RMSE (Baharuddin, 2013) and MAE.

Table 8: Qualitative range for different error statistic

	RMSE %	MBE ±%
Good accuracy	<= 20	< 10
Moderate accuracy	20 - 40	10 - 30
Low accuracy	> 40	> 30

4.5 Error inference from similar research

This section tries to infer the reasoning behind simulation error from other similar validation studies. It then tries to relate those reasoning with the context of this study. This section can be helpful in gaining useful insight over the different kinds of error that may be expected from this experiment, which are discussed later in section 5.

4.5.1 Mismatch error

Mismatch error refers to the positional mismatch/misalignment between the simulation model and reality. Even a misalignment of just 1mm in modeling a window frame can produce large errors (Mardaljevic, 1999). This is especially true in sensor positions that are subject to high direct solar beam, reflected light from the surrounding building surfaces, or direct light from lamps. The first two situations are mostly found in locations near windows. Misalignment in modeling a small physical surface/component such as window bars, handle etc. can cast dark shadows on the sensors which may result in high RER (Mardaljevic, 1999).

Another form misalignment could be from building's North rotation offset, which relates the orientation of the building with true North.

All these uncertainties, or may be others, makes it virtually impossible to relate a particular RER with a single misalignment form (Mardaljevic, 1999).

In this study, though the windows seldom face sun, there is a potential if getting high RER especially at locations near window since the windows receive a lot of reflected light from the white surface of the opposite building (refer Figure 13) as also observed in Figure 14.

4.5.2 Error in modeling sky luminance distribution

The most important constituent of any daylight simulation is imitation of sun and sky luminance (or brightness) distribution/pattern, as these two are the only sources of daylight illumination. A good understanding about the creation of sky in *Radiance* is crucial in analyzing illuminance results; therefore, it will be discussed here in detail. However for more details, the reader is encouraged to read (Mardaljevic, 1999), (Ward G. J., 1994), (Ward & Shakespeare, 1998).

Low accuracy was observed from vertical illuminance sensors facing North, as compared to sensors facing other directions (Mardaljevic, 1999), (Baharuddin, 2013) using Perez sky model. The rationale behind this must be the hybrid deterministic/stochastic sampling approach of *Radiance*.

In simulations, in general, a deterministic model has no stochastic (random) elements and the entire input and output relation of the model is conclusively determined. A stochastic model has one or more stochastic element. The system having stochastic element is generally not solved analytically.

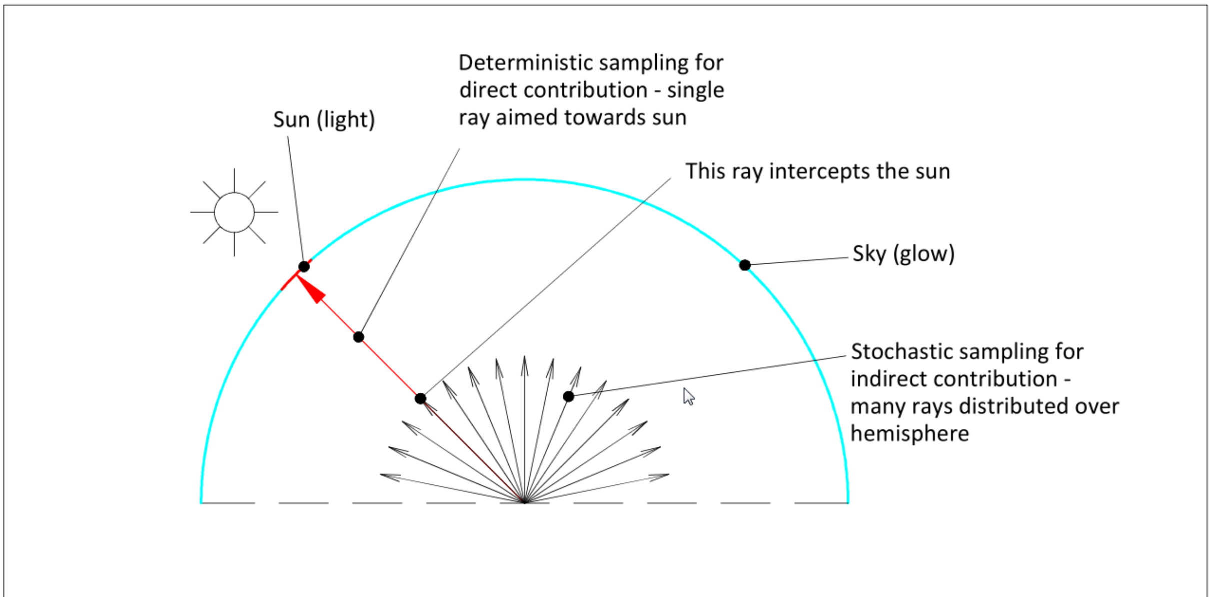
In the case of simulating a stochastic model, a random number is normally generated by some method to execute trial. Such a simulation is called the Monte Carlo method or Monte Carlo simulation (Ayani, 2003).

In *Radiance*, deterministic sampling is performed on surfaces with *light* as material type; and stochastic sampling on surfaces with *glow* with material type. Material *light* is applied to direct or concentrated sources of light such as sun or light fixtures; and material *glow* is applied to other indirect sources of light such as sky, as shown in Figure 27. *Radiance* uses hybrid deterministic/stochastic sampling approach to trace back the source of light (Mardaljevic, 1999). This means that during simulation, rays are traced backward from the point of interest to the source of light. This determines whether the point is illuminated by direct light, or indirect light (inter-reflected light). In case of direct sunlight (or artificial light), the direct source of light (sun or light fixture) is known, so a single ray (shadow ray) is sent from the point of intersection with the scene to the source using deterministic sampling. In case of indirect light from sky (or any type of reflected light), it is difficult to pin point the light source therefore a number of shadow rays are sent in different direction, from the point of intersection, to determine if this point is illuminated by the light source. This is done, in *Radiance*, by a stochastic sampling procedure known as hemispherical sampling (Mardaljevic, 1999).

The above mentioned observation of achieving comparatively lesser accuracy from vertical sensors facing North sky (diffuse sky) could reasonably be due to the randomness (stochastic element) involved in the simulation. From this understanding, it can be inferred that building with only North facing windows would observe lesser accuracy in internal illuminance as compared to buildings with South facing windows (Grobe, 2014).

In this study, the seminar hall has only North-East facing windows which seldom face direct sun, as evident from Figure 28. This means that these windows get most of the illumination from the diffuse North sky, and therefore may observe lesser accuracy. This phenomenon could be even more profound (i.e. even lesser accuracy) in sky model with diffuse profile of sun, for example intermediate sunny sky and/or overcast sky. Perez All Weather sky model selects appropriate type of sky based on the input irradiance levels; therefore, low accuracy may be expected when the measured irradiance levels are smaller.

Therefore, mapping of actual sky luminance distribution/pattern (e.g. from sky-scanner) is more relevant buildings with only North facing windows.


Deterministic sampling for direct contribution - single ray aimed towards sun
Sun (light)
This ray intercepts the sun
Sky (glow)
Stochastic sampling for indirect contribution - many rays distributed over hemisphere

Reference: This figure has been reproduced from [Mardaljevic J., 1999]

Figure 27: Figure showing *Radiance's* deterministic and stochastic sampling methods for light (sun) and glow (sky) material types

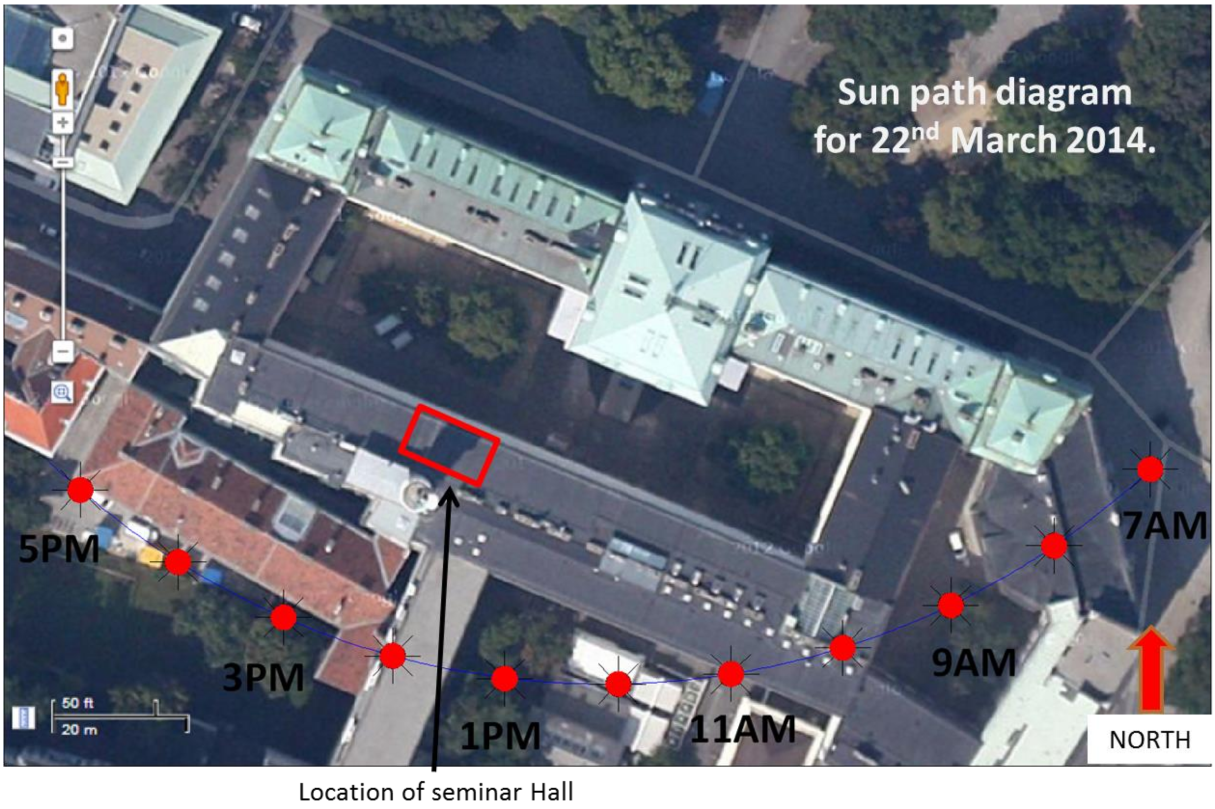


Figure 28: Figure showing the sun-path and seminar hall during the experiment days

Real sky observes a high variance in luminance distribution due to breaks in clouds. However, mathematical sky model such as Perez model creates a continuous sky with no cloud breaks from minimal input of only direct normal and diffuse horizontal irradiance. Also it should be considered that sky models provide luminance distribution that represent average sky luminance distribution over the year. Due to this reason, there could be errors in comparing discrete simulation results with measurement values (Grobe, 2014).

This issue can be catered well with the use of sky-scanner because then the luminance distribution of the actual local sky can be modeled, in *Radiance*, more effectively.

4.5.3 Error in surface property measurement

It is rather easy to measure the reflectance of an opaque surface such as wall, furniture etc., however, translucent surface requires much product information, and experience in measurement as well as modeling, as it involves scattering of light.

In this study, reflectance of opaque surfaces was measured as discussed in 4.2.1, however, due to unavailability of specifications of the translucent partition door (shown in Figure 14), it was modeled based on specification of a similar looking product (refer APPENDIX for specifications). The effect of this on simulation results might be hard to identify, however, it is advised to precisely measure or use manufacturer's specification for modeling such complex material.

4.5.4 Photometric data conversion

In Radiance, the IES file of the luminaire is converted into radiance format using *ies2rad* program. A correctly converted luminaire must have the same luminous intensity distribution as the actual luminaire, and also its shape/appearance should remain same. Ies2rad program creates a simple emitting surface such as polygon, sphere, etc. and then maps appropriate luminous intensity distribution over it as a pattern. Due to the approximation in the program, it lacks the ability to precisely represent the geometry of complex luminaire type such as those with direct/indirect light distribution. To overcome this, precise luminaire geometry should be modeled, using specification from the manufacturer, and then placed into an imposter object made of *illum* material type. This method is comparatively complex and more time consuming, and it also increases the simulation time (Cvetković, Lenard, & Mudri, 2005).

In this study, direct/indirect luminaries are present as previously mentioned in 4.2.2, and *ies2rad* program have been used to convert IES file to *Radiance* format, as it makes the process (use case) simpler for the user, and also it is computationally cheap. Also from Figure 22 the light distribution of the converted luminaire is qualitatively analyzed and found suitable. However, inaccurate results may be in-countered.

5 RESULTS AND DISCUSSION

5.1 Daylighting cases

It is important to compare outdoor daylight illumination from both measured and simulation realms before comparing the indoor illuminance. If outdoor illumination from the simulation does not show good accuracy with the measured outdoor illumination, then internal daylight illumination from simulation cannot be accurate (Mardaljevic, 1999). Figure 29 and Figure 30 compares outdoor horizontal illuminance, measured and simulated, for both the experiment days; and Figure 31 shows error in simulation results; Table 9 and Table 10 shows different statistics.

During data analysis for partly cloudy day, some outliers were observed in RERs starting from the late afternoon. This happened because the measured diffuse horizontal irradiance was somehow greater than the measured global horizontal irradiance (refer Figure 25) which normally shouldn't be the case. Therefore, the direct normal irradiance was calculated improperly, and resulted in improper sky luminance distribution during that duration. In order to improve the quality (remove confounds) of the results, time steps when measured global irradiance was less than 50 W/m2 were excluded from the analysis for both the days, as previously discussed in 4.2.3.1. Such low irradiance levels were mostly observed during partly-cloudy day (refer Figure 25), and due to their exclusion gaps are shown in time series graphs below. This observation raises the questions on uncertainty in measurements of irradiance values from the weather station, especially during low irradiance level. The analysis of measurement uncertainty of different sensors is outside the scope of this project, and hence, although being crucial, has not been dealt with in this study.

For outdoor sensor during clear-sunny day case, RER mostly falls within the range of ±15 % (with a few outliers); however, for partly-cloudy case the error is dominantly negative. This can be inferred as the reason for high negative RER and MBE for all the internal virtual sensors for partly-cloudy day case, as shown in figures below and also in Table 10 as a summary. For outdoor sensors, MBE and RMSE for clear-sunny sky (MBE -8.79%, RMSE 16.68%) fall within the previously established range of "good accuracy" (refer 4.4) but for partly-cloudy day case (MBE -14.59%, RMSE 16.60%) they fall in the range of "moderate accuracy". Therefore, at this stage only it can be estimated that internal illuminance on clear-sunny sky case will be more accurate than partly-cloudy case. As anticipated, average indoor horizontal MBE, and average indoor horizontal RMSE, for clear-sunny sky, falls within the range of "good accuracy" (refer Table 9), however for partly-cloudy day they fall within the range of "moderate accuracy" (refer Table 10). H1, and H4 contributed large negative bias to the averages for both days, however H9 contributed to positive bias. These are the sensors located closest to the windows, and therefore might be subject to shadows or strong reflected light (mismatch error) as discussed earlier in 4.5.1. As reported by (Mardaljevic, 1999), RMSE tends to decrease with increase in distance from the window. Similar trend has been observed in all the sensor rows during partly cloudy day case. However, such trend is not clearly observed in clear-sunny day case.

For a better comparison between internal illuminance for all the three experiment cases, the time-series illuminance graphs and RER histogram (from Figure 32 to Figure 43) are arranged in the order of increasing distance of sensors from the window (refer for example Figure 15). The vertical axes in both these graphs are matched for better comparison of the distribution. Time series graphs and histograms for both experiment days are placed on the same page for better comprehension of the reader Individual histogram also shows MBE and RMSE. The bin size is 1%, and the distribution is normalized to the total of 1.

As previously established in 4.5.2, better accuracy is obtained during clear sunny day as compared to intermediate/partly cloudy day due to deterministic and stochastic sampling procedure of *Radiance*. Also Perez sky model (a mathematical model), create a continuous pattern of sky however, in actual there is variance in sky due to cloud breaks. These factors strongly suggest that inaccuracies observed at indoor virtual sensor locations (Figure 27 to Figure 51) can be explained by the inaccuracies observed from sky model (Figure 29 to Figure 31).

5.2 Artificial lighting case

Artificial lighting case showed average MBE and RMSE of 7.42% and 26.44% respectively, as shown in Table 11. H3 contributed extreme overestimation (average RER of 82%) to the average. Sensor H3, H10, and V1 contributed bias to the average. Removing these three sensors produced an average MBE and RMSE of 2.49% and 10.68% respectively, which falls within the range of “good accuracy”. H3, H10 are oriented parallel to the luminaire, which is same with majority of sensors, however V1 is oriented perpendicular to the luminaire, as can be seen in Figure 15. The observed bias may be due to inaccuracy in modeling the exact shape, as well as location, of the luminaire which affect the transmission of light at particular angles, as discussed previously in 4.5.1 and 4.5.4.

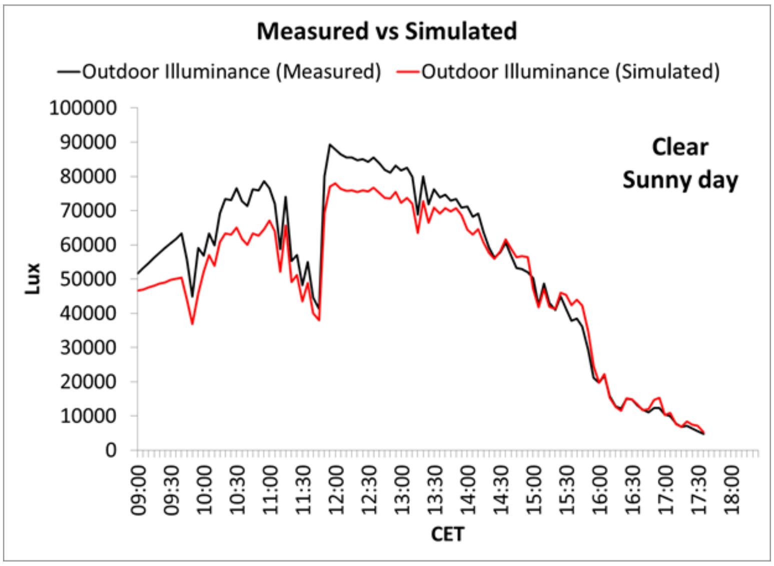


Figure 29: Simulated and measured daylight outdoor illuminance values on clear-sunny day

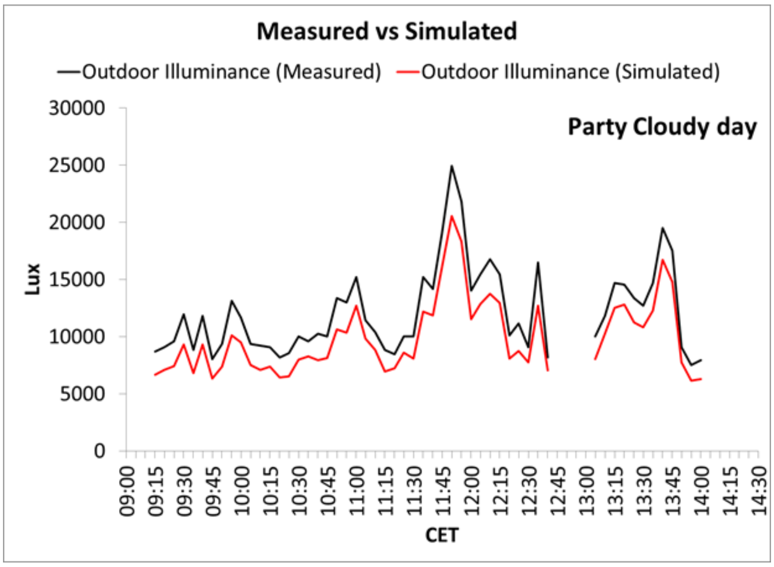


Figure 30: Simulated and measured daylight outdoor illuminance values on partly-cloudy day

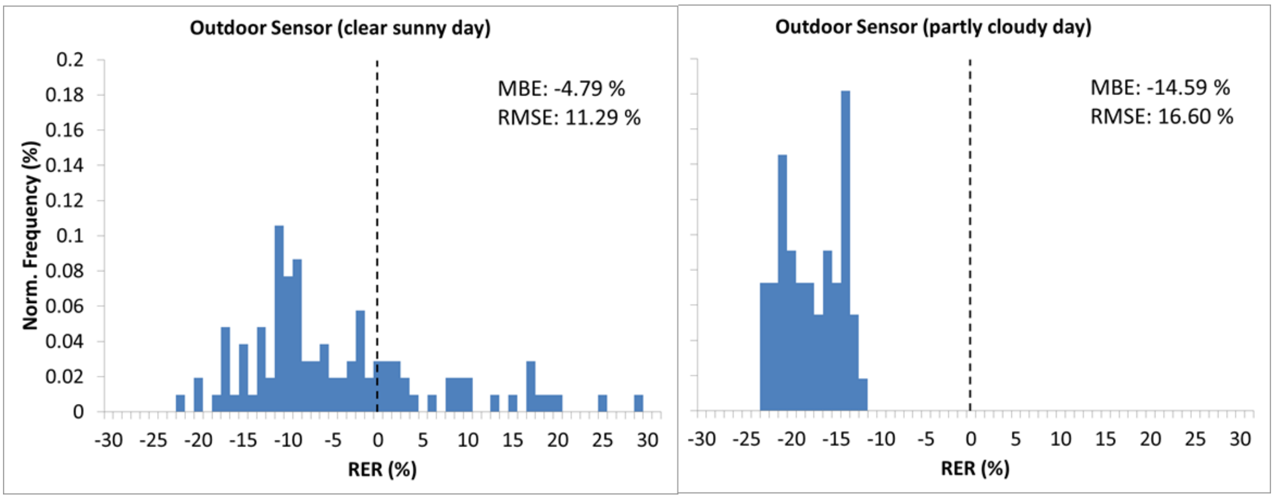


Figure 31: Error statistics for outdoor illuminance for both clear-sunny and partly-cloudy day

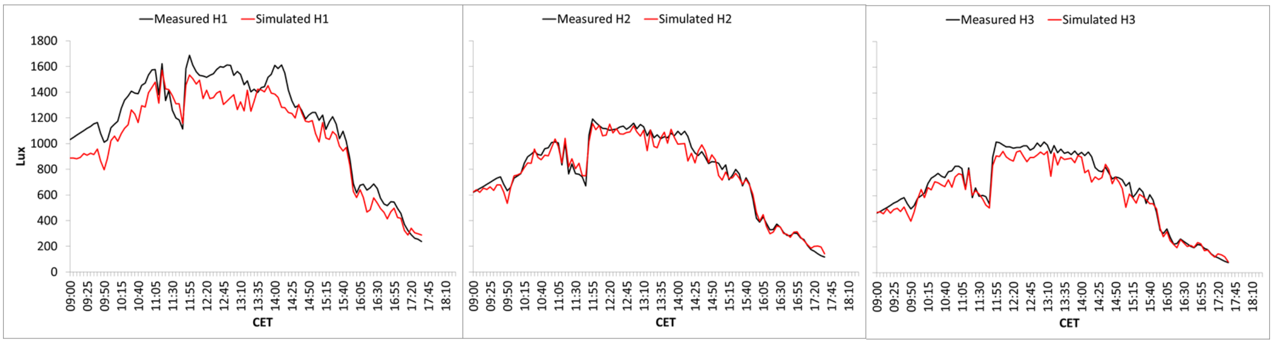


Figure 32: Simulated and measured daylight illuminance for H1, H2, H3 (row 1) on clear-sunny day

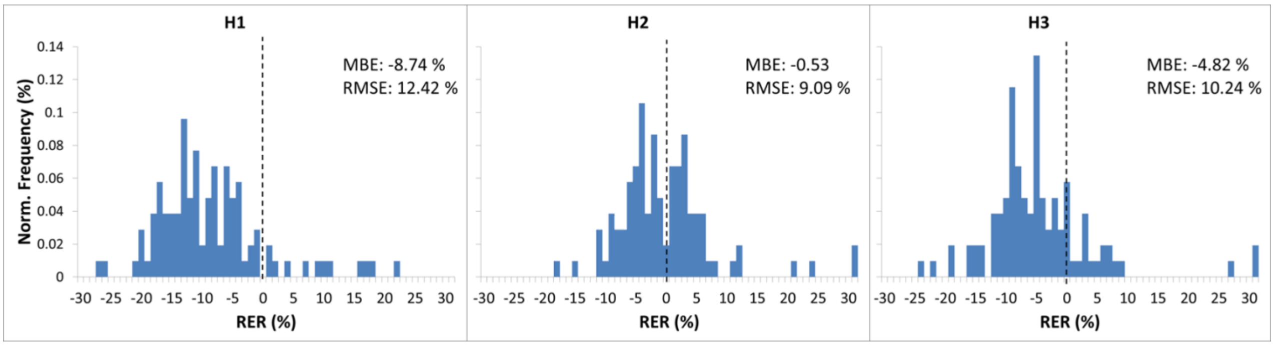


Figure 33: Error statistics for H1, H2, H3 (row 1) on clear-sunny day

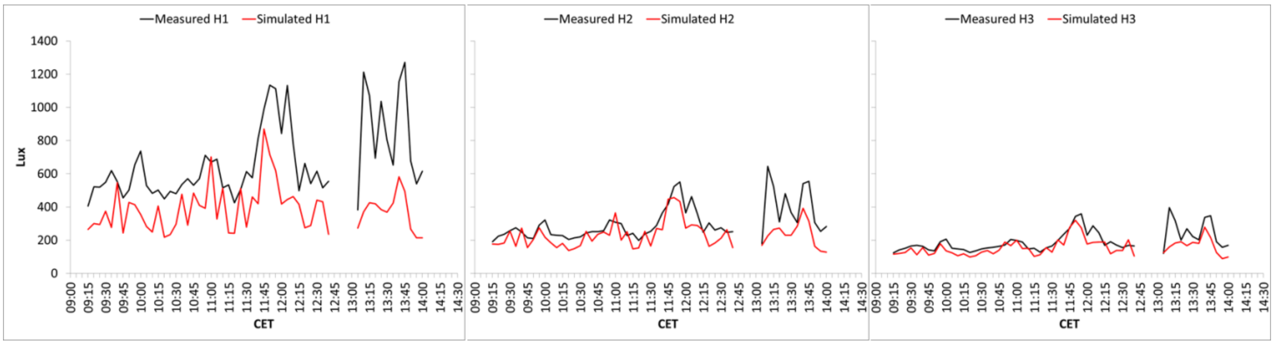


Figure 34: Simulated and measured daylight illuminance for H1, H2, H3 (row 1) on partly-cloudy day

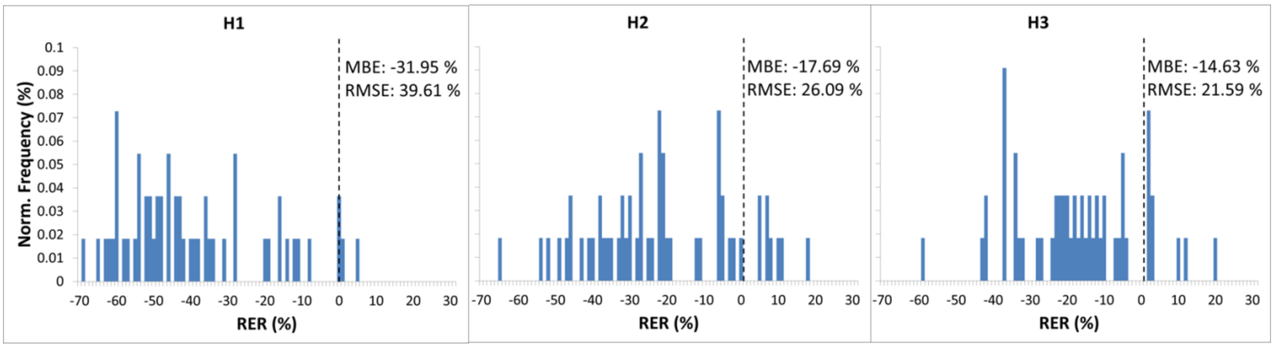


Figure 35: Error statistics for H1, H2, H3 (row 1) on partly-cloudy day

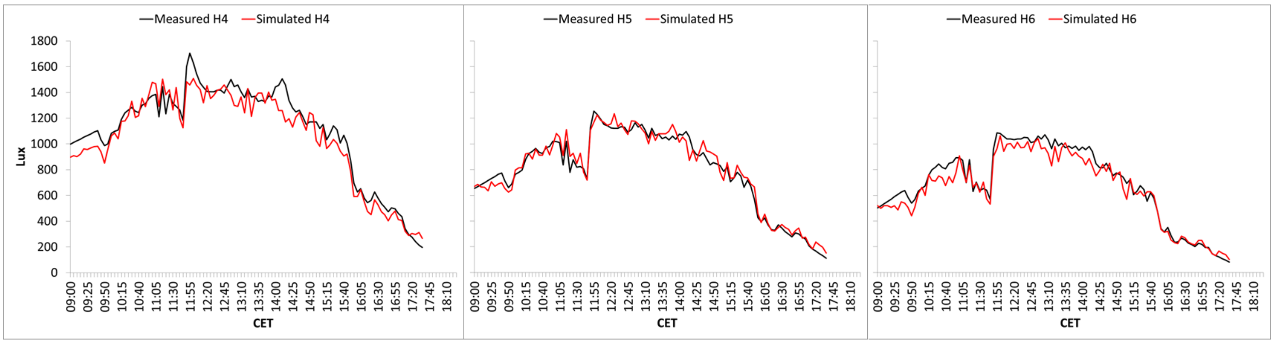


Figure 36: Simulated and measured daylight illuminance for H4, H5, H6 (row 2) on clear-sunny day

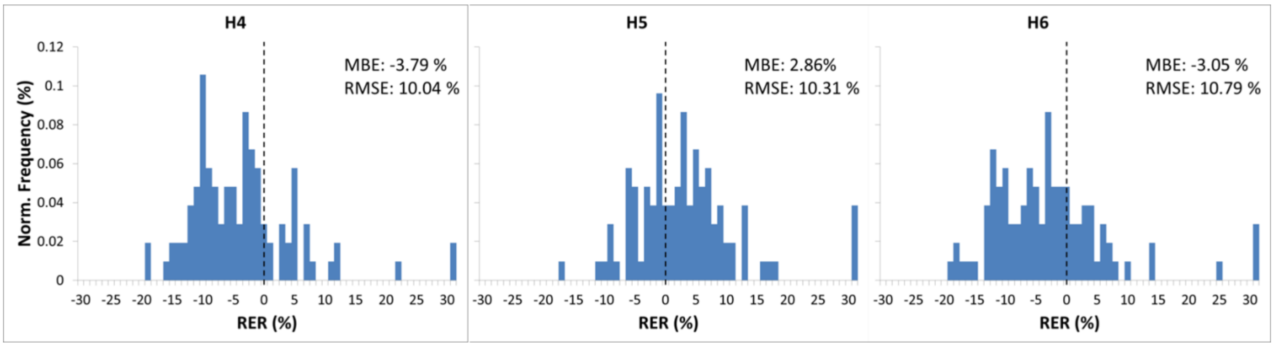


Figure 37: Error statistics for H4, H5, H6 (row 2) on clear-sunny day

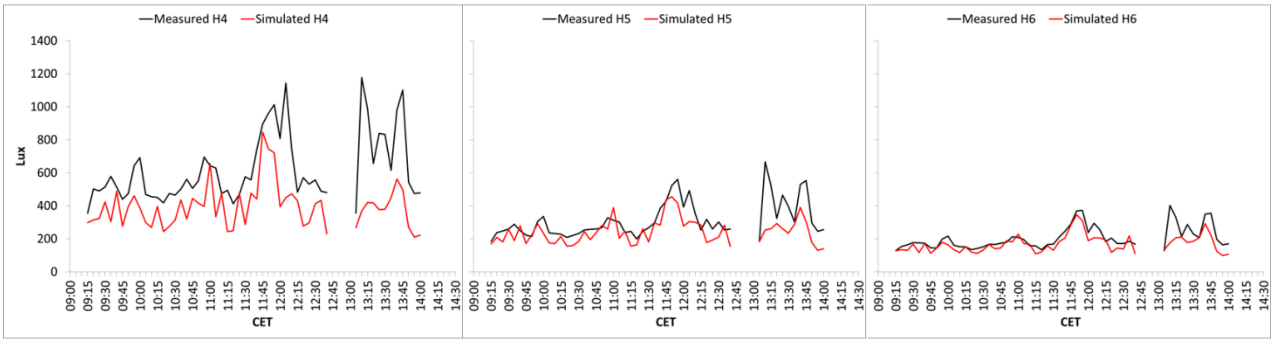


Figure 38: Simulated and measured daylight illuminance for H4, H5, H6 (row 2) on partly-cloudy day

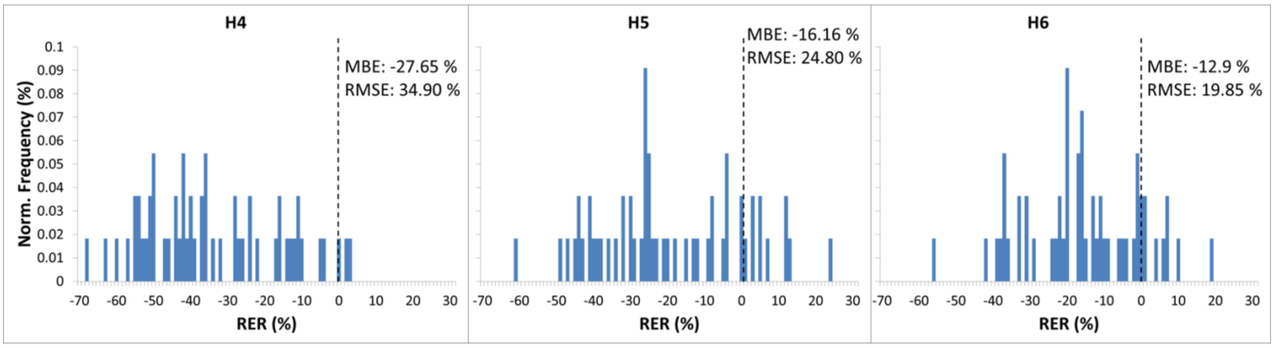


Figure 39: Error statistics for H4, H5, H6 (row 2) on partly-cloudy day

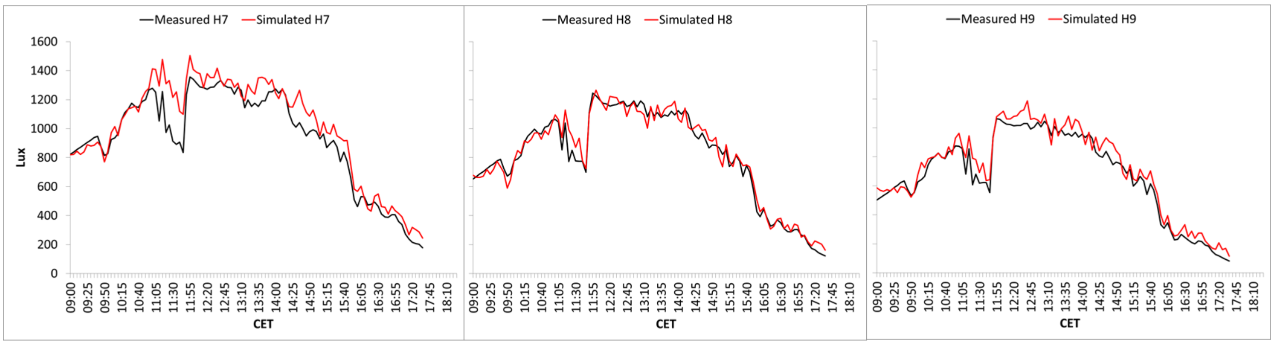


Figure 40: Simulated and measured daylight illuminance for H7, H8, H9 (row 3) on clear-sunny day

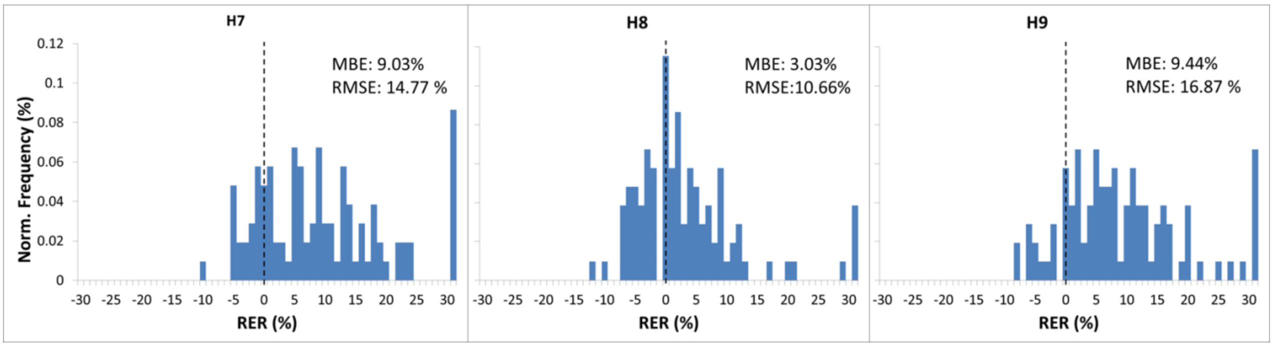


Figure 41: Error statistics for H7, H8, H9 (row 3) on clear-sunny day

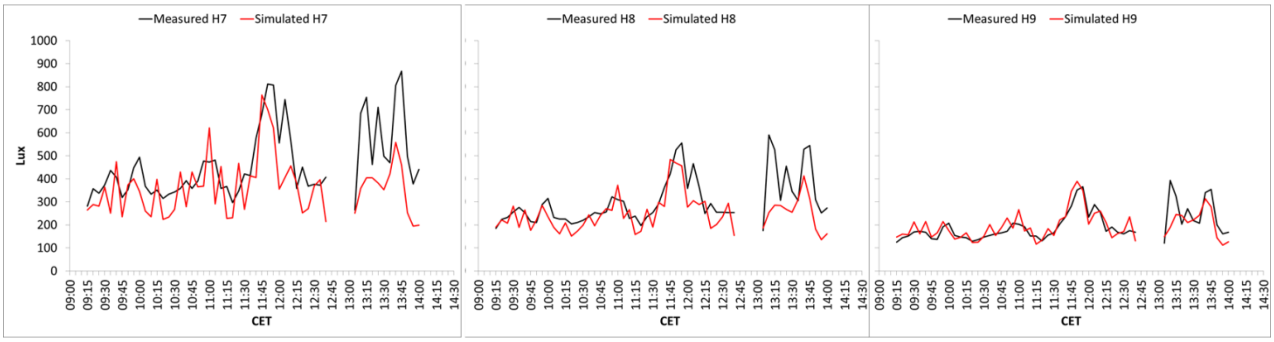


Figure 42: Simulated and measured daylight illuminance for H7, H8, H9 (row 3) on partly-cloudy day

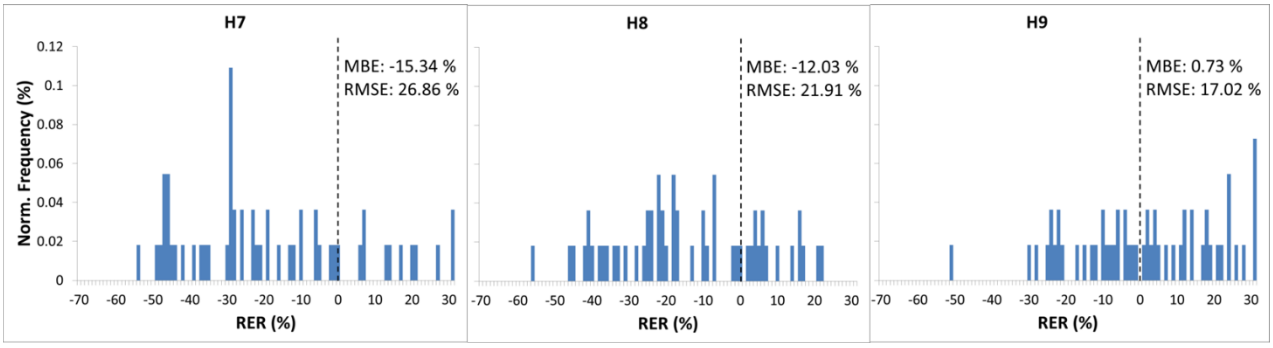


Figure 43: Error statistics for H7, H8, H9 (row 3) on partly-cloudy day

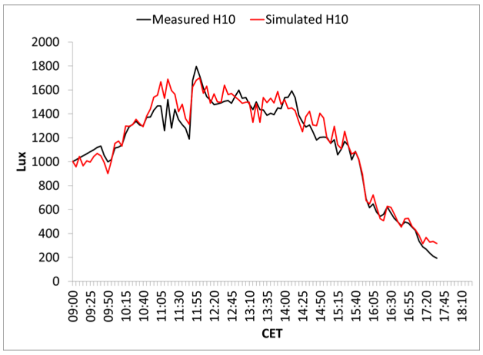


Figure 44: Simulated and measured daylight illuminance for H10 on clear-sunny day

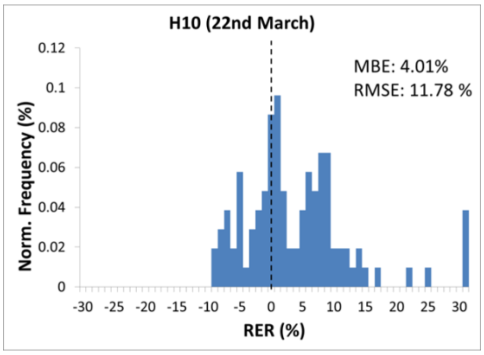


Figure 45: Error statistics for H10 on clear-sunny day

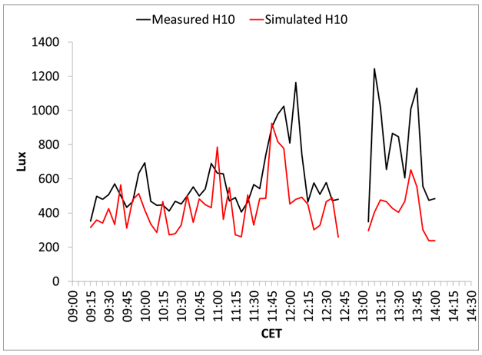


Figure 46: Simulated and measured daylight illuminance for H10 on partly-cloudy day

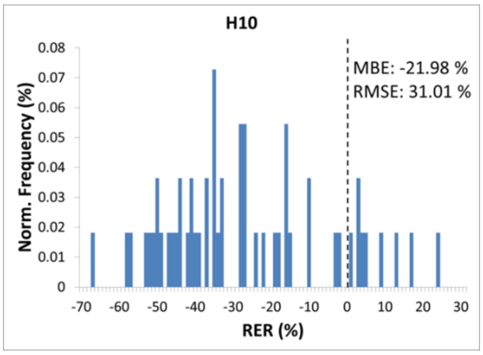


Figure 47: Error statistics for H10 on partly-cloudy day

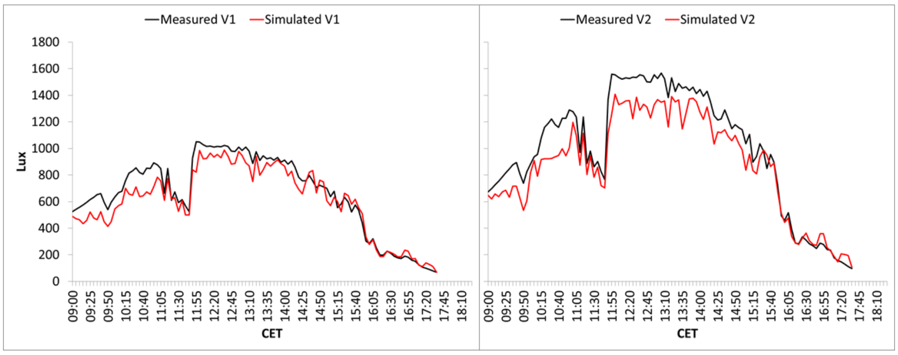


Figure 48: Simulated and measured daylight illuminance for V1, V2 on clear-sunny day

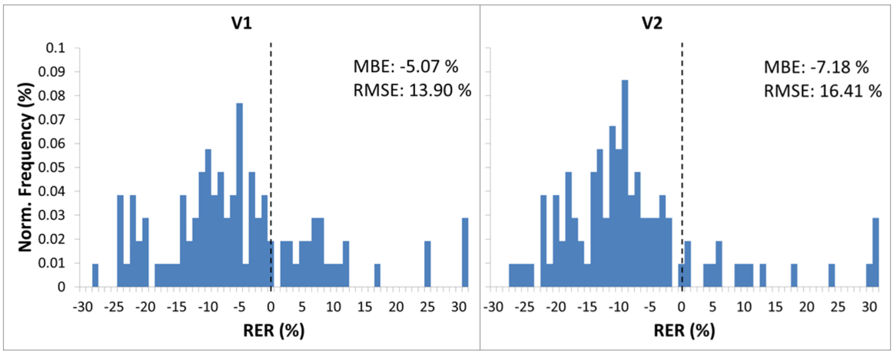


Figure 49: Error statistics for V1, V2 on clear-sunny day

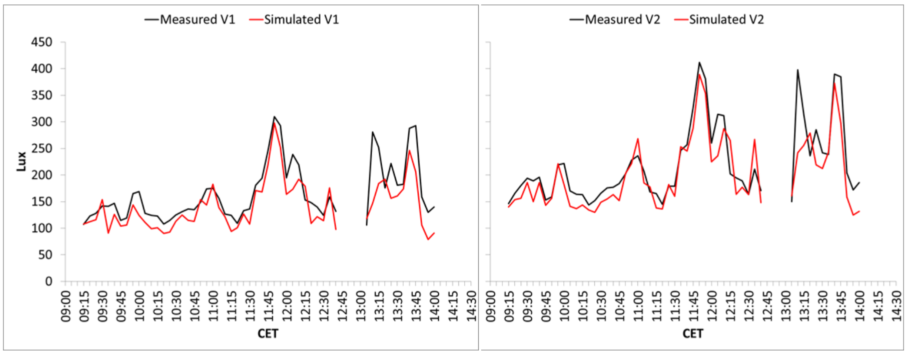


Figure 50: Simulated and measured daylight illuminance for V1, V2 on partly-cloudy day

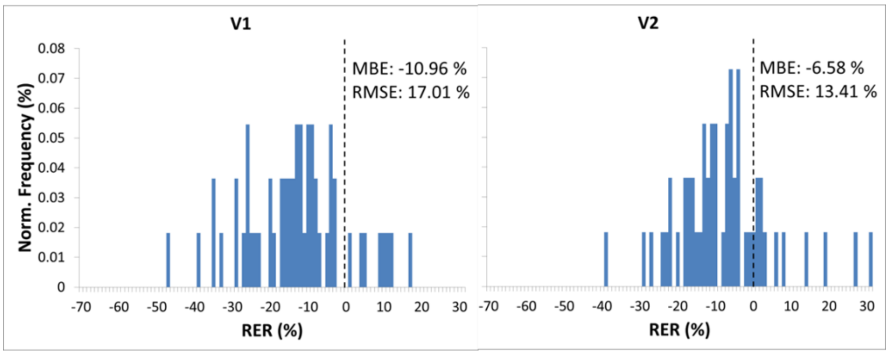


Figure 51: Error statistics for V1, V2 on partly-cloudy day

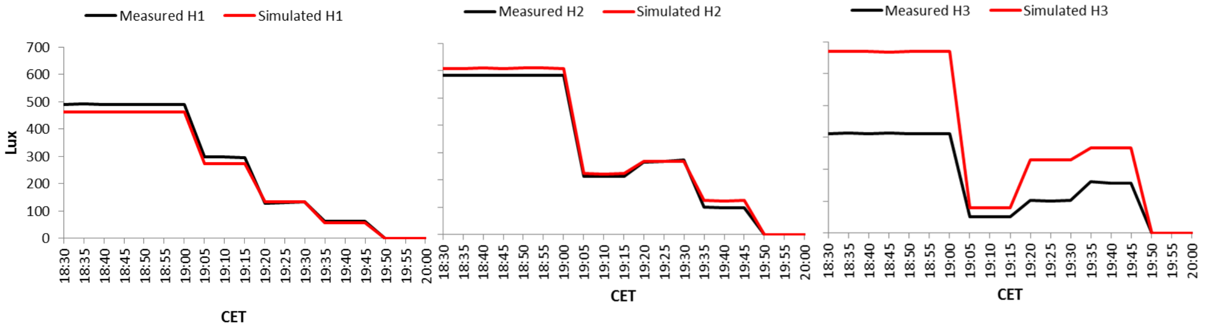


Figure 52: Simulated and measured daylight illuminance for H1, H2, H3 (row 1) for artificial lighting case

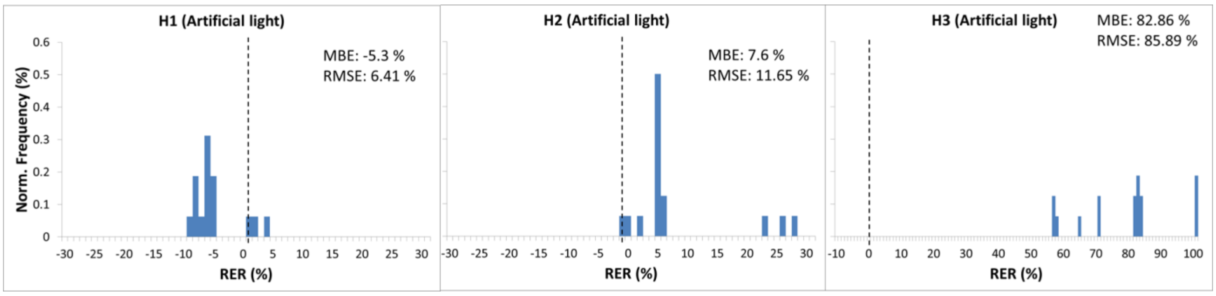


Figure 53: Error statistics for H1, H2, H3 on artificial lighting case

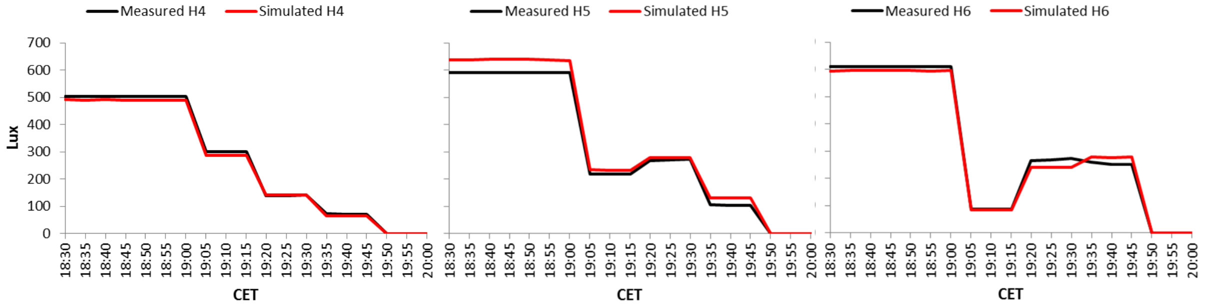


Figure 54: Simulated and measured daylight illuminance for H4, H5, H6 (row 2) for artificial lighting case

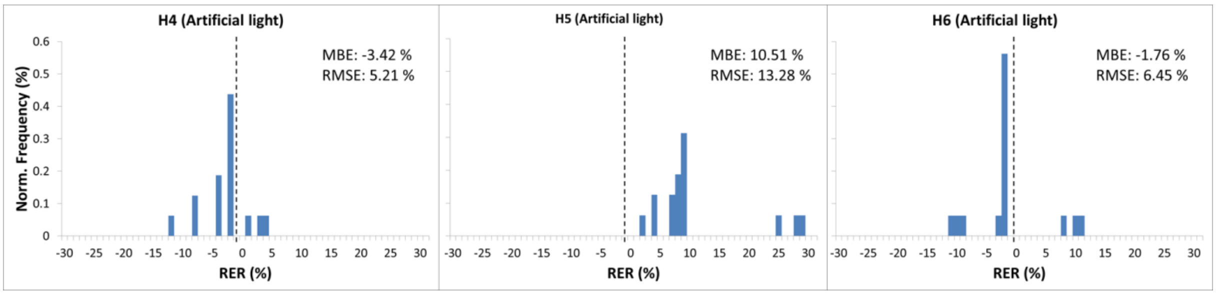


Figure 55: Error statistics for H4, H5, H6 on artificial lighting case

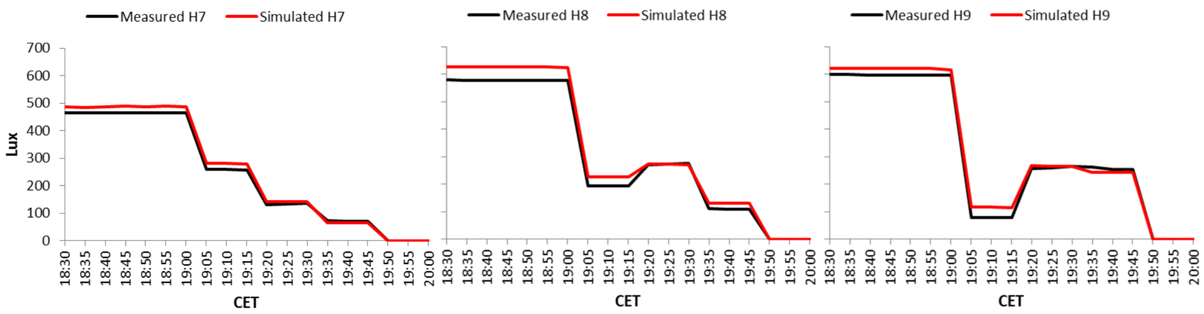


Figure 56: Simulated and measured daylight illuminance for H6, H7, H8 (row 3) for artificial lighting case

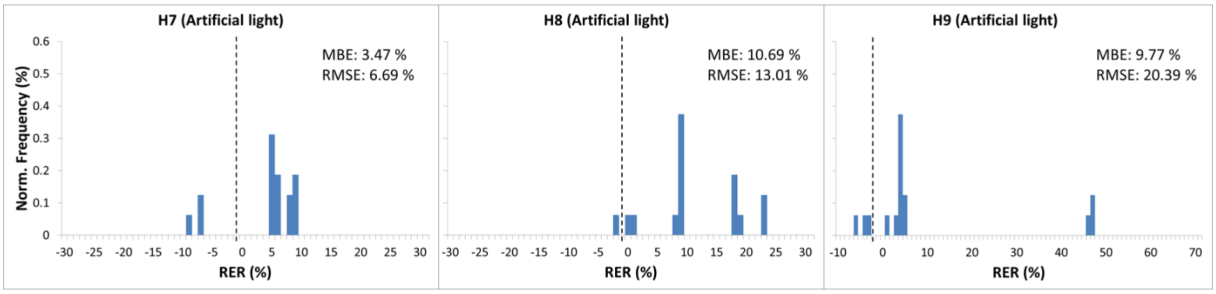


Figure 57: Error statistics for H6, H7, H8 on artificial lighting case

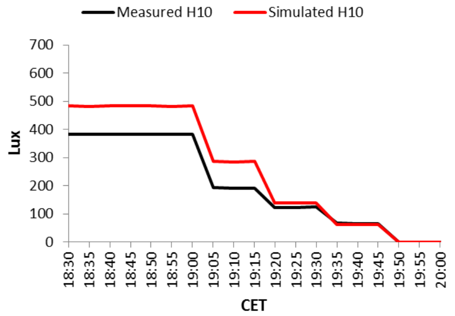


Figure 58: Simulated and measured daylight illuminance for H10 for artificial lighting case

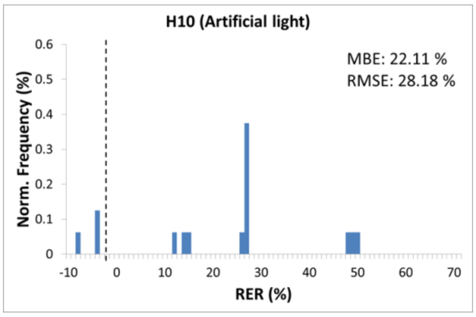


Figure 59: Error statistics for H10 on artificial lighting case

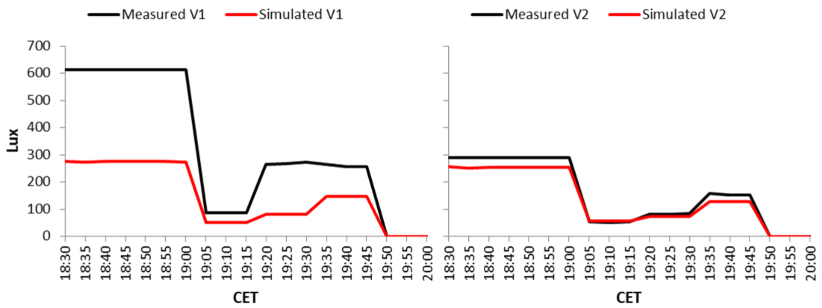


Figure 60: Simulated and measured daylight illuminance for V1, V2 for artificial lighting case

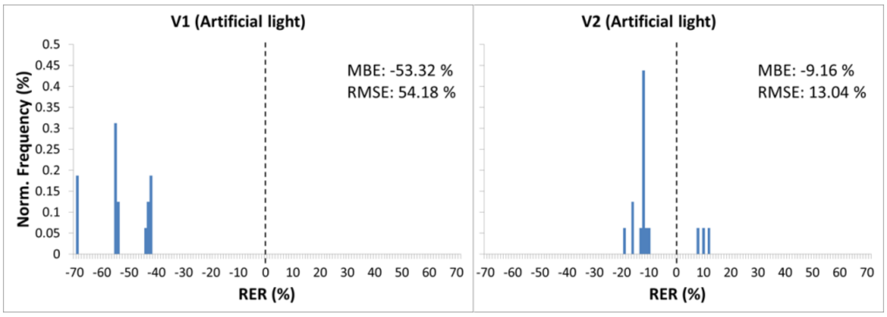


Table 9: Table showing different error statistics for daylight simulation on 22nd March (clear sunny day)

22nd March (Illuminance, Daylight)														
		Row 1			Row 2			Row 3						Avg. indoor
	Outdoor	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	V1	V2	hori. error
MAE	5329.94	120.96	36.82	49.07	73.37	39.25	48.14	80.07	38.61	50.60	63.71	67.61	120.88	60.06
MBE	-4.79	-8.74	-0.53	-4.82	-3.79	2.86	-3.05	9.03	3.03	9.44	4.01	-5.07	-7.18	0.74
RMSE	11.29	12.42	9.09	10.24	10.04	10.31	10.79	14.77	10.66	16.87	11.78	13.90	16.41	11.70
CV(RMSD)	12.37	12.19	6.20	9.81	8.29	6.40	9.05	11.80	6.37	9.42	7.54	13.10	14.90	-
R Square	0.97	0.94	0.98	0.98	0.95	0.97	0.97	0.93	0.97	0.97	0.96	0.95	0.97	-

Table 10: Table showing different error statistics for daylight simulation on 23rd March (partly cloudy day)

23rd March (Illuminance, Daylight)														
		Row 1			Row 2			Row 3						Avg. indoor
	Outdoor	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	V1	V2	hori. error
MAE	1728.16	227.33	65.45	33.84	182.80	63.51	31.62	102.29	54.17	25.74	162.45	22.04	22.41	94.92
MBE	-14.59	-31.95	-17.69	-14.63	-27.65	-16.16	-12.90	-15.34	-12.03	0.73	-21.98	-10.96	-6.58	-16.96
RMSE	16.60	39.61	26.09	21.59	34.90	24.80	19.85	26.86	21.91	17.02	31.01	17.01	13.41	26.36
CV(RMSD)	3.73	26.06	12.69	8.01	22.99	12.31	7.22	15.22	10.24	5.86	20.60	4.88	3.44	-
R Square	0.99	0.37	0.64	0.73	0.42	0.59	0.78	0.45	0.64	0.76	0.39	0.86	0.85	-

Table 11: Table showing different error statistics for artificial light simulation

Illuminance, Artificial light														
		Row 1			Row 2			Row 3						Avg. indoor
	Outdoor	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	V1	V2	hori. error
MAE	-	18.75	18.94	162.13	10.44	30.75	16.44	16.50	32.19	19.88	65.75	211.44	23.56	32.65
MBE (%)	-	-5.30	7.60	82.96	-3.42	10.51	-1.76	3.47	10.69	9.77	22.11	-53.32	-9.16	11.39
RMSE (%)	-	6.41	11.65	85.89	5.21	13.28	6.45	6.69	13.01	20.39	28.18	54.18	13.04	16.43
CV(RMSD)	-	8.60	6.93	113.17	4.25	11.24	5.83	7.38	11.98	7.14	39.19	75.33	17.65	-
R Square	-	1.00	1.00	0.99	1.00	1.00	0.99	1.00	1.00	0.99	0.99	0.95	1.00	-
Note: After removing H3, H10, avg. hori. MAE= 16.39, avg. hori. MBE = 3.16%, and avg. hori. RMSE = 8.31%														

5.3 Future works

It has been established that in order to increase the accuracy of virtual sensors, a comprehensive representation of actual sky luminance distribution is necessary. Also any possible mismatch, between actual building and model geometry, in terms of building the orientation, placement of sensors, and fenestration geometry, etc. needs to be minimized. Therefore, a more detailed study with the use of sky-scanner is recommended.

6 CONCLUSION

A simulation-based supervisory software application, named as *RealTimeRadiance*, was developed in Java to demonstrate the implementation of virtual illuminance sensors. As the name suggests, this application provides continuous supply of near real-time illuminance data using a calibrated *Radiance* model of building and sky condition. Reflectance of various surfaces in a test room were measured and used in the simulation model. The luminance distribution of the sky was modelled by Perez All Weather Sky model using measured global and horizontal irradiance data from the on-situ weather station. In order to improve the accuracy (error statistics) of the simulation results the model was calibrated until the illuminance results converged. Acquired knowledge about the best practices for model calibration, and optimization to reduce the computational time has been reported (refer 4) which will help the user in calibrating the model for use in *RealTimeRadiance*.

In order to test the accuracy of calibrated model, horizontal and vertical indoor illuminance, was measured at various locations in a test room in Vienna, at 15 minute interval, for a weekend. The measure illuminance data was then compared with simulated illuminance data. The simulated daylight illuminance data for the first day, which was a clear sunny sky day, were found to have "good accuracy"; for the second day, which was a partly (intermediate) cloudy sky day, had "moderately accuracy". The results for artificial lighting case had "good accuracy". The accuracy of simulated results was categorized into qualitative levels of accuracy, based on mean bias error, and root mean bias error statistics. The reasons for comparatively lower accuracy during daylight cases were mostly due to less accurate representation of sky luminance distribution in sky models. Care should be taken when the sky luminance distribution is obtained from a mathematical sky model, especially in case of buildings with North facing windows. Also care must be taken with global and diffuse horizontal irradiance measurement, especially on cloudy days. For accurate sky models, the use of sky-scanner is suggested. *RealTimeRadiance*, due to its scalable modular architecture, can include a module for sky-scanner.

It has been described above that the developed application, *RealTimeRadiance,* has the potential to automate the process of continuously running *Radiance* simulation with real-time monitored data as input to create dynamic boundary conditions. Depending upon the outside sky conditions, the current version of the application provides resulting illuminance with varying degree of reliability. Therefore, due care must be taken while using the results for cost critical building monitoring and control processes.

7 Bibliography

Arnold, K., Gosling, J., & Holmes, D. (2006). *The Java Programming Language, 4th Edition.* Addison-Wesley.

Ayani, R. (2003). *Modeling and Simulation.* Retrieved May 22, 2014, from ict.kth.se: http://www.ict.kth.se/courses/2G1126/vt03/Pmodule-vt03/L01.pdf

Baharuddin. (2013). The Accuracy of RADIANCE Software to Model Overcast Sky Condition. *International Seminar on Architecture in Urbanized Maritime Culture: Chances and Challenges in Design and Planning for Sustainable Future, and the 3rd CONVEEESH* (pp. 1-10). Makassar: Hasanuddin University, Makassar.

Blancol, M. M., Alarcón, P. D., López, M. T., & Lara, C. M. (2001). Computing the solar vector. *Solar Energy, 70*(5), 431-441.

Blewitt, A. (2009). *Modular Java: What Is It?* Retrieved May 15, 2014, from infoq.com: http://www.infoq.com/articles/modular-java-what-is-it/

Brunner, K. (2013). *Java solar positioning code.* Retrieved December 10, 2013, from github.com: https://github.com/KlausBrunner/solarpositioning

Compagnon, R. (1997). *RADIANCE: a simulation tool for daylighting systems.* Retrieved December 2, 2013, from radsite.lbl.gov: http://radsite.lbl.gov/radiance/refer/rc97tut.pdf

Cvetković, M., Lenard, J.-D., & Mudri, L. (2005). Simulation of Luminaires in Radiance: Verification Method. *BCI'05 - Balkan Conference on Informatics.* Ohrid, Macédoine.

Delaunay, J. (n.d.). *gendaylit*. Retrieved May 28, 2014, from manpages.ubuntu: http://manpages.ubuntu.com/manpages/lucid/man1/gendaylit.1.html#contenttoc5

Delaunay, J. J. (n.d.). *gendaylit manpage*. Retrieved October 2, 2014, from manpages.ubuntu.com: http://manpages.ubuntu.com/manpages/lucid/man1/gendaylit.1.html

Grobe, L. (2014). *Help analyzing/calibrating a Radiance model to match simulated and measured illuminance*. Retrieved May 14, 2014, from comments.gmane: http://comments.gmane.org/gmane.comp.graphics.radiance.general/4038

Guglielmetti, R., & Scheib, J. (2012). Challenges to integrated daylighting and electric lighting simulation methods in a whole-building energy simulation context. *SimBuild 2012, Fifth National Conference of IBPSA-USA* (pp. 585-594). Madison, Wisconsin: IBPSA-USA.

Hiscocks, P. D. (2011). *Measuring Reflectance.* Retrieved May 14, 2014, from ee.ryerson.ca: http://www.ee.ryerson.ca/~phiscock/astronomy/light-pollution/reflectance-apparatus.pdf

Jackson, R. (2012). *UML Class Diagram Relationships, Aggregation, Composition*. Retrieved May 14, 2014, from usna86-techbits.blogspot.in: http://usna86-techbits.blogspot.in/2012/11/uml-class-diagram-relationships.html

Jacobs, A. (2014). *ltpict.pl.* Retrieved May 22, 2014, from github: https://github.com/NREL/Radiance/blob/combined/src/util/ltpict.pl

JavaFX 2.2 API Docs. (n.d.). *JavaFX 2.2 API Documentation*. Retrieved May 20, 2014, from docs.oracle.com: http://docs.oracle.com/javafx/2/api/

Larman, C. (2004). *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development (3rd Edition).* Prentice Hall.

Mahdavi, A. (2001). Simulation-based control of building systems operation. *Building and Environment*, 789-796.

Mahdavi, A., Brahme, R., & Gupta, S. (2001). Self-aware buildings: a simulation-based approach. *BS-2001, Seventh International IBPSA Conference*, (pp. 1241-1248). Rio de Janeiro.

Mardaljevic, J. (1999). *Daylight Simulation: Validation, Sky Models and Daylight Coefficients.* De Montfort University Leicester, Institute of Energy and Sustainable Development. Leicester: De Montfort University Leicester.

Metsker, S. J., & Wake, W. C. (2006). *Design Patterns in Java.* Addison-Wesley Professional.

Mischler, G. (2004). *Lighting Design Glossary*. Retrieved June 02, 2014, from schorsch.com: http://www.schorsch.com/en/kbase/glossary/skies.html

Orenstein, D. (2000). *QuickStudy: Application Programming Interface (API)*. Retrieved June 2, 2014, from computerworld: http://money.howstuffworks.com/business-communications/how-to-leverage-an-api-for-conferencing1.htm

Paul, J. (2014). *Difference between Association, Composition and Aggregation in Java, UML and Object Oriented Programming*. Retrieved May 14, 2014, from javarevisited.blogspot.in: http://javarevisited.blogspot.in/2014/02/ifference-between-association-vs-composition-vs-aggregation.html

Perez, R., Seals, R., & Michalsky, J. (1993). All-weather model for sky luminance distribution - Preliminary configuration and validation. *Solar Energy, 50*(3), 235-245.

Reinhart, C. (2011). Daylight performance predictions. In J. L. Hensen, & R. Lamberts, *Building Performance Simulation for Design and Operation* (pp. 235-276). Routledge .

Tahmasebi, F., & Mahdavi, A. (2012). Monitoring-based optimization-assisted calibration of the thermal performance model of an office building. *1-ICAUD, 1st International Conference on Architecture & Urban Design* (pp. 1111-1116). Department of Architecture, EPOKA University.

Vernon, V. (n.d.). *Understanding UML Class Relationships*. Retrieved May 15, 2014, from vaughnvernon.co: https://vaughnvernon.co/?page_id=31

Ward, G. J. (1994). The RADIANCE Lighting Simulation and Rendering System. *SIGGRAPH conference*, (pp. 459-472).

Ward, G. L. (n.d.). *rad*. Retrieved May 28, 2014, from radsite: http://radsite.lbl.gov/radiance/man_html/rad.1.html

Ward, G. L., & Shakespeare, R. (1998). *Rendering with Radiance - The Art and Science of Lighting Visualization.* Morgan Kaufmann Publishers.

Ward, G. L., & Shakespeare, R. (1998). *Rendering With Radiance: The Art And Science Of Lighting Visualization.* Booksurge Llc.

Ward, G. L., Shakespeare, R., Mardaljevic, J., & Ehrlich, C. (1998). *Rendering with Radiance:A Practical Tool for Global Illumination.* Retrieved February 10, 2014, from radsite.lbl.gov: http://radsite.lbl.gov/radiance/refer/s98c33.pdf

Wikipedia. (n.d.). *Application Programming Interface*. Retrieved May 14, 2014, from Wikipedia: http://en.wikipedia.org/wiki/Application_programming_interface

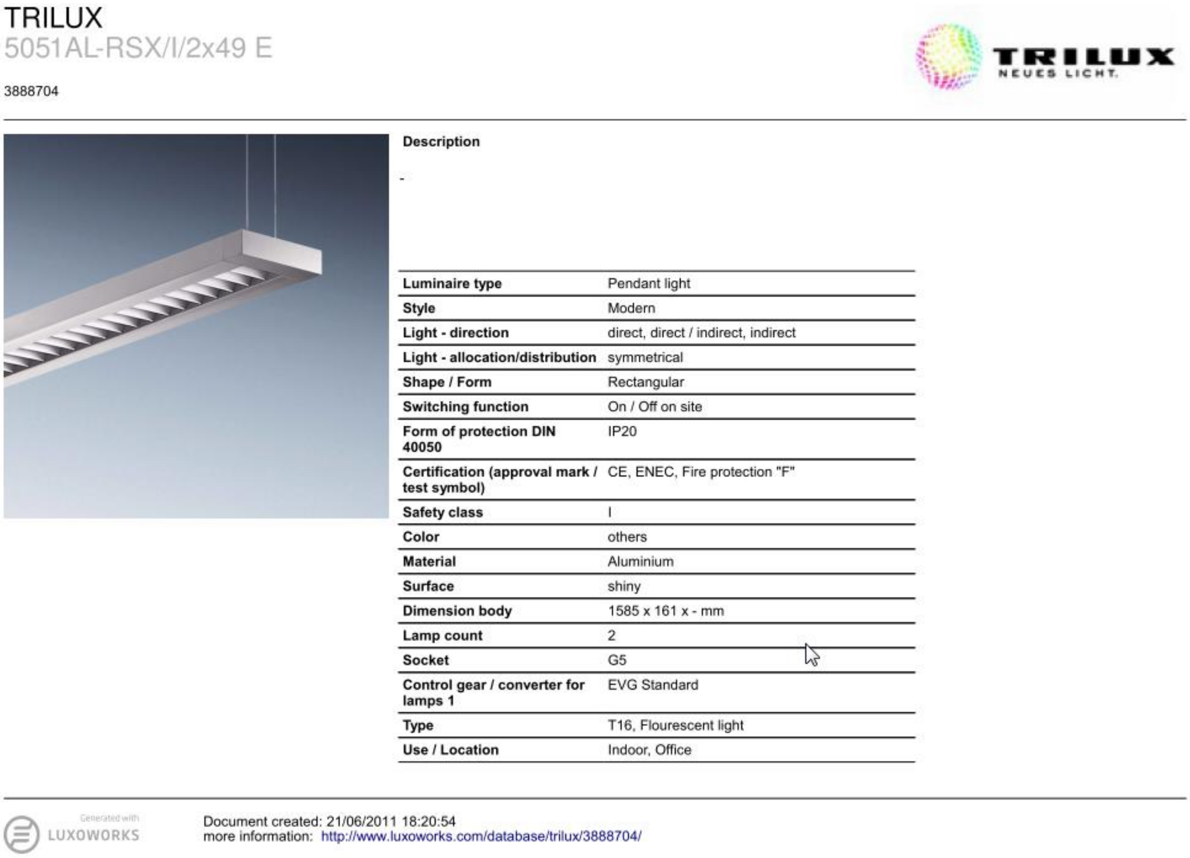
Zach, R., Glawischnig, S., & Mahdavi, A. (n.d.). *Incorporation of run-time simulation-powered virtual sensors in building monitoring systems.* Retrieved May 15, 2014, from lists.oasis-open.org: https://lists.oasis-open.org/archives/obix/201304/msg00004/MOST.pdf

Zach, R., Glawischnig, S., Hönisch, M., Appel, R., & Mahdavi, A. (2012). MOST An open-source, vendor and technology independent toolkit for building monitoring, data preprocessing, and visualization. *ECPPM 2012, 9th European Conference on Product and Process Modelling.* Reykjavik, Iceland.

Zhu, Y. (2010). *Lighting Calculations.* Retrieved April 12, 2014, from emt-india.net: http://www.emt-india.net/Presentations2010/33-34-Light_6-7_9-10Dec2010/33-Light_6-7Dec2010/Lighting_Calculations-YitingZhu.pdf

8 APPENDIX

8.1 Specification of the luminaire:



TRILUX

5051AL-RSX/I/2x49 E

3888704

Description

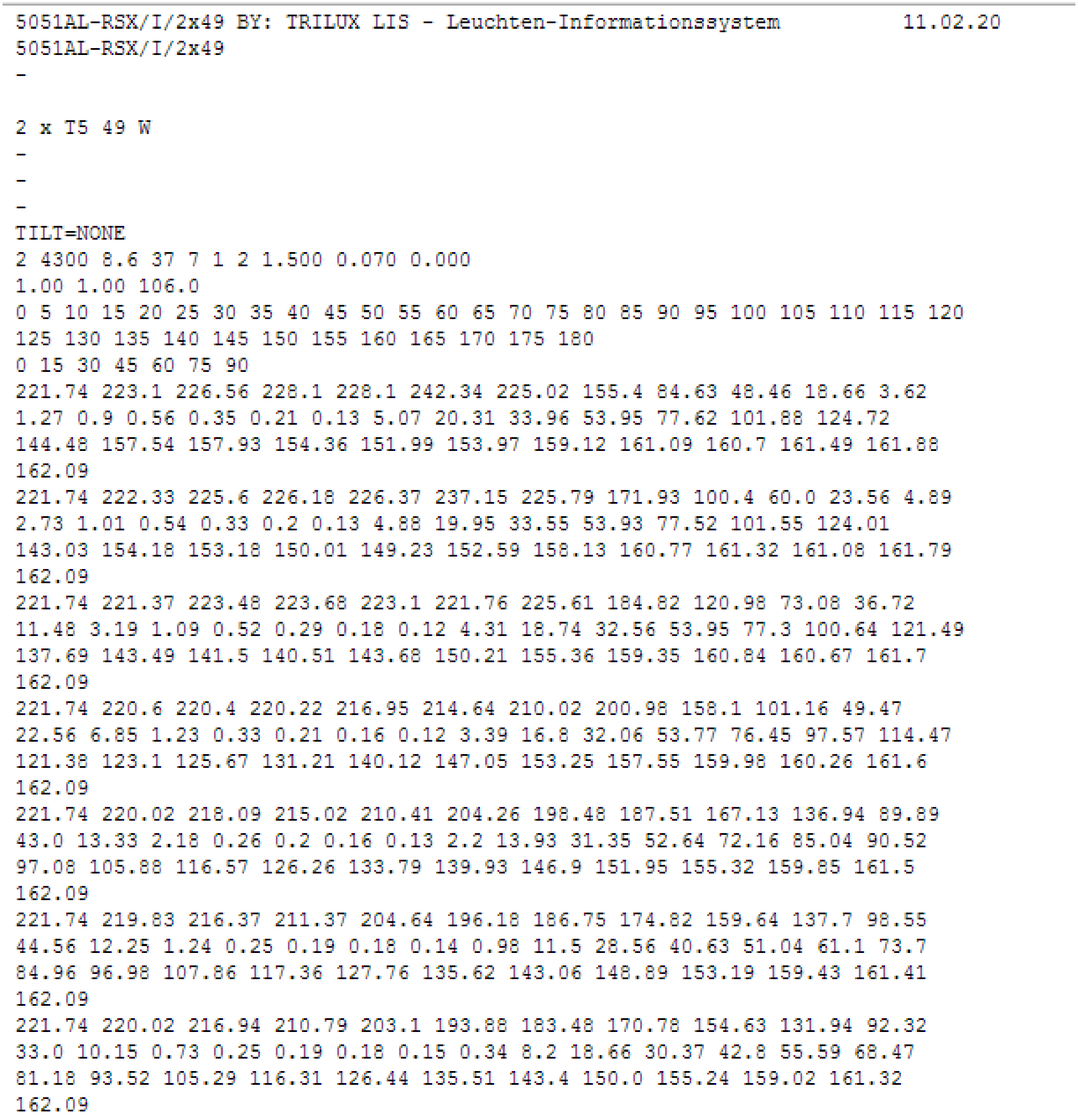
-

Luminaire type	Pendant light
Style	Modern
Light - direction	direct, direct / indirect, indirect
Light - allocation/distribution	symmetrical
Shape / Form	Rectangular
Switching function	On / Off on site
Form of protection DIN 40050	IP20
Certification (approval mark / test symbol)	CE, ENEC, Fire protection "F"
Safety class	I
Color	others
Material	Aluminium
Surface	shiny
Dimension body	1585 x 161 x - mm
Lamp count	2
Socket	G5
Control gear / converter for lamps 1	EVG Standard
Type	T16, Flourescent light
Use / Location	Indoor, Office

Generated with LUXOWORKS

Document created: 21/06/2011 18:20:54
more information: http://www.luxoworks.com/database/trilux/3888704/

8.2 IES file of the luminaire:

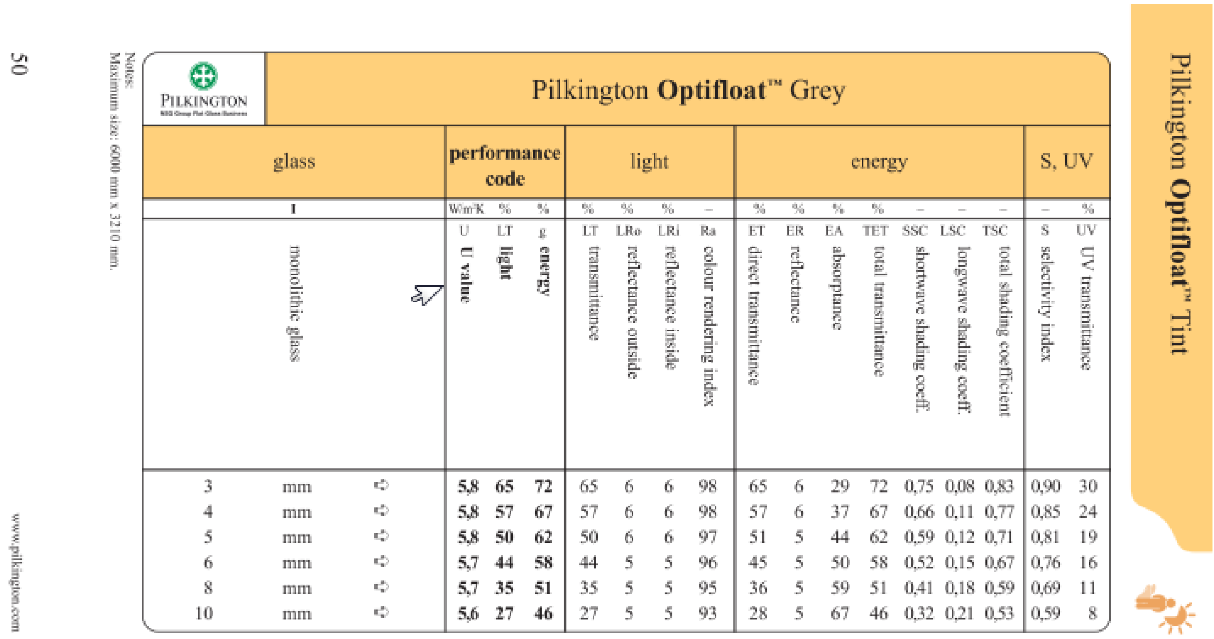


```
5051AL-RSX/I/2x49 BY: TRILUX LIS - Leuchten-Informationssystem          11.02.20
5051AL-RSX/I/2x49
-

2 x T5 49 W
-
-
-
TILT=NONE
2 4300 8.6 37 7 1 2 1.500 0.070 0.000
1.00 1.00 106.0
0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100 105 110 115 120
125 130 135 140 145 150 155 160 165 170 175 180
0 15 30 45 60 75 90
221.74 223.1 226.56 228.1 228.1 242.34 225.02 155.4 84.63 48.46 18.66 3.62
1.27 0.9 0.56 0.35 0.21 0.13 5.07 20.31 33.96 53.95 77.62 101.88 124.72
144.48 157.54 157.93 154.36 151.99 153.97 159.12 161.09 160.7 161.49 161.88
162.09
221.74 222.33 225.6 226.18 226.37 237.15 225.79 171.93 100.4 60.0 23.56 4.89
2.73 1.01 0.54 0.33 0.2 0.13 4.88 19.95 33.55 53.93 77.52 101.55 124.01
143.03 154.18 153.18 150.01 149.23 152.59 158.13 160.77 161.32 161.08 161.79
162.09
221.74 221.37 223.48 223.68 223.1 221.76 225.61 184.82 120.98 73.08 36.72
11.48 3.19 1.09 0.52 0.29 0.18 0.12 4.31 18.74 32.56 53.95 77.3 100.64 121.49
137.69 143.49 141.5 140.51 143.68 150.21 155.36 159.35 160.84 160.67 161.7
162.09
221.74 220.6 220.4 220.22 216.95 214.64 210.02 200.98 158.1 101.16 49.47
22.56 6.85 1.23 0.33 0.21 0.16 0.12 3.39 16.8 32.06 53.77 76.45 97.57 114.47
121.38 123.1 125.67 131.21 140.12 147.05 153.25 157.55 159.98 160.26 161.6
162.09
221.74 220.02 218.09 215.02 210.41 204.26 198.48 187.51 167.13 136.94 89.89
43.0 13.33 2.18 0.26 0.2 0.16 0.13 2.2 13.93 31.35 52.64 72.16 85.04 90.52
97.08 105.88 116.57 126.26 133.79 139.93 146.9 151.95 155.32 159.85 161.5
162.09
221.74 219.83 216.37 211.37 204.64 196.18 186.75 174.82 159.64 137.7 98.55
44.56 12.25 1.24 0.25 0.19 0.18 0.14 0.98 11.5 28.56 40.63 51.04 61.1 73.7
84.96 96.98 107.86 117.36 127.76 135.62 143.06 148.89 153.19 159.43 161.41
162.09
221.74 220.02 216.94 210.79 203.1 193.88 183.48 170.78 154.63 131.94 92.32
33.0 10.15 0.73 0.25 0.19 0.18 0.15 0.34 8.2 18.66 30.37 42.8 55.59 68.47
81.18 93.52 105.29 116.31 126.44 135.51 143.4 150.0 155.24 159.02 161.32
162.09
```

8.3 Sky model generated using gendaylit:

```
# xform -rz 22
# gendaylit 03 22 12:40:00 -a 48.1987 -o -16.3695 -m -15 -W 825.11 148.30
# Ground ambient level: 25.8

void light solar
0
0
3 6.722e+06 6.722e+06 6.722e+06

solar source sun
0
0
4 0.007067 -0.745442 0.666533 0.533000

void brightfunc skyfunc
2 skybright perezlum.cal
0
10 3.906e+01 2.455e+01 -0.971167 -0.323734 13.031139 -3.451024 0.253582 0.007067 -0.745442
0.666533

skyfunc glow sky_glow
0
0
4         1         1         1         0

sky_glow source sky
0
0
4         0         0         1         360
```

8.4 Specification of a translucent door:



50

Notes:
Maximum size: 6000 mm x 3210 mm.

PILKINGTON
NSG Group Flat Glass Business

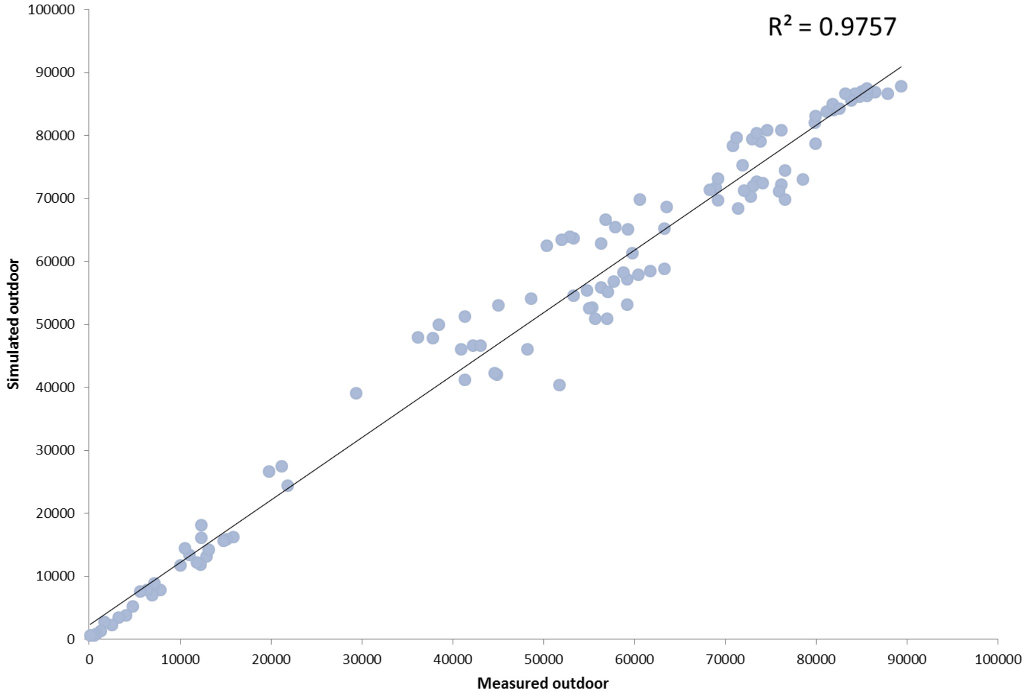
Pilkington **Optifloat**™ Grey

glass			performance code			light				energy							S, UV	
I			W/m²K	%	%	%	%	%	–	%	%	%	%	–	–	–	–	%
monolithic glass			U **U value**	LT **light**	g **energy**	LT transmittance	LRo reflectance outside	LRi reflectance inside	Ra colour rendering index	ET direct transmittance	ER reflectance	EA absorptance	TET total transmittance	SSC shortwave shading coeff.	LSC longwave shading coeff.	TSC total shading coefficient	S selectivity index	UV UV transmittance
3	mm	➪	**5,8**	**65**	**72**	65	6	6	98	65	6	29	72	0,75	0,08	0,83	0,90	30
4	mm	➪	**5,8**	**57**	**67**	57	6	6	98	57	6	37	67	0,66	0,11	0,77	0,85	24
5	mm	➪	**5,8**	**50**	**62**	50	6	6	97	51	5	44	62	0,59	0,12	0,71	0,81	19
6	mm	➪	**5,7**	**44**	**58**	44	5	5	96	45	5	50	58	0,52	0,15	0,67	0,76	16
8	mm	➪	**5,7**	**35**	**51**	35	5	5	95	36	5	59	51	0,41	0,18	0,59	0,69	11
10	mm	➪	**5,6**	**27**	**46**	27	5	5	93	28	5	67	46	0,32	0,21	0,53	0,59	8

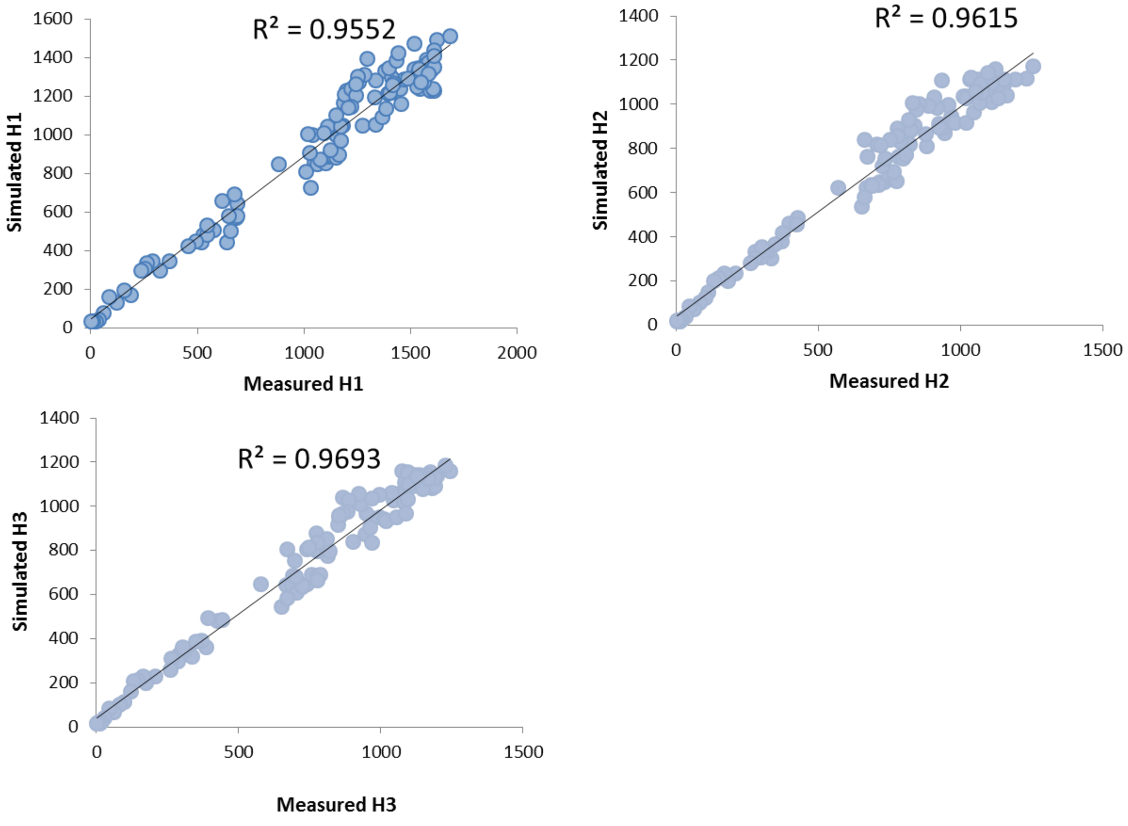
www.pilkington.com

Pilkington **Optifloat**™ Tint

8.5 Coefficient of determination (R^2) graphs for different scenarios:



Measured vs Simulated outdoor illuminance (22nd March, Daylight)

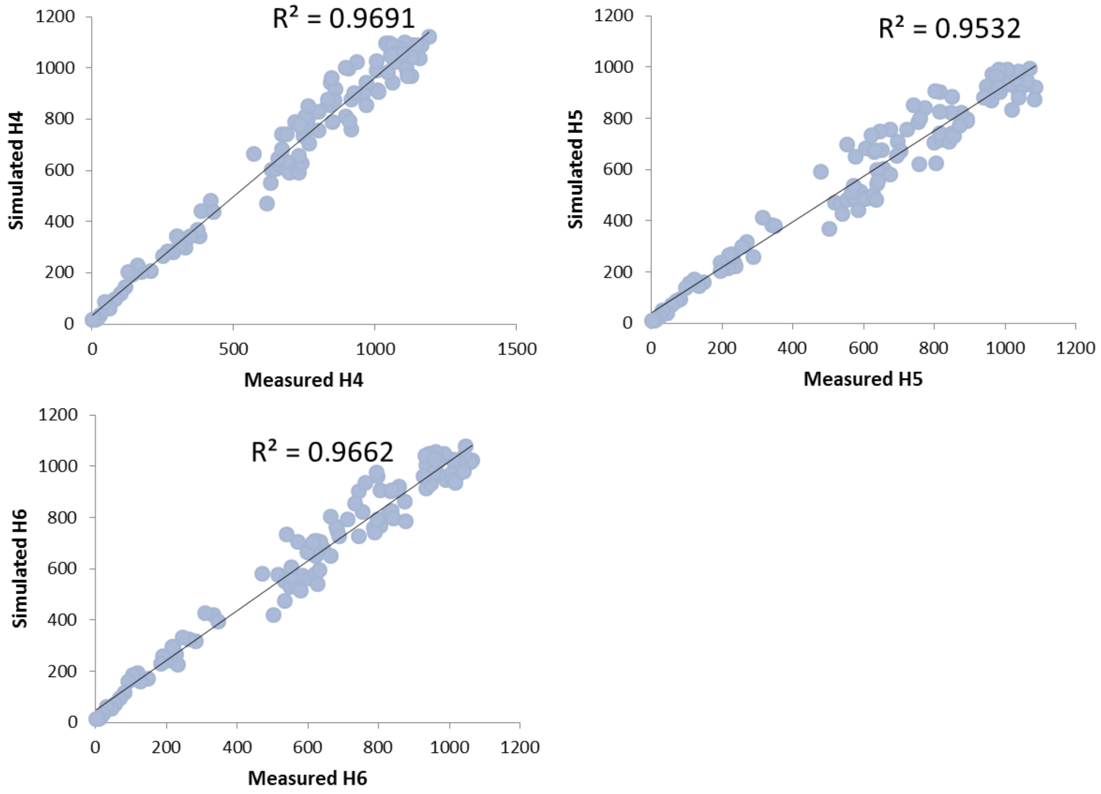


Measured vs Simulated indoor illuminance (22nd March, Daylight)

Sensors: H1, H2, H3

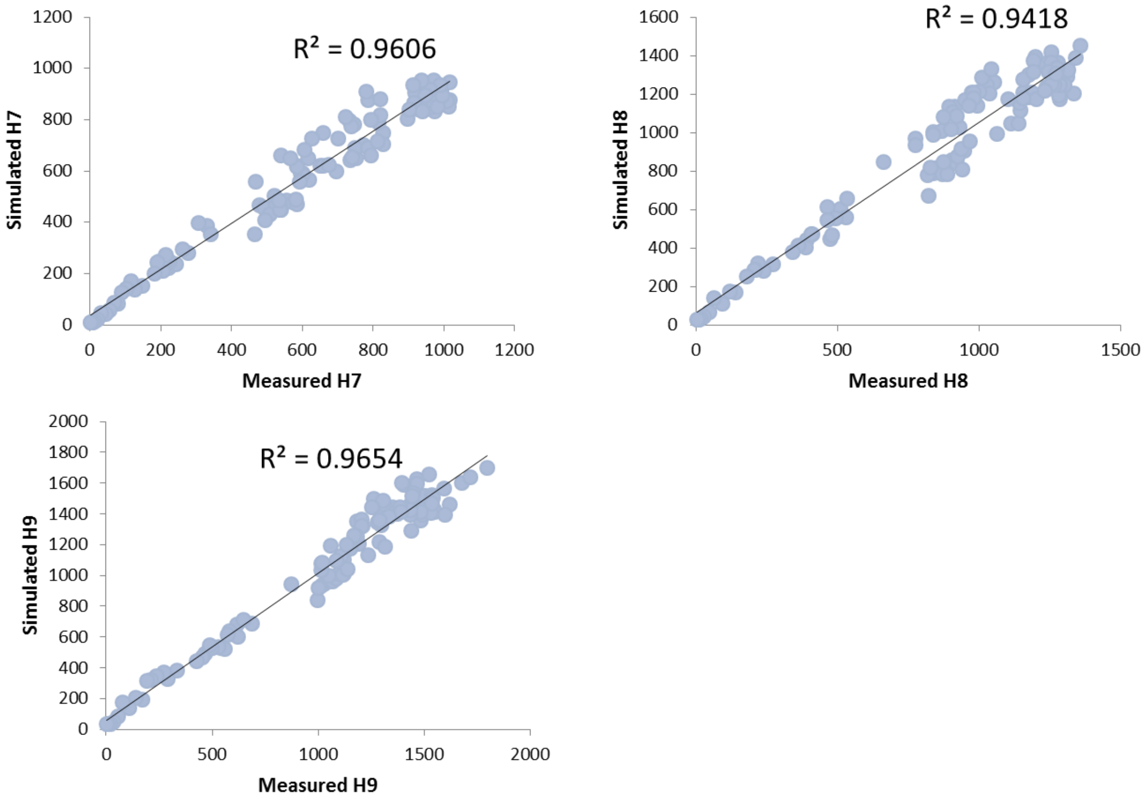
Measured vs Simulated indoor illuminance (22nd March, Daylight)

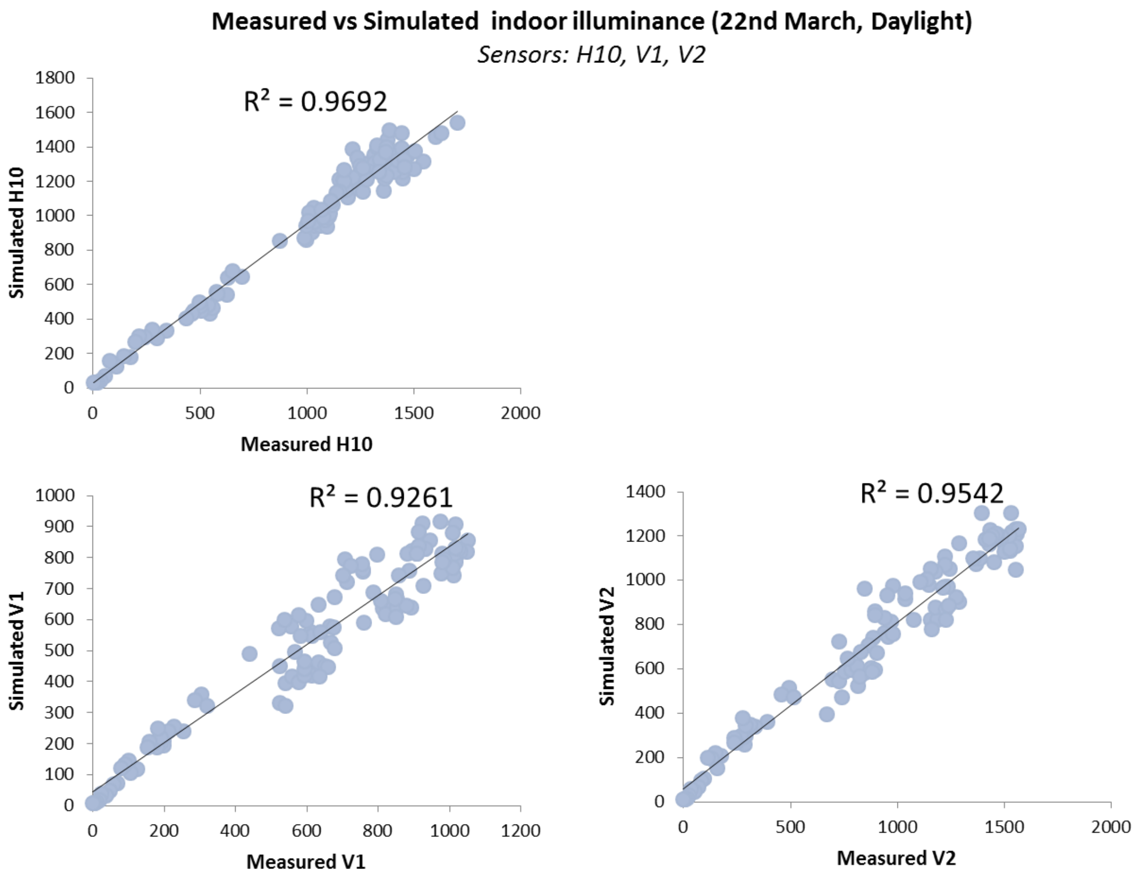
Sensors: H5, H6, H7

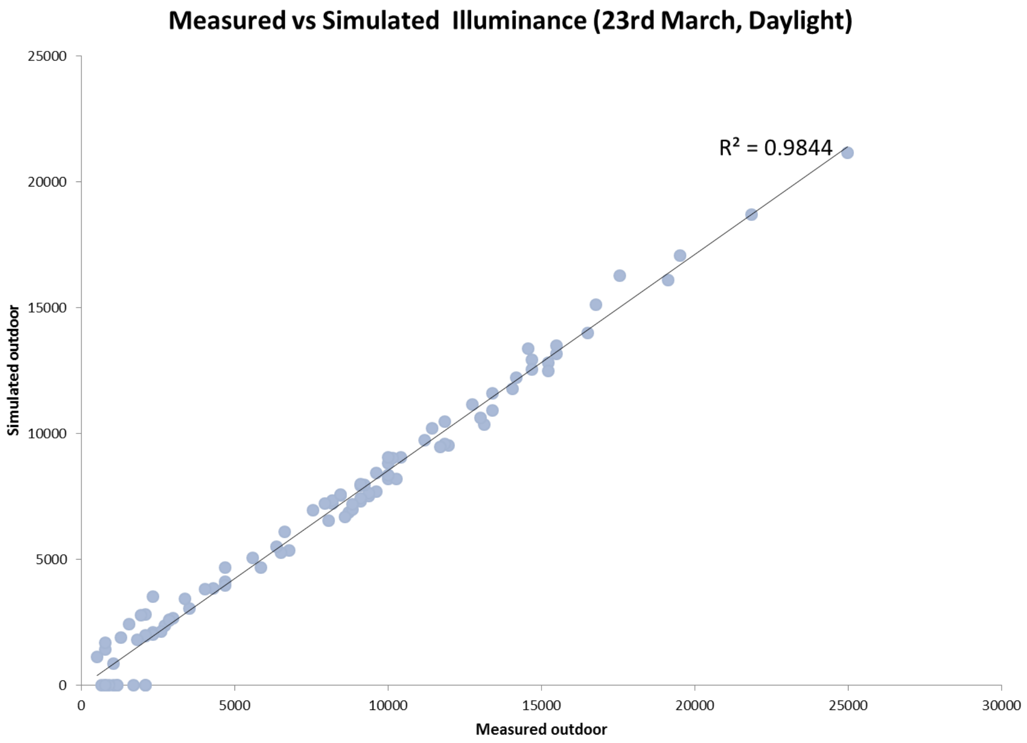


Measured vs Simulated indoor illuminance (22nd March, Daylight)

Sensors: H7, H8, H9

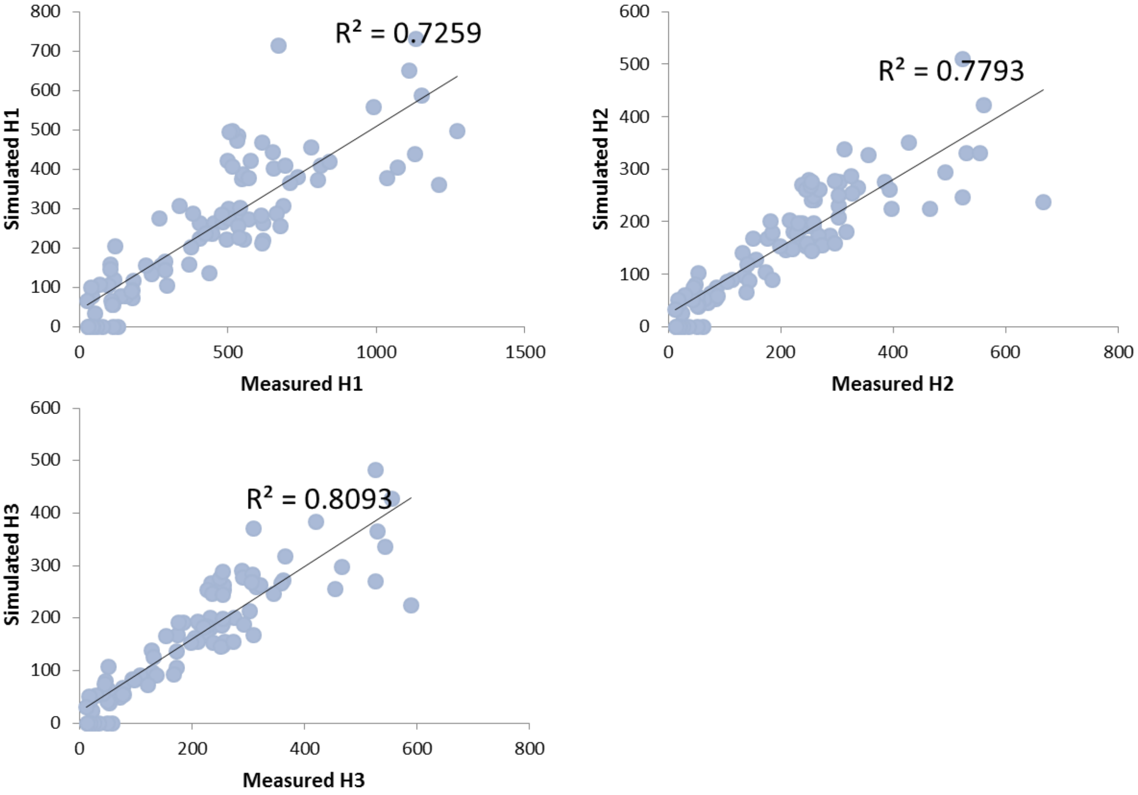



Measured vs Simulated indoor illuminance (22nd March, Daylight)
Sensors: H10, V1, V2
R² = 0.9692
Simulated H10
Measured H10
0
200
400
600
800
1000
1200
1400
1600
1800
0
500
1000
1500
2000
R² = 0.9261
Simulated V1
Measured V1
0
100
200
300
400
500
600
700
800
900
1000
0
200
400
600
800
1000
1200
R² = 0.9542
Simulated V2
Measured V2
0
200
400
600
800
1000
1200
1400
0
500
1000
1500
2000


Measured vs Simulated Illuminance (23rd March, Daylight)
R² = 0.9844
Simulated outdoor
Measured outdoor
0
5000
10000
15000
20000
25000
0
5000
10000
15000
20000
25000
30000

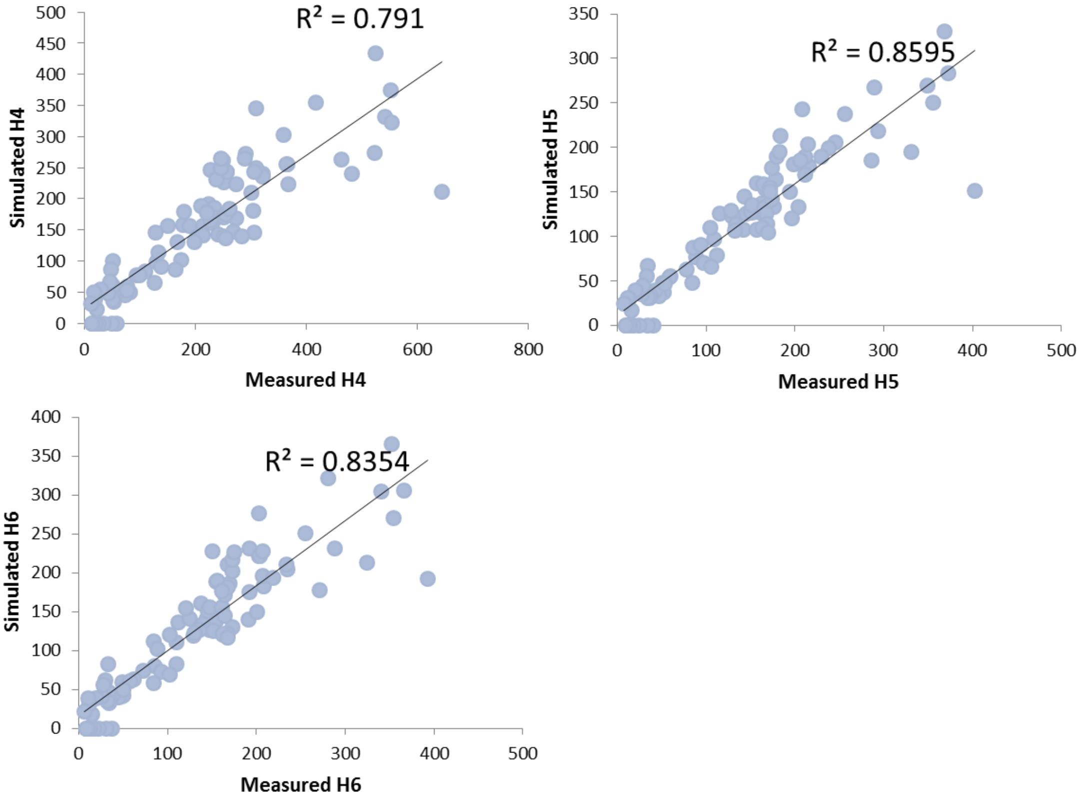
Measured vs Simulated indoor illuminance (23rd March, Daylight)

Sensors: H1, H2, H3



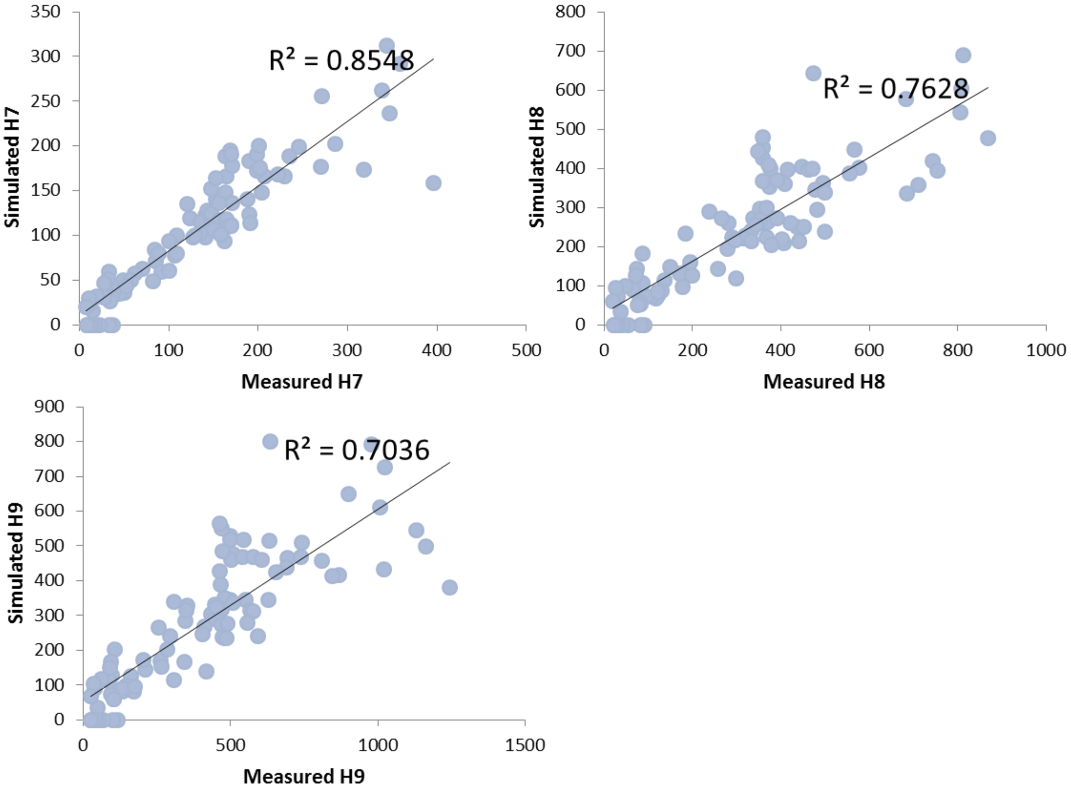
Measured vs Simulated indoor illuminance (23rd March, Daylight)

Sensors: H5, H6, H7



Measured vs Simulated indoor illuminance (23rd March, Daylight)

Sensors: H8, H9, H10


R² = 0.8548
Simulated H7
Measured H7
R² = 0.7628
Simulated H8
Measured H8
R² = 0.7036
Simulated H9
Measured H9