# Development of a microscale model for the thermal environment in complex areas

Inaugural-Dissertation
zur Erlangung der Doktorwürde
der Fakultät für Umwelt und Natürliche Ressourcen
der Albert-Ludwigs-Universität
Freiburg im Breisgau

vorgelegt von

**Dominik Fröhlich**

Freiburg im Breisgau
2016

| | |
|---|---|
| **Dekan** | Prof. Dr. Tim Freytag |
| **Referent** | Prof. Dr. Andreas Matzarakis |
| **Koreferent** | Prof. Dr. Rüdiger Glaser |
| **Disputationsdatum** | 30.01.2017 |

# Preface

The work at hand was written at the Chair of Environmental Meteorology, Faculty of Environment and Natural Resources of the Albert-Ludwigs-University Freiburg in the years 2013 to 2016.

The author wishes to thank Prof. Dr. Andreas Matzarakis, Research Center Human Biometeorology, German Meteorological Service (DWD), for supervising this Dissertation. In spite of own challenges he had to meet leaving the Chair of Meteorology and Climatology, University of Freiburg and taking office at the DWD, he continuously supported the work on this Dissertation by his kind and valuable advice and support. He also takes gratitude for giving me the opportunity to write this Dissertation.

I also want to express my deep gratitude to Prof. Dr. Rüdiger Glaser, head of the Chair of Physical Geography at the Albert-Ludwigs-University Freiburg, for assistance and becoming my second advisor.

The author wants to acknowledge the contribution of the Heinrich-Böll foundation to this dissertation project. As a fellow of the Heinrich-Böll foundation, the author not only received financial support by the foundations scholarship, but also profited from the experience and ideas provided by other fellows.

Furthermore, the author wants to thank Dr. Rainer Röckle of the iMA Richter & Röckle GmbH & Co. KG for the friendly help and guidance.

The author wants to thank his colleagues and friends of the Chair of Environmental Meteorology for the friendly environment and the encouragement in the progress of this dissertation project. This specifically holds for Ronja Vitt, Shi-Qi Yang, Spyridon Paparrizos and Yung Chang Chen, who were working on their PhD projects during the final phase of my work. It furthermore especially holds for the masters degree student Marcel Gangwisch who patiently went through many hours of pair programming and debugging in the course of implementing new functionality in the advanced SkyHelios model.

Not at least, the author thankfully wants to mention Ruth Kapanski, my parents Ingrid Fröhlich and Dr. Otmar Fröhlich, as well as my friends for leading me the way back to the world outside my office and for their patience and support.

# Contents

# 1 Summary

The increasing fraction of people living in urban areas is increasing the demand for the assessment of thermal conditions for humans within the urban environment. This can be achieved best by calculating thermal indices, representing the thermal perception and thermal stress of an average human being. To serve the needs of urban planning and architecture, thermal indices must be determined in high spatial and temporal resolution for rather large areas. Thermal indices are based on a number of meteorological and physiological parameters, that need to be provided in the desired spatial and temporal resolution.

The desired spatial resolution, as well as the spatial inhomogeneity of the urban environment makes it impossible to measure all the meteorological data required for the determination of thermal indices. The only feasible way of determining thermal indices in high resolution for urban areas of interest therefore is the calculation of all relevant parameters by a numerical model based on data from a reference station.

Besides air temperature ($T_a$), the most important meteorological input parameter for the calculation of thermal indices is the mean radiant temperature ($T_{mrt}$). It summarizes the effect of short- and longwave radiation. $T_{mrt}$ can not be measured, but needs to be calculated by complex radiation modelling taking all effects of the current environment into account. In urban areas with all types of different vertical structures, materials and surfaces $T_{mrt}$ is highly volatile in space and therefore needs to be calculated in high spatial resolution.

The advanced SkyHelios model now is capable of determining the mean radiant temperature spatially. This can be done based on astronomic calculations together with location, date and time, as well as cloud cover information, or based on an undisturbed global radiation provided as input parameter.

An other very important parameter in urban biometeorology is wind speed ($v$). This is because it shows great impact on humans for itself, but also because it influences other parameters that are important in urban biometeorlogy. Assessment of wind speed in complex urban environments requires spatially resolved data, as it shows strong spatial variability. This spatially resolved data can only be provided by numerical modelling.

A wind model suitable for the application in the field of urban biometeorology needs to meet quite some demands. One of them is that it should be rather fast, to allow for a sufficient number of data sets to achieve statistical significance. The model should also be able to cope with time independent

input data, as long data series often contain gaps and inconsistent time steps. Only a diagnostic wind model is considered to meet the requirements, as it is rather fast and fully time independent by design.
Currently only few operational diagnostic wind models exist. Most of them are designed for the use in particle dispersion modelling. Examples for models like this are TALdia or QUIC-URB. They are unfortunately not suitable for the integration in the SkyHelios model for technical reason or legal issues. A new wind model therefore had to be developed, improved and implemented in the SkyHelios model.
The diagnostic wind model implemented into the SkyHelios model is based on the approach and some of the parametrizations of the ABC model. Following this approach, first an initial wind field is set up that already must contain the different modifications to the undisturbed wind field caused by the obstacles in the model domain. This initial wind field contains a lot of divergence that is reduced by the multiplication by an Lagrangian multiplier. The Lagrangian multiplier has to be determined previously by solving a Poisson equation using the Successive-Over-Relaxation method.
During the construction of the initial wind field modifications have to be calculated for every obstacle. The model calculates four different types of modifications. A stagnation zone windward from the obstacle based on an improved parametrization, a recirculation on the lee-side, as well as a velocity deficit zone adjacent to the recirculation. If a street canyon is detected, a vortex is placed inside it using an advanced parametrization.
The calculations can be performed based on spatially determined roughness length($z_0$) and displacement height ($z_d$), that can be determined by the advanced SkyHelios model on the spatial basis of voronoi cells and three different approaches.
Spatially resolved $T_a$, v, and $T_{mrt}$ can be used to calculate the three commonly used thermal indices Perceived Temperature (PT), Universal Thermal Climate Index (UTCI) and Physiologically Equivalent Temperature (PET) spatially for any point within the model area.
The SkyHelios model was tested using two test domains. One of them is the place of the old synagogue, a popular place close to the city center of Freiburg, Southwest Germany. The second one is the "Institutes Quarter" north of the city center of Freiburg. The first test domain was also used for a comparison of the modelling results to on-site measurements.
Results show, that the wind model is able to estimate wind direction rather well while wind speed is underestimated by approximately factor 2 at a height of 1.5 m. Global radiation and $T_{mrt}$ are determined in quite good precision by the advanced SkyHelios model. Inaccuracy in thermal indices calculated by the advanced SkyHelios model mostly are arising from the underestimation in wind speed, as well as uncertainties regarding the position of the on-site stations.
All in all, an operational model was developed in the course of this dissertation project. It successfully calculates human thermal perception based on thermal indices for large complex urban areas in high resolution and rather short time.

# 2 Zusammenfassung

Durch den zunehmenden Anteil der urbanen Bevölkerung ist auch die Bedeutung der Bewertung human-biometeorologischer Parameter im städtischen Umfeld gestiegen. Eine Möglichkeit hierfür ist die Berechnung thermischer Indizes, die das thermische Empfinden und den thermischen Stress eines standardisierten Menschen repräsentieren sollen. Um den Anforderungen von Stadtplanern und Architekten gerecht zu werden, müssen thermische Indizes in städtischen Umgebungen in hoher räumlicher und zeitlicher Auflösung bestimmt werden. Hierzu müssen alle zur Berechnung der Indizes benötigten meteorologischen Parameter in der gewünschten Auflösung zur Verfügung gestellt werden.

Die gewünschte räumliche Auflösung sowie die räumliche Inhomogenität der städtischen Umgebung machen die Messung aller erforderlichen Parameter zur Berechnung thermischer Indizes unmöglich. Der einzige erfolgversprechende Weg zur Berechnung thermischer Indizes für städtische Umgebungen in hoher räumlicher Auflösung ist die Berechnung durch numerische Modelle auf Basis von Messwerten einer Referenzstation.

Neben der Lufttemperatur ($T_a$) ist die mittlere Strahlungstemperatur ($T_{mrt}$) der wichtigste meteorologische Eingangsparameter zur Berechnung thermischer Indizes. Sie fast den Einfluss aller kurz- und langwelligen Strahlungsflüsse zu einer Temperatur zusammen. $T_{mrt}$ kann nicht gemessen werden, sondern muss durch komplexe Strahlungsmodellierung unter Berücksichtigung der lokalen Umgebungseinflüsse bestimmt werden. In städtischen Umgebungen mit verschiedensten horizontalen und vertikalen Strukturen, Materialien und Oberflächen weist $T_{mrt}$ eine große räumliche Varianz auf und muss daher in möglichst hoher Auflösung bestimmt werden.

Das erweiterte SkyHelios Modell kann nun auch die mittlere Strahlungstemperatur bestimmen. Dies kann auf Grundlage astronomischer Berechnungen unter Einbeziehung von Ort, Datum und Zeit, sowie dem Bedeckungsgrad geschehen. Eine weitere Möglichkeit ist die Bestimmung von $T_{mrt}$ auf Basis der als Eingangsparameter zur Verfügung gestellten ungestörten Globalstrahlung einer Referenzstation.

Ein weiterer sehr bedeutender Parameter in der Human-Biometeorologie ist die Windgeschwindigkeit ($v$). Diese beeinflusst Menschen einerseits direkt, andererseits hat die Windgeschwindigkeit einen großen Einfluss auf viele andere Größen, die wiederum eine große Bedeutung für die Human-Biometeorologie haben. Die Bewertung der Windgeschwindigkeit im städtischen Raum setzt räumlich hoch aufgelöste Daten voraus, da $v$ räumlich sehr inhomogen ist. Diese können

ausschließlich durch die Anwendung von Modellen erlangt werden.
Ein Windfeldmodell für die Nutzung im Bereich der urbanen Human-Biometeorologie muss einige Anforderungen erfüllen. Zunächst sollte es in möglichst kurzer Zeit gültige Windfelder berechnen können um den Einsatz langer, und damit repräsentativer, sowie statistisch signifikanter Datenreihen zu ermöglichen. Des weiteren ist die Berechnung einzelner Zustände unabhängig von Zeitschritten von großer Wichtigkeit, da sie Berechnungen für einzelne Situationen und lange Datenreihen mit Lücken ermöglicht. Diesen Anforderungen kann nur ein diagnostisches Windfeldmodell gerecht werden, da es relativ wenig Rechenzeit benötigt und durch seine Arbeitsweise vollständig unabhängig von Zeitschritten ist.
Von diesem Modelltyp existieren derzeit jedoch nur wenige Implementierungen. Beispiele für diesen Modelltyp sind die Modelle TALdia und QUIC-URB. Diese sind jedoch leider aufgrund technischer Anforderungen oder Lizenzbedingungen nicht geeignet, um in das SkyHelios Modell integriert zu werden. Ein neues Windmodell musste daher entwickelt, angepasst und verbessert sowie in SkyHelios implementiert werden.
Das in SkyHelios integrierte diagnostische Windfeldmodell basiert auf dem Ansatz und einigen Parametrisierungen des ABC Modells. Diesem Ansatz folgend wird zunächst ein Ausgangswindfeld berechnet. Dieses muss bereits alle Modifikationen des ungestörten Profils durch die Strömungshindernisse innerhalb des Modellgebiets enthalten. Es enthält jedoch zunächst noch starke Divergenzen, die durch Multiplikation mit einem Lagrang'schen Multiplikationsfaktor minimiert werden. Dieser muss zuvor iterativ durch die Lösung einer Poisson-Gleichung bestimmt werden, wofür das Successive Over-Relaxation Verfahren eingesetzt wird.
Zur Erstellung des Ausgangswindfelds müssen für alle Strömungshindernisse vier verschiedene Modifikationen des Windfelds berechnet werden. Eine luvseitige Stagnationszone entsprechend einer verbesserten Parametrisierung, ein Rücklauf im Lee des Hindernisses, ein sich daran anschießender Bereich mit geringerer Windgeschwindigkeit, sowie ein horizontaler Vortex in einer Straßenschlucht, falls das Modellgebiet eine solche umfasst.
Die Windfeldberechnungen können auf Grundlage lokaler Werte für die Rauigkeitslänge ($z_0$) und die Verdrängungshöhe ($z_d$) durchgeführt werden. Diese kann das Modell SkyHelios auf räumlicher Bases von Voronoi-Zellen unter Anwendung drei verschiedener Ansätze berechnen.
Die räumliche Verteilung von $T_a$, v und $T_{mrt}$ kann anschließend zur räumlichen Berechnung der drei gängigen thermischen Indizes Gefühlte Temperatur (PT), Universal Thermal Climate Index (UTCI) und Physiologisch Äquivalente Temperatur (PET) für jeden Punkt innerhalb des Modellgebiets genutzt werden.
Das Modell wurde mit zwei Modellgebieten getestet. Eines der Modellgebiete ist der Platz der alten Synagoge, einem beliebten Platz in der Nähe der Stadtmitte von Freiburg im Südwesten Deutschlands. Das andere Modellgebiet ist das "Institutsviertel" nördlich der Innenstadt Freiburgs. Das erste Modellgebiet wurde zusätzlich für einen Vergleich mit Messungen vor Ort genutzt.
Die Ergebnisse bescheinigen dem Windmodell eine ausreichende Abschätzung der Windrichtung.

Die Windgeschwindigkeit in einer Höhe von 1.5 m wird dagegen stark (um circa Faktor 2) unterschätzt. Globalstrahlung, sowie $T_{mrt}$ werden durch das verbesserte SkyHelios Modell mit guter Genauigkeit berechnet. Ungenauigkeiten in der Bestimmung von thermischen Indizes durch das verbesserte SkyHelios Modell rühren vornehmlich von der Unterschätzung der Windgeschwindigkeit, sowie von Unsicherheiten der Positionen der Stationen vor Ort her.

Die Ergebnisse zeigen die Funktionsfähigkeit des erweiterten SkyHelios Modells. Dieses ist nun in der Lage in kurzer Zeit das thermische Empfinden von Menschen auf Grundlage thermischer Indizes für große, komplexe, urbane Modellgebiete in hoher Auflösung zu berechnen.

# 3 Introduction

Humans are influenced by climate conditions not only during their free time (e.g. Endler 2010), but also during their daily live indoors (e.g. Höppe 1993b) as well as outdoors (e.g. Höppe 1999).
In the year 2011 more than half the worlds population lived in cities (Population Division 2012). This fraction is growing and is expected to continue growing in the future (Population Division 2012). Health and well-being of the urban population, thus, is already an important issue in urban planning (e.g. Helbig et al. 1999, Matzarakis et al. 2008), but will be of increasing importance in the future. Numerous studies have been carried out in the last years that show strong correlation between health, as well as mortality on the one side and urban biometeorology on the other side. Especially heat stress during the summer months seems to lead to an increase in mortality (e.g. Koppe et al. 2004, Conti et al. 2005, Muthers et al. 2010, Matzarakis et al. 2011, Nastos and Matzarakis 2012).
Cities in general show slightly different diurnal variability in air temperature compared to their surroundings (e.g. Oke 1995, Helbig et al. 1999). This is mostly due to modifications in the radiation budget by ground sealing, different surface materials and many vertical surfaces (Oke 1995, p. 276ff). Most materials typically used in cities convert a huge fraction of the incident short wave radiation to longwave radiation (compare to fig. 7.13 in Oke 1995, p. 255), thus warm their surrounding (Oke 1995, p. 276ff). Additionally many of them have high heat storage capacities (Oke 1995, p. 284). This leads to less cooling at night time. The two effects significantly increase air temperature in cities compared to rural areas. The phenomena is called the Urban Heat Island (UHI, Oke 1995, p. 288ff).
Another increase in urban temperatures is caused by climate change. E.g. for Freiburg (south-west Germany), an increase of days with heat stress by up to 5 % is expected (Matzarakis and Endler 2010).
To improve health and well-being of the inhabitants (e.g. Tromp 1980, p. 90ff), as well as to increase the city's attractivity to tourism, some cities (e.g. Freiburg) already are showing some effort to improve thermal urban bioclimate (Matzarakis et al. 2008). This can be modified by urban planning through changes in building configuration (e.g. Lin et al. 2010a, Hwang et al. 2011, Herrmann and Matzarakis 2012), surface materials (e.g. Lin et al. 2010b) and urban green (e.g. Shashua-Bar et al. 2011, Charalampopoulos et al. 2015). To take the necessary steps city planners and decision makers need information that puts a lot of parameters together forming an easy to understand way to present information (Matzarakis et al. 2008). This information can be best provided through maps

(Matzarakis and Mayer 1992, Matzarakis 2001).
The most common way to show information about thermal biometeorology is the calculation of thermal indices like the Predicted Mean Vote (PMV, Fanger 1972) Physiologically Equivalent Temperature (PET Höppe 1993a; 1999), the Perceived Temperature (PT, Staiger et al. 2012) and the Universal Thermal Climate Index (UTCI, Jendritzky et al. 2012) combining several aspects to approximate the thermal perception of a sample human being (e.g. Fröhlich and Matzarakis 2013, Lopes et al. 2011, Lin et al. 2010a, Hwang et al. 2011, Lin et al. 2013, Charalampopoulos et al. 2015). Thermal indices are taking into account many parameters (e.g. Fanger 1972, Höppe 1999, Staiger et al. 2012). They can be divided in meteorological and physiological parameters (Höppe 1993a). The physiological parameters mainly consist of information about the physiology of the human body, as well as e.g. posture and sex, that are required for most of the thermal indices (e.g. for PMV, PET, and PT Fanger 1972, Höppe 1999, Staiger et al. 2012). Examples for the meteorological parameters are air temperature, vapour pressure, wind speed and the different radiation fluxes (e.g. Fanger 1972, Höppe 1999, Jendritzky et al. 2012, Staiger et al. 2012).
Some meteorological parameters are rather stable and do not vary a lot in space. This mainly holds for the parameters air temperature and vapour pressure. Within urban areas, the radiation fluxes are strongly influenced, among others, by shape, surface and position of surrounding obstacles. They therefore can show strong variability within rather small distances. Another important meteorological parameter for the calculation of thermal indices, that is strongly influenced by the urban morphology, is wind speed (VDI 2008).
As some meteorological input parameters show a lot of spatial variation in cities, also the results for the indices themselves show strong variations in space. To consider these variations thermal indices need to be calculated spatially in order to provide results that are more meaningful. To be able to run the calculations for thermal indices spatially, spatial information about all the necessary input parameters, especially the radiation fluxes and wind speed, is required (Matzarakis et al. 2009).
This information can not be gathered by using climate station records. As some of the parameters show strong spatial variation, numerous stations would be required, that would need to be placed all over the area of interest. Such a grid of stations would not only be very expensive, but also impossible to set up in a city, as the stations would stand in the way of the inhabitants. Therefore spatial information for within urban areas can only be provided by modelling. Thereby physical modelling using wind tunnel experiments, or numerical modelling can be done. As wind tunnel experiments are expensive, complex and time consuming, only numerical models are suitable for productive use in urban biometeorology.
Currently, the most common models used in urban biometeorology are either limited to an individual point of interest (e.g. RayMan, Matzarakis et al. 2007; 2010), or take a lot of time to calculate rather short model periods (e.g. ENVI-met, Bruse 1999). This leads to rather short timespans being modelled, that are statistically unsafe. In order to calculate spatial information for long periods of

several years, all required parameters have to be calculated in a very short time (Matzarakis and Matuschek 2011).

# 4 State of the Art

Atmospheric processes and the involved parameters are hard to determine for complex environments (Matzarakis et al. 2010). One of the most complex kind of environment is the street level of built-up areas (Hwang et al. 2011). The volume ranging from the ground to the roof-level, the urban canopy layer (Oke 1995, p. 274) is, on the other hand, the sphere where information is needed at most. Measurements only can deliver punctual insights, as the urban canopy layer is highly heterogeneous. However, for most purposes spatial distribution of many parameters are required. It is not only difficult, but also expensive to set up lots of measuring stations and interpolate their readings over a desired area. The application of models is, thus, the more promising approach for urban environments (Hwang et al. 2011).

## 4.1 Urban Climate and Modelling

Most meteorological parameters are modified a lot by cities (Matzarakis et al. 2009, Matzarakis and Endler 2010, Matzarakis et al. 2010). While variations in air temperature ($T_a$) and air humidity (RH) within small distances mostly do not exceed the range of accuracy of measurements, wind speed and direction as well as the different radiation fluxes vary strongly in urban environments.
The radiation fluxes, in the field of urban biometeorology often summarized as the mean radiant temperature ($T_{mrt}$, see section 4.3.5.4), show very strong variation in very short distances. This is due to the vertical extension of the urban structures causing shading and reflections (e.g. Hwang et al. 2011, Emmanuel and Johansson 2006) as well as the different surface materials reflecting, absorbing, storing and emitting radiation (e.g. Salata et al. 2015). $T_{mrt}$ can not be measured. It is always determined by equations. There are different techniques available, but all of them bear uncertainties (Chen et al. 2014). The most common way of determining $T_{mrt}$ therefore is through numerical modelling.
Other two of the parameters that are modified a lot by the urban environment, are the wind speed and the wind direction. The wind field in and over a build-up area with a high density of obstacles with large vertical extension is disturbed up to a hight of approximately 500 m, while there may be an undisturbed wind field in a hight of 300 m over the surroundings of a city already (Kuttler 2000, p. 427).

Determination of wind speed and direction within the urban canopy layer faces many challenges. Due to heterogeneity and the limited possibilities of taking measurements, there are only insufficient observations of the full wind field and turbulence in urban areas (Roth 2000).
For most parameters, there are two different ways of modelling: physical or numerical modelling (e.g. Matzarakis 2001, Grimmond et al. 2010).

## 4.2 Physical Models

Physical models are reproductions of the complex urban environment in simplified way in a small scale (e.g. Helbig et al. 1999). As scaling sometimes breaks similarity requirements (Schatzmann et al. 1987) physical models can only be applied in some cases. Typical applications are shading models or wind tunnel experiments.
Physical wind models are mostly experiments in a wind tunnel. The model area therefore has to be rebuild as a true to scale model that is placed inside a wind tunnel. For use in urban meteorology, the wind tunnel has to be configured to provide a vertical profile that compares well to the actual one (e.g. Macdonald et al. 1998, Matzarakis 2001). Also all the requirements of similitude (geometric, kinematic, and dynamic similitude) must be met to be able to compare results from small-scale wind tunnel experiments to prototype scale (Schatzmann et al. 1987).

Wind tunnel experiments are most commonly used for the analysis of air pollutant dispersion (e.g. Schatzmann et al. 1987, König-Langlo and Schatzmann 1991, Kastner-Klein et al. 2001), but can also provide useful information for thermal urban biometeorology (Matzarakis 2001). Another important application of wind tunnel experiments is the validation of numerical models (e.g. Schlünzen et al. 2003).
Physical modelling, however, can be very challenging e.g. for low wind speed, or varying vertical profiles (Gross et al. 1994).

## 4.3 Numerical Modelling Techniques

Numerical models are the reproduction of all relevant parts of the actual environment by physical equations and parametrizations. As both, measurements and physical models are expensive and not always applicable, numerical models are used regularly. This especially holds for studies that require meteorological information for within urban areas.
The most simple, but least precise "model" is the assumption, that some station (mostly rather remote, influenced by different urban structures, and recording at a wrong level) provides data,

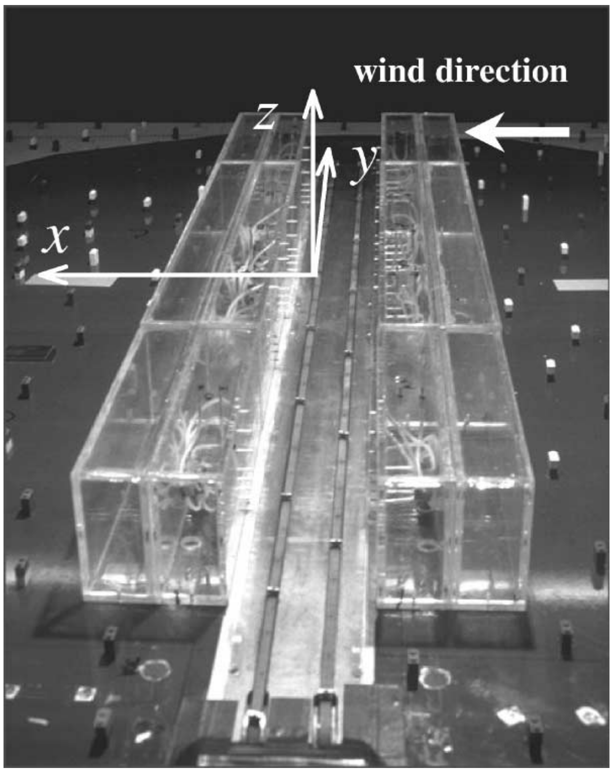


**fig. 4.1:** Example for a physical wind model: Model of a street canyon in the atmospheric boundary layer wind tunnel, Karlsruhe (Kastner-Klein et al. 2001).

that is representative also for the point or area of interest (e.g. wind speed ($v$, in m/s) and wind direction($WD$, in °): eq. 4.1).

$$v_{station} = v_{site} \quad \text{and} \quad WD_{station} = WD_{site} \tag{4.1}$$

### 4.3.1 Numerical Wind Modeling

Sticking to the example of wind modelling, this simple assumption can be improved a lot by using a vertical profile (see sections 4.3.1.1 to 4.3.1.1). To achieve better results, more sophisticated models are required. The two basic types of numerical models are explained in the following two sections (sections 4.3.3 and 4.3.4).

#### 4.3.1.1 Vertical Wind Profile

Neglecting altitude-correction of wind speed is commonly used, but leads to inaccuracy when it comes to thermal comfort assessment. A little better assumption, that can at least be considered valid for spatially averaged conditions, is the application of an appropriate vertical profile. This can be used, to altitude-correct wind measurements (compare to figs. 4.2a and 4.2b). Vertical profiles are also commonly used to generate initial data for more sophisticated wind models.

**Logarithmic Profile** The best known vertical profile to approximate the average wind speed ($\bar{u}$, in m/s) in a desired height ($z$, in m), that is also applied the most, is the logarithmic wind profile (eq. 4.2, compare to e.g. Oke 1995, Helbig et al. 1999).

$$\bar{u}(z) = \frac{u_*}{\kappa} ln\left(\frac{z-d}{z_0}\right) \tag{4.2}$$

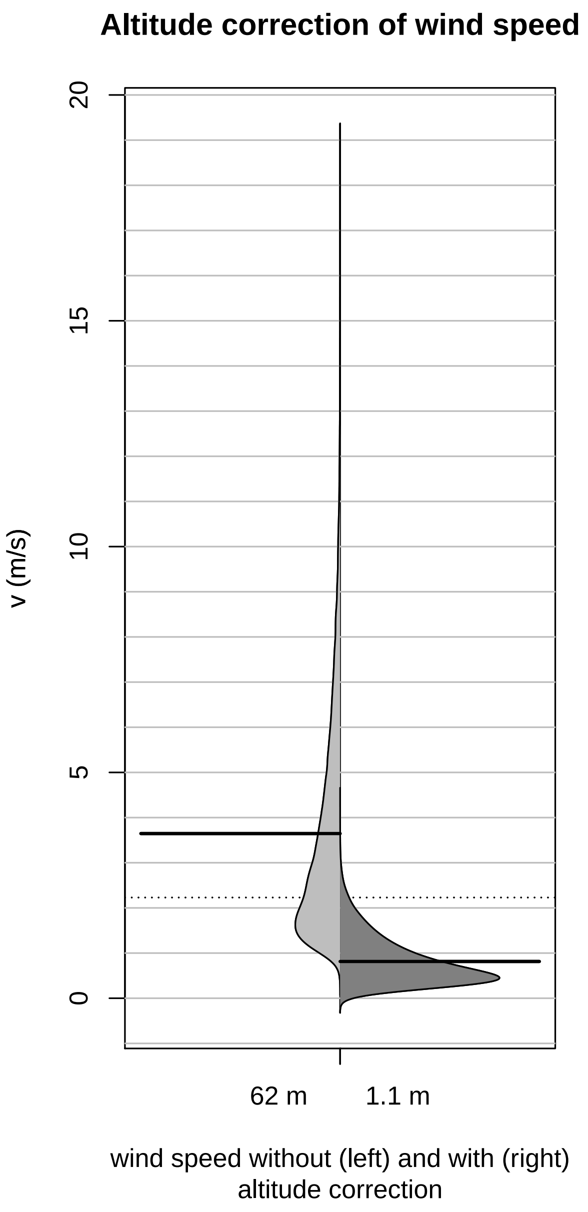
In eq. 4.2 $u_*$ is the shear velocity in m/s, $\kappa$ the Von Kármán constant, $z$ the target height above ground (in m), $d$ the zero plane displacement (in m), and $z_0$ the surface roughness length (in m). However, by definition, the logarithmic profile is only valid above the displacement height. It therefore can hardly be considered valid within urban areas.

**Power-Law Profile** While it provides a way better estimation than just using station readings, the logarithmic profile will fail to calculate a wind speed for any height smaller than the zero plane displacement hight. Therefore the power-law profile (e.g. Kuttler 2000, p. 428) is the better approach for within urban areas. It calculates a power-law index profile based on a reference mean wind speed $\bar{u}_{ref}$ in m/s at a reference height $z_{ref}$ in m using eq. 4.3.

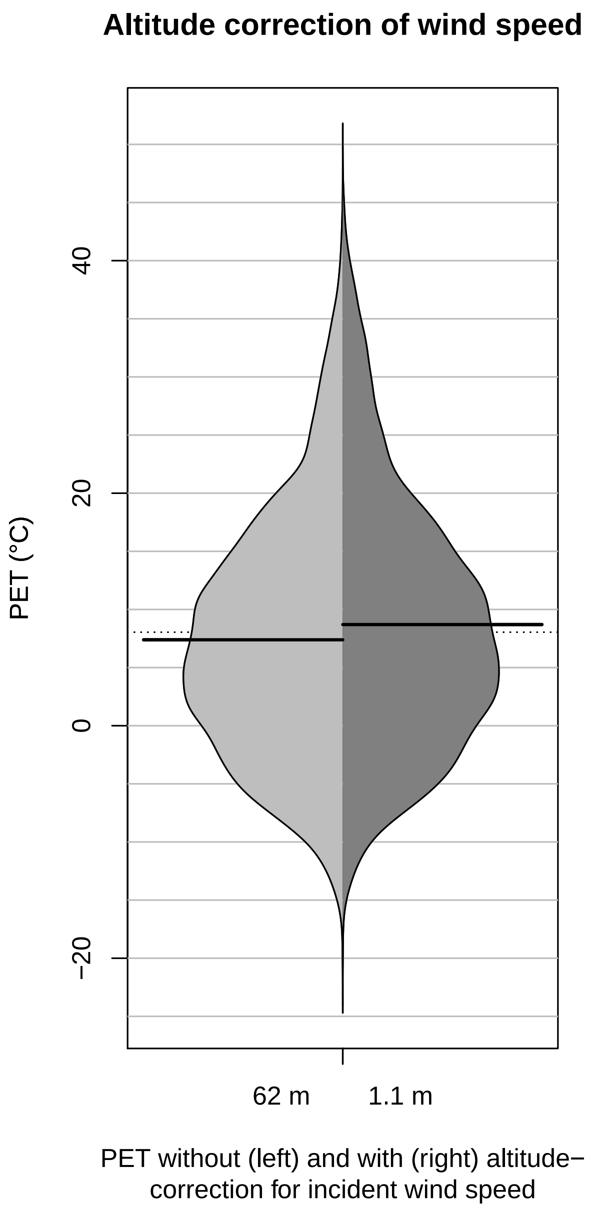
$$\bar{u}(z) = \bar{u}_{ref}\left(\frac{z}{z_{ref}}\right)^a \tag{4.3}$$

As the equation for the attenuation coefficient $a$ is not stated in Kuttler (2000), it was taken from Matzarakis et al. (2009). According to Matzarakis et al. (2009) $a$ is calculated by eq. 4.4.

$$a = 0.12 \cdot z_0 + 0.18 \tag{4.4}$$



**(a)** Distribution of wind speed.



**(b)** Distribution of PET.

**fig. 4.2:** Effect of altitude correction using a logarithmic wind profile on wind speed measurements and PET (see section 4.3.6.3) based on the same dataset. Meteorological data covering the period 1999-01-01 to 2010-12-31 in hourly resolution recorded by the urban climate station Freiburg. Sensor height for wind speed is 62 m. The target height is the average height of the human gravity center of 1.1 m above ground as required for the calculation of thermal indices (see section 4.3.6 Matzarakis et al. 2009).

This leads to a stable profile for any value of $z$. However $d$ is neglected in this approach, what is considered a strong limitation.

**Urban Canopy Profile** A vertical wind profile that is capable of estimating the wind speed for any height $z$ in meters above or below the displacement height is the urban canopy profile introduced by Macdonald (2000). It is basically a combination of both, a modified logarithmic profile (eq. 4.2) and a modified power-law profile (eq. 4.5, compare to eqs. (4.3) and (4.4)). The first one is applied above the displacement height, while the latter is used to derive wind speed below $d$. The methodology was first introduced by Cionco (1965; 1972) trying for the consideration of vegetation in the vertical profile.

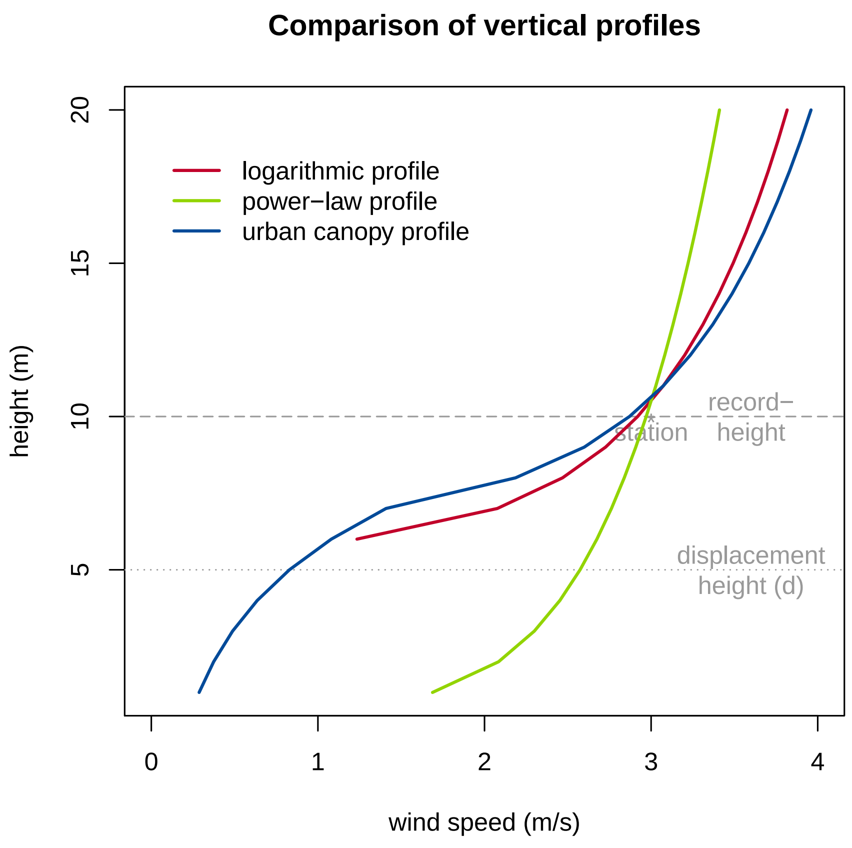


**fig. 4.3:** Comparison of the logarithmic profile (red), the power-law profile (green) and the urban canopy profile (blue) for an incident wind speed of 3.0 m/s recorded in 10 m. All profiles consider $z_0$ of 0.1 m, but only the logarithmic profile and the urban canopy profile take the displacement height of 5 m into account. The logarithmic profile is unable to calculate wind speed for any height $z \leq d$. The vertical resolution for all three profiles is 1 m.

$$\bar{u}(z) = \bar{u}_{ref}^{\left(a \frac{z}{z_{ref}} - 1\right)} \tag{4.5}$$

The urban canopy profile provides a pretty good assumption of the mean wind speed in urban areas (Macdonald 2000). However, for many studies spatially averaged conditions are insufficient. If wind speed is required for a certain point within urban structures, a prognostic (section 4.3.3) or a diagnostic wind model (section 4.3.4) needs to be applied.

### 4.3.2 Estimation of roughness length and displacement height

Air moving over a rough surface is influenced by the surface roughness. This especially holds for urban areas, that strongly modify the wind field through their high aerodynamic roughness (Landsberg 1981). The most common parameters describing surface roughness in the context of urban climate are the zero-plane displacement height ($z_d$), the roughness length ($z_0$) and the dimensionless drag coefficient $C_{D(z)}$ (Lettau 1969, Counihan 1975, Wieringa 1993). Their calculation is usually based on the frontal area density $\lambda_f$ and the building plan area density $\lambda_p$ (both dimensionless) (Bottema 1997, Bottema and Mestayer 1998, Grimmond and Oke 1999). Alternative ways to describe roughness are the effective height $h_{eff}$ (in m) (Matzarakis and Mayer 1992), the porosity of the urban canopy layer $Por$ (dimensionless) or the urban directionality (Compagnon and Raydan 2000, Ratti et al. 2006).
For most vertical wind profiles (refer to section 4.3.1.1) the roughness length ($z_0$) is required. More sophisticated vertical profiles (e.g. the urban canopy profile after Macdonald 2000) also require the displacement height ($z_d$). $z_0$ (and $z_d$ if considered by the approach) shows strong influence on the vertical profile of wind speed (e.g. for the urban canopy profile, see fig. 6.1). Local $z_0$ and $z_d$ should therefore be considered in all studies in the field of urban biometeorology (Ketterer et al. 2016).
As a simplification for applied studies Davenport et al. (2000) introduced the "Davenport classes". There are eight classes for different land use types providing statistical values for roughness. Only two of the eight classes are representing settlements and urban areas. Those can not be distinguished any further, what must be considered insufficient for the application in the field of urban biometeorology.
Statistical values for different land-use types are available in the literature (e.g. Helbig et al. 1999), but are mostly derived from wind-tunnel experiments with homogeneous arrays of obstacles (Grimmond and Oke 1999). As the urban morphology is usually very diverse, statistical values abbreviated from wind-tunnel experiments with regular arrays are found to be inaccurate (Kanda et al. 2013).
As measurements within urban areas are expensive and error prone (Grimmond et al. 1998), the calculation of local roughness parameters based on the actual urban morphology seems to be more promising. From the many approaches available, the three approaches by Lettau (1969), Matzarakis and Mayer (1992), and Bottema and Mestayer (1998) will be presented in detail in this study, as they were implemented into the SkyHelios model (section 4.4.2, Ketterer et al. 2016). All of them need (apart from other input data) a reference area for each obstacle. Most studies are

using rather arbitrary reference areas (Ratti et al. 2006), e.g. lot areas (Gal et al. 2009) or regular grids (Matzarakis and Mayer 1992).

#### 4.3.2.1 Voronoi diagram

To overcome this shortcoming, Ketterer et al. (2016) are proposing the calculation of a voronoi diagram to obtain more distinct reference areas. A voronoi diagram is the separation of an area into voronoi cells. Those are defined as all points $p$, that are closer to an obstacle, than to any other (Ottmann and Widmayer 2012).
The calculation of the voronoi diagram can be done based on an approach after Fortune (1987), using a sweepline to run over all the obstacles determining the voronoi cells corners (Fortune 1987).

#### 4.3.2.2 Morphometric approach after Lettau (1969)

Lettau (1969) introduced an approach based on observations during wind-tunnel experiments with irregular arrays of homogeneous obstacles. He proposes the calculation of the aerodynamic roughness length $z_0$ depending on the average height $\bar{h}$ in m, the frontal area $A_{frontal}$ in $m^2$ and the reference area $A_{total}$ in $m^2$, eq. 4.6

$$z_0 = 0.5 \cdot \lambda_f \cdot \bar{h} \tag{4.6}$$

with $\lambda_f$ according to eq. 4.7.

$$\lambda_f = A_{frontal} \cdot A_{total}^{-1} \tag{4.7}$$

As this approach is based on the frontal area, $z_0$ after Lettau (1969) is dependent on the wind direction.

#### 4.3.2.3 Effective heights after Matzarakis and Mayer (1992)

Matzarakis and Mayer (1992) introduced the "effective heights" $h_{eff}$ (m) that are defined as the sum of the averaged and weighted heights of buildings, vegetation and other objects. The averaged heights of building, vegetation and other surfaces within each reference area are weighted by the area occupied by them (eq. 4.8).

$$h_{eff} = \lambda_{p,B} \cdot \bar{h}_B + \lambda_{p,V} \cdot \bar{h}_V + \lambda_{p,S} \cdot \bar{h}_S \tag{4.8}$$

| | | |
|---|---|---|
| $\lambda_{p,B}$: | plan area density for buildings | (dimensionless) |
| $\bar{h}_B$: | average building height | (m) |
| $\lambda_{p,V}$: | $\lambda_p$ for vegetation | (dimensionless) |
| $\bar{h}_V$: | average vegetation height | (m) |
| $\lambda_{p,S}$: | $\lambda_p$ for all other surfaces inside reference area | (dimensionless) |
| $\bar{h}_S$: | average height of the other surfaces | (m) |

The approach after Matzarakis and Mayer (1992) does not calculated $z_0$ directly. However, $z_0$ can be abbreviated from the $h_{eff}$ assuming the relationship proposed by Kondo and Yamazawa (1986) according to eq. 4.9

$$z_0 = 0.25 \cdot h_{eff} \tag{4.9}$$

The approach is only based on morphology and is, thus, independent from incident wind direction.

#### 4.3.2.4 Approach after Bottema (1997) and Bottema and Mestayer (1998)

Bottema (1997) published an approach based on the dimensions of the luv-side vortex zone and the lee-side recirculation zone of an obstacle. As both are usually unavailable, the roughness length $z_0$ is calculated depending on the volumetric average of the obstacles height $h_v$ in m, the frontal area density $\lambda_f$ and the plan area density $\lambda_p$ (eq. 4.10, Bottema and Mestayer 1998).

$$z_0 = (h_{v,i} - z_d)exp(-\frac{\kappa}{\sqrt{0.5 \cdot C_{Dh} \cdot \lambda_f}}) \tag{4.10}$$

Eq. 4.10 applies the dimensionless von-Karman constant (0.4, $\kappa$), the isolated obstacle drag coefficient $C_{Dh}$, and the frontal area density $\lambda_f$. The zero-plane displacement height ($z_d$) is calculated by the volumetric averaged obstacle height $h_i$ in m and the planar area density $\lambda_p$. In eq. 4.10, $C_{Dh}$ is considered constant (Bottema and Mestayer 1998).

$$z_d = h_i \cdot \lambda_p^{0.6} \tag{4.11}$$

Considering not only buildings, but also trees and forests, local $z_d$ is calculated based on porosity $Por$ (eq. 4.11). The zero-plane displacement height for forests is calculated by eq. 4.12 (Brutsaert 1975).

$$z_{d,forest} = 0.66 \cdot h_{forest} \tag{4.12}$$

$z_d$ for trees with a base area $A_{tree}$ in $m^2$ can be estimated by eq. 4.13 (Grimmond and Oke 1999).

$$z_d = h_i \cdot \left( \frac{(1 - Por_{tree})A_{tree}}{A_{total}} \right)^{0.6} \tag{4.13}$$

The drag coefficient $C_{Dh}$ for trees and forests is reduced by Porosity (eq. 4.14).

$$z_0 = (h_{v,i} - z_d)exp(-\frac{\kappa}{\sqrt{0.5 \cdot C_{Dh} \cdot (1 - Por) \cdot \lambda_f}}) \tag{4.14}$$

Porosity can be defined individually for any type of trees and forests but stays constant within the whole study area (e.g. 0.99 for Picea abies forests, 0.2 for mixed forests and 0.3 for trees).

The dimensionless frontal area density $\lambda_f$ is the ratio of the sum of the individual frontal areas $A_{frontal}$ in $m^2$ of each obstacle (i) divided by the total reference area $A_{total}$ in $m^2$. $\lambda_f$ can be calculated by eq. 4.15.

$$\lambda_f = \frac{\sum_{i=0}^{n} A_{frontal,i}}{A_{total}} \tag{4.15}$$

The frontal area is dependent on the incident wind direction. Overlapping frontal areas are only considered once.

The dimensionless planar area density $\lambda_p$ is the ratio of the sum of individual planar areas (projected roof area) $A_{planar}$ in $m^2$ of each obstacle $i$ compared to obstacle $i$'s whole reference area $A_{total}$ in $m^2$ (eq. 4.16).

$$\lambda_p = \frac{\sum_{i=0}^{n} A_{planar,i}}{A_{total}} \tag{4.16}$$

The volumetric height $h_{v,i}$ (m) is the sum of the product of each the obstacles volume $V_i$ in $m^3$ and height $h_i$ in m divided by the sum of the volume of all the individual obstacles in $m^3$ (eq. 4.17).

$$h_{\mathrm{v,i}} = \frac{\sum_{i=0}^{n} V_i \cdot h_i}{\sum_{i=0}^{n} V_i} \tag{4.17}$$

### 4.3.3 Prognostic models

Two different types of approaches are commonly used in numerical micro-scale wind modelling: "Prognostic", "dynamic", or "predictive" models, and "diagnostic" or "kinematical" models (Ratto et al. 1994). The first kind of models, the prognostic models, will be introduced in this section.
Prognostic wind field models solve time dependent hydrodynamic or thermodynamic equations (Ratto et al. 1994). Mostly the advection terms, equations to consider radiation, as well as terms for the calculation of turbulent momentum, heat and moisture fluxes are integrated in prognostic small-scale models. Models used for urban biometeorology (e.g. Bruse 1999) therefore mostly assume the air of the lower part of the atmosphere to be incompressible (Boussinesq approximation,

Boussinesq 1897). This leads to the precondition that the flow field has to be free of divergence at any time. Examples for prognostic models are MISKAM (Mikroskaliges Klima- und Ausbreitungs-Modell, Eichhorn 1989, Eichhorn and Kniffka 2010), the ENVI-met model (Bruse 1999), or the MUKLIMO_3 model (Sievers 1995; 2012).
Prognostic models can deliver valuable information for urban areas, as they are able to calculate spatial wind speed and direction quite accurately. However they have some limitations that make them unsuitable for the use with an urban biometeorological model. First, they tend to become unstable in too complex obstacle settings. This is major a problem, as prognostic models calculate from a given point in time to the future, along a certain time step (that can be fix or variable). Thus, they can not be restarted using the latest valid setting, and have to be re-run for the whole period. Another disadvantage comes along with the time step that has to be very small in order to achieve a valid result in a complex environment. This makes running prognostic models very expensive and generally leads to few calculated cases with rather short modelled timespans (Ratto et al. 1994). However, prognostic models can be run with very limited computational effort, if the model domain is very small, or the grid spacing is very large. So models with very large grid spacing were already used for questions of air pollutant control long time ago (e.g. Gross et al. 1987).

### 4.3.4 Diagnostic models

Diagnostic models follow a different approach. They basically deploy statistical information gathered from physical models and observations (e.g. from Hunt et al. 1978, Hosker 1985). In contrast to prognostic models, they first apply empirical parametrization (compare to fig. 4.4) to calculate a modified initial wind field (see section 4.3.4.1). Afterwards, divergence has to be minimized to receive a valid wind field (Röckle 1990). Therefore only conservation of mass has to be considered (Singh et al. 2008). The main advantage of this approach is that the differential equations to solve are reduced to one, as only conservation of mass has to be achieved iteratively. As iterations are the most time-consuming part in running wind solvers, a diagnostic wind model may be far faster than a prognostic one. The conservation of momentum is replaced by the modifications contained by the initial wind field. If the initial wind field was a proper guess, the result will be a very good estimation of the actual wind field.
Another advantage of diagnostic models is that they, in contrast to prognostic models (refer to section 4.3.3), calculate a whole new wind field for each timestep instead of calculating the velocity field for the next out of the current timestep's wind field. Time independent calculation of wind fields with variable timestepping is therefore possible (Röckle 1990). This may significantly reduce computational effort, as in many cases results are required for only some certain hours a day. If, e.g., results for 2 p.m. are needed for every day of a week, this would be seven wind fields to calculate

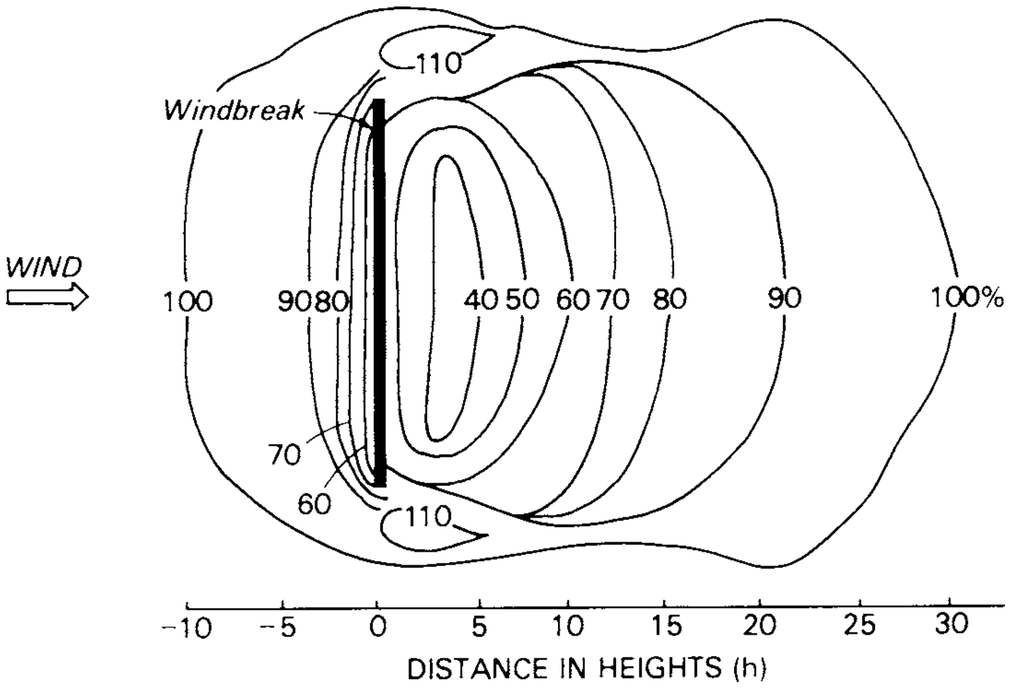


**fig. 4.4:** Wind speed modification by an obstacle perpendicular to the incident wind direction (Oke 1995, p. 245). Wind speed is denoted as percentages of incident wind speed.

for a diagnostic model, but, depending on the timestep, several hundreds for a prognostic wind solver.

#### 4.3.4.1 Initial wind field

The initial wind field of a diagnostic model is way more complex than the one for a prognostic model. While it is sufficient for a prognostic model to provide wind speed, direction, and atmospheric stability, a diagnostic model needs a full-featured flow field. This flow field must already contain all the calculated deformations to the wind field caused by obstacles or terrain (see fig. 4.7). The calculation of the initial wind field therefore is the most complex part of the diagnostic model (Röckle 1990). In example a diagnostic model is not able to create dynamic flow effects, so they must already be part of the initial wind field (Röckle 1990).
According to Röckle (1990) there are two kinds of parameters that mainly influence urban wind fields. One group of parameters are the meteorological parameters like the surrounding wind field with its specific wind speed and direction, its vertical profile, and turbulence. The other group, the aerodynamic parameters, mainly comprise influence of obstacles and terrain that guide and modify air currents (compare to fig. 4.5, Hunt et al. 1978).

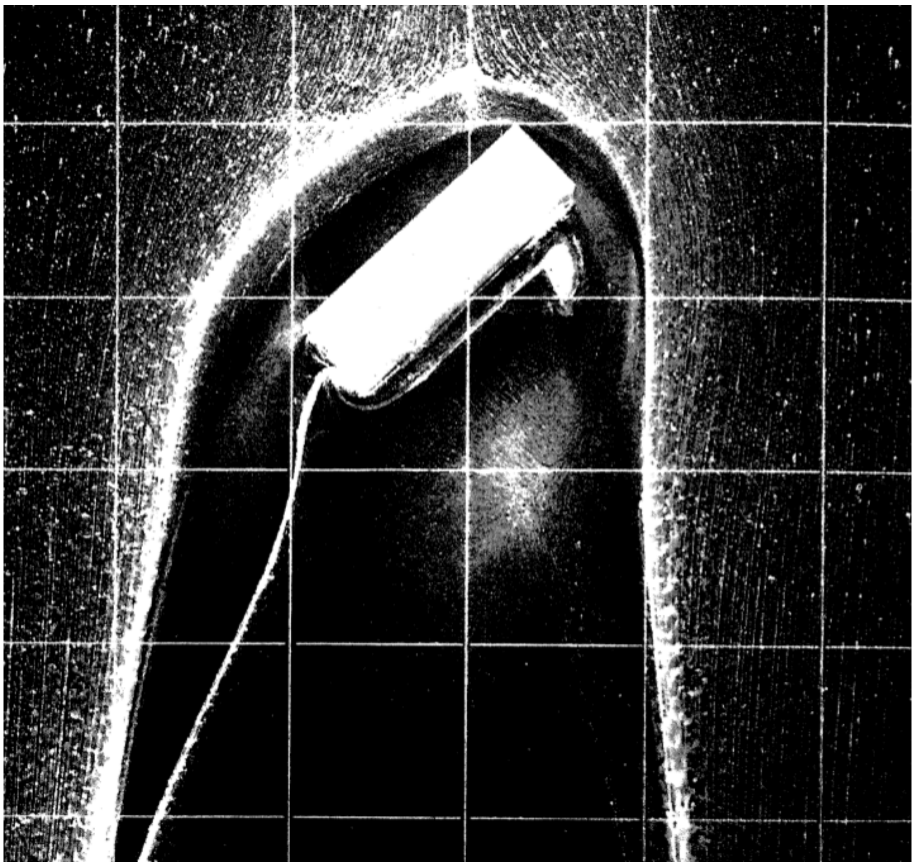


**fig. 4.5:** Wind flow around an obstacle visualized by the distribution of deposited zine oxide powder. Modified after Hunt et al. (1978).

#### 4.3.4.2 Divergence

The second step in operating a diagnostic wind model is to remove, or minimize the divergence in the flow field, as air must not vanish or be created moving from one cell to another. Assuming the air inside the model to be incompressible, this can be done by simple comparison of the in- and outflow into and out of a specific cell. To avoid divergence, the sum of all inflow and the sum of all outflow must be equal.
To achieve this, most diagnostic models follow the approach after Sasaki (1958; 1970a;b) applying variational analysis. As this means iterating over the whole model grid for several times, this is the main time-consuming part in running a diagnostic model. Some numeric procedures to speed up this process will be described below.

**Gauss-Seidel method** The Gauss-Seidel method is a rather classic relaxation method, following the approach that an initial distribution will converge towards an equilibrium solution along with the number of iteration steps (Press et al. 2007, p. 1060). In difference to other relaxation methods, not only the results from the last iteration step are used, but also the already available results for previously calculated cells (most commonly the cells with lower values of x, y, or z). An example for a one dimensional distribution is given in eq. 4.18, where $\rho$ is an exemplary source term.

$$x_{i,j}^{new} = \frac{1}{4}\left(x_{i+1,j}^{old} + x_{i-1,j}^{new} + x_{i,j+1}^{old} + x_{i,j-1}^{new}\right) - \frac{\triangle^2}{4}\rho_{i,j} \tag{4.18}$$

As this method is rather slowly converging, it is only of theoretical interest by itself. However, it is the basis of most of the faster converging methods (e.g. the sucessive over-relaxation method, compare to section 4.3.4.2). It also is important for multigrid methods (section 4.3.4.2 Press et al. 2007, p. 1060).

**Successive over-relaxation** Faster convergence than provided by the Gauss-Seidel method is achieved, if an over-correction is done at every cell and iteration step, as described in Press et al. (2007, p. 1061 ff). This is achieved by including an over-relaxation parameter $\omega$. The Successive Over-Relaxation method (SOR) will thereby stay converging for $\omega$ between zero and two. However, for an $\omega$ of less than one, the convergence will be slower than by using the standard Gauss-Seidel method (under-relaxation). $\omega$ of one will result in the standard Gauss-Seidel method. Thus, an $\omega$ between one and two is to be used to achieve faster convergence.
As most problems are impossible to solve analytically, the optimal $\omega$ that causes fastest convergence can not be calculated. However, a value that is very close to the optimal $\omega$ has to be used, for the method will not be much faster than the classic Gauss-Seidel method otherwise. To make an optimal guess, the actual numeric problem is projected to a known problem of the same size and with the same boundary conditions (e.g. the calculation of $\rho_{Jacobi}$ eq. 4.19).

$$\rho_{Jacobi} = \frac{cos\frac{\pi}{N_x} + \left(\frac{\triangle x}{\triangle y}\right)^2 cos\frac{\pi}{N_y}}{1 + \left(\frac{\triangle x}{\triangle y}\right)^2} \tag{4.19}$$

Another acceleration can be achieved by splitting the model grid into red and black (Röckle 1990) or "odd" and "even" (Press et al. 2007, p. 1064) cells like on a chequerboard. Now for every iteration step, first, the red cells are updated. Afterwards the black cells are updated based on the already updated red ones.
SOR can be even more accelerated using Chebyshev acceleration. Chebyshev acceleration modifies $\omega$ at the end of every iteration step. Divergence during the first iteration steps can be reduced, and the convergence is speeded up by using an $\omega$ of one for the first part of the first

iteration step (red cells) and adjust $\omega$ according to eq. 4.20 for the second part (black cells) of the first iteration step.

$$\omega^{1/2} = 1/(1-\rho^2_{Jacobi}/2) \tag{4.20}$$

$$\omega^{n} = 1/(1-\rho^2_{Jacobi}\omega^{n}/4) \tag{4.21}$$

For any later iteration step $n$, $\omega$ is adjusted according to eq. 4.21. $\omega$ will now converge to the optimal $\omega$ along with an increasing number of iteration steps.

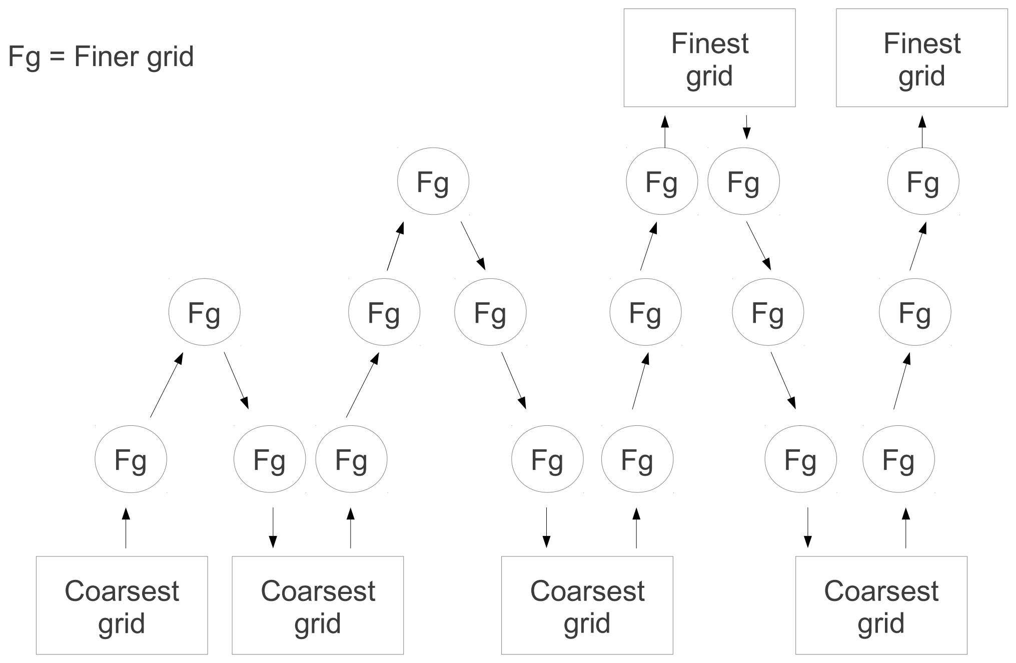


**fig. 4.6:** Structure of a full multigrid cycle with four grids (the target grid and three coarser ones). Modified after Press et al. (2007).

**Multigrid method** Multigrid methods as described in Press et al. (2007, p. 1066 ff) can be divided into two types: The basic multigrid methods and the Full MultiGrid algorithm (FMG). In this section only FMG will be described, as it is faster and more commonly used in wind modelling (e.g. in

Wang et al. 2005).
The basic approach of FMG is that the initial values for a finer grid, have to match the interpolated results of a coarser grid. Due to the smaller dimensions, the solution for the coarser grid is more easy to determine. For the interpolation from the finer to the coarser grid and back, a linear "restriction operator" (Press et al. 2007, p. 1068) has to be used. At the coarsest level, an exact solution can be calculated. It is used as input for the finer grids. The solution for the finer grid is approximated by first interpolating from the coarsest to the next finer grid. Now a correction is made by averaging back to the coarser grid. In the next step the interpolation is performed again from the coarsest to the next finer grid (the corrected one), and another interpolation is run to the next finer grid. This "v-cycles" are repeated until the finest grid, the actual model domain, is reached (compare to fig. 4.6).
Although the calculation of a full cycle may require a lot of grids to be calculated, the better correction may still make FMG to perform out a "rapid" direct solver (like SOR). A disadvantage of FMG is that the mathematical problem needs to be defined for every single grid.

#### 4.3.4.3 ABC model

Most diagnostic small scale models calculating air flow around buildings are based on, or at least strongly inspired by a model concept presented by Röckle (1990).
The methodology described there was first applied in the ABC (Airflow around Building Clusters) model, the original Röckle model. The ABC model is able to approximate a wind field in high spatial resolution respecting cubic obstacles in complex configurations (Gross et al. 1994). It was later implemented into the dispersion model ASMUS (Ausbreitungs- und StrömungsModell für Urbane Strukturen, Gross et al. 1994, Gross 1997).

**Initial wind field** For a single cuboid shaped obstacle with a certain length $l$ parallel to the wind direction, a width of $wi$ perpendicular to the air current, and a height of $h$ (all in m), the ABC model specifies three separate zones of different modifications to the wind field. They are called the front eddy system, continuing sidewards as a horseshoe eddy, the close wake, and the far wake (compare to fig. 4.7).

For a non-perpendicular incident flow, the front eddy will get smaller. The length of the far wake, however, will be extended from 1.4 $h$ for perpendicular incident flow to up to 2 $h$ for diagonal incident flow (Röckle 1990).

**Vertical profile** As there were only wind tunnel studies for buildings available concerning adiabatic atmospheric conditions, the ABC model uses a logarithmic wind profile, for the vertical gradient

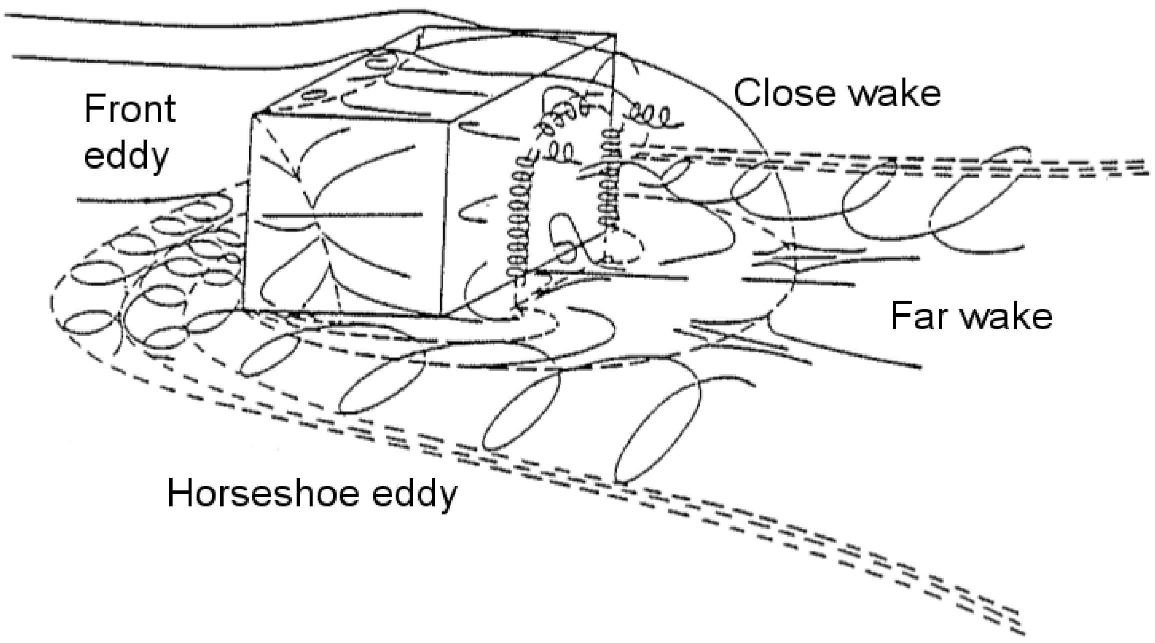


**fig. 4.7:** Draft of the different characteristical elements of the flow around a cubical obstacle (modified after Hunt et al. 1978).

of the incident flow (compare to section 4.3.1.1). To avoid the zero plane displacement relation, eq. 4.2 is used for two different altitudes (eq. 4.22).

$$\bar{u}(z) = \bar{u}_{ref} \frac{ln\left(\frac{z-d}{z_0}\right)}{ln\left(\frac{z_{ref}-d}{z_0}\right)} \tag{4.22}$$

Thereby $\bar{u}_{ref}$ is the stream velocity in m/s at a height of $z_{ref}$ m. Eq. 4.22 is applied to calculate the rooftop wind speed. For the flow within the mean obstacles height ($z < \bar{h}$) a wind speed constantly increasing along the height is assumed. As a reference speed, the approximated wind speed for the mean obstacle height is used.

$$\bar{u}(z) = \bar{u}(\bar{h}) \tag{4.23}$$

For the zero plane displacement $z_d$ (in m) a literature value after Panofsky and Dutton (1984) of 0.8 times the mean building height is used. The mean building height is approximated by the ratio of the buildings volume to the under-roof area (eq. 4.24).

$$\bar{h} = \frac{\sum_i (w_i \cdot l_i \cdot h_i)}{\sum_i (w_i \cdot l_i)} \tag{4.24}$$

The roughness length $z_0$ (m) is assumed to be 20% of the ratio of the total buildings volume to the model area (eq. 4.25).

$$z_0 = 0.2 \frac{\sum_i (w_i \cdot l_i \cdot h_i)}{N_x \Delta x \cdot N_y \Delta y} \tag{4.25}$$

The velocity components in x and y direction (both in m/s) are calculated from the mean velocity $\bar{u}$ (m/s) at the specific height $z$ (m) and the wind direction $WD$ (°) using eq. 4.26.

$$\begin{aligned} u_z &= -\bar{u}(z) \cdot sin(WD) \\ v_z &= -\bar{u}(z) \cdot sin(WD) \end{aligned} \tag{4.26}$$

For the initialisation of the lattice, eq. 4.26 is also used for the initial wind speed in x and y direction ($x^0$ and $y^0$, both in m/s). The initial vertical speed $z^0$ (m/s) is set to zero for the whole grid (Röckle 1990).

**Front eddy system** On the front side of an obstacle the ABC model considers a stagnation zone. Due to the vertical atmospheric wind profile resulting in increased wind speed along increasing height, pressure is assumed to be higher in front of the upper part of the building, than in front of the lower part. This leads to a downwards air current that turns against incident flow when it reaches the foot of the building and forms an eddy in front of it. Dependent on turbulence in the incident flow, the front eddy system can extend over the sides of the obstacle, where it forms a horseshoe shaped zone of eddies (compare to fig. 4.7).
The spatial extension of the front eddy system is calculated by eq. 4.27 in the ABC model.

$$\frac{L_f}{h} = \frac{2.0 \cdot \left(\frac{wi}{h}\right)}{1 + 0.8 \cdot \left(\frac{wi}{h}\right)} \tag{4.27}$$

$L_f$ stands for the streamwise horizontal extension of the front eddy system from the buildings wall in m. Vertically, it reaches up to the stagnation point at approximately 0.6 times the obstacles height. To consider a non perpendicular incident flow, the front eddy zone is split up into one for each of the two walls in the windwards directions (fig. 4.8). The equations for the calculation of the extent and the presetting of the front eddy zones are given in tab. 4.1. Vertically, the obstacles influence is decreased along a quarter ellipse with a maximum extension at ground level decreasing up to the obstacles top.

**tab. 4.1:** The length of the two semiaxis $a_x$ and $a_y$ (both in m) and the presetting (in m/s) for the calculation of the front eddy zone in the x ($FE_x$) and the y ($FE_y$) direction of a single obstacle setting. Modified after Röckle (1990).

| Front eddy zone | $FE_x$ | $FE_y$ |
|---|---|---|
| Semiaxis $a_x$ (m) | $L_f sin^2(WD)\sqrt{1-\left(\frac{z}{0.6h}\right)^2}$ | $\frac{l}{2}$ |
| Semiaxis $a_y$ (m) | $\frac{wi}{2}$ | $L_f cos^2(WD)\sqrt{1-\left(\frac{z}{0.6h}\right)^2}$ |
| Presetting (m/s) | $u_0 = 0$ | $v_0 = 0$ |

**Close wake** In the lee of an obstacle, a cavity zone with low wind speed, and a mean wind direction opposite to the incident flow but with high turbulence is considered. The wake zone in the ABC model is defined as an ellipsoidal volume with a length of 1 $L_R$ (m) in the lee of the obstacle and a width of approx. the obstacles width $wi$. The horizontal recirculating cavity length is controlled by the furthest distance from the obstacle of the air-flow re-attaching to the ground, $L_R$ (Hosker 1985). It is calculated by (eq. 4.28, Hosker 1985).

$$\frac{L_R}{h} = \frac{1.8 \cdot \frac{wi}{h}}{\left(\frac{l}{h}\right)^{0.3} \cdot \left(1 + 0.24 \cdot \frac{wi}{h}\right)} \tag{4.28}$$

Again, $wi$ and $h$ are the width and height of the obstacle in m, its length in flow direction is specified as $l$ (m).

For the calculation of an obstacles close and far wake, the obstacle has to be projected resulting in a perpendicular incident flow. The projected obstacle is then moved along the wind direction until its rear wall covers the furthest lee side edge of the original obstacle (as shown by the dashed line in fig. 4.8). While the height of the projected obstacle is the same as that of the original one, its length ($l_{po}$) and width ($w_{po}$) in m have to be calculated. The width is obtained by the projection of the object perpendicular to the incident flow (compare to fig. 4.8). $l_{po}$ is calculated geometrically, respecting conservation of area. $w_{po}$ and $l_{po}$ are inserted in eq. 4.28 to calculate the extension of the close wake $L_R$. For the far wake, 3 $L_R$ is used. Basically both, the close and the far wake zone are ellipsoids with their half axis $a_{x'}$ and $a_{y'}$ (both in m) that are defined in eq. 4.29.

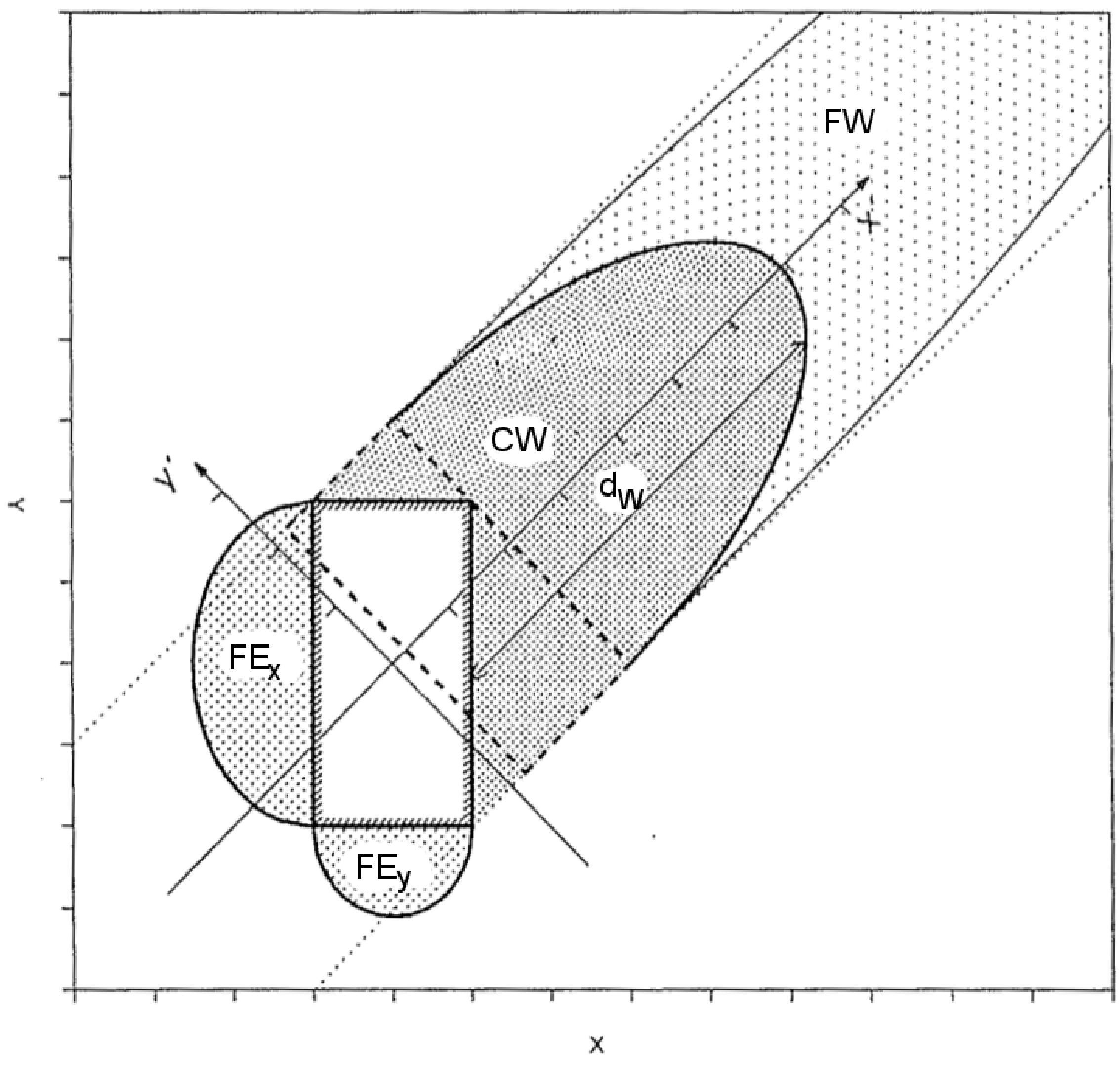


**fig. 4.8:** Sketch showing the different zones of an obstacles influence on the wind field ($FE_x/FE_y$ = front eddy system in x and y direction, CW = close wake, FW = far wake, $d_W$ = Distance between back wall of the virtual obstacle and the end of the close wake zone) assuming incident flow from the lower left corner (↗). Modified after Röckle (1990)

$$a_{x'} = L_R\sqrt{1-\left(\frac{z}{h}\right)^2} \qquad a_{y'} = \frac{w_{po}}{2} \tag{4.29}$$

Within the close wake, an air flow in opposition to the incident flow is assumed. Its wind speed is proportional to the rooftop wind speed and is decreasing along increasing distance to the obstacle. The influence of the close wake on the x and y component of the wind field in m/s is calculated for a point of the lattice with a specific distance to the lee side wall of the obstacle $d_l$ in m by eq. 4.30.

$$
\begin{aligned}
u^0 &= -u(h)\left(1-\frac{d_l}{d_W}\right)^2 \\
v^0 &= -v(h)\left(1-\frac{d_l}{d_W}\right)^2
\end{aligned}
\tag{4.30}
$$

Vertically, the decrease is handled like the one of the front eddy system.

**Far wake** In further distance to the obstacle than the close wake, a far wake zone with reduced wind speed is defined. Thereby the reduction in the flow velocity compared to the incident flow decreases with increasing distance $d$ in m to the obstacle. In the ABC model this decrease is defined as $d^{-1.5}$. The horizontal extension varies according to incident wind speed and the building properties, but may not exceed 80 times the obstacles height (Röckle 1990).

The elliptical far wake zone is calculated analogue to the close wake zone, but with a semiaxis $a_{x'}$ (m) that is three time as long as the one of the close wake eq. 4.31.

$$
\begin{aligned}
a_{x'} &= 3L_r \cdot \sqrt{1-\left(\frac{z}{h}\right)^2} \\
a_{y'} &= \frac{w_{po}}{2}
\end{aligned}
\tag{4.31}
$$

Points in the far wake are addressed the same way than those in the close wake, while their wind speed is increasing along increasing distance to the obstacles lee side wall up to a maximum equal to the incident flow at a distance of 3 $L_r$. The influence of the far wake on the wind fields x and y component in m/s is calculated by eq. 4.32.

$$
\begin{aligned}
u^0 &= u(z)\cdot\left(1-\left(\frac{d_W}{d_l}\right)^{1.5}\right) \\
v^0 &= v(z)\cdot\left(1-\left(\frac{d_W}{d_l}\right)^{1.5}\right)
\end{aligned}
\tag{4.32}
$$

The vertical decrease is handled analogue to the one of the front eddy system.

**Interaction between influence of buildings** To address problems in a multi obstacle setting, in that a lattice point is situated inside several zones (e.g. inside the far wake zone of a windwards obstacle, but also inside the front eddy zone of the next obstacle), the different zones in the ABC model have different priorities. In case of conflict, always only the highest priority is concerned in the construction of the initial wind field. The priorities are as follows:

1. close wake
2. front eddy system
3. far wake

To approximate the interaction between the influence of different obstacles, the ABC model distinguishes three different regimes for different combination of obstacles. Following the approach of Hosker (1985) they can be separated by the distance $s$ in m between two facing walls and the metric height $h$ of the upstream building.

I If two obstacles are placed in a distance to each other of at least $s_i$ m, their influence on the wind field is considered to be independent. Meeting the precondition of $0.5 \leq w/h \leq 4$, this distance can be calculated by eq. 4.33.

$$\frac{s_i}{h} = 1 + 1.4\left(\frac{w}{h}\right)^{0.25} \tag{4.33}$$

II For obstacles that are located closer together, interactions between the close wake of the upstream building and the front eddy system of the downstream building have to be considered. If $w/h \leq 2$ the minimum distance $s_s$ for this second regime can be calculated by eq. 4.34. If $wi/h > 2$, $s_s$ is calculated by eq. 4.35.

$$\frac{s_s}{h} = 1.25 + 0.15\left(\frac{w}{h}\right) \tag{4.34}$$

$$\frac{s_s}{h} = 1.55 \tag{4.35}$$

III If two obstacles are closer together than $s_s$ the air current is assumed to disconnect from the ground and to flow over the gap. Between the two buildings a horizontal vortex flow is assumed (Röckle 1990).

Each lattice point is classified into one of the three regimes separately in x and y direction.
For the points in regime I, no changes have to be made and the wind field modification can be calculated according to tab. 4.1 and eqs. (4.30), (4.32) and (4.32).
For points to be found to fit into regime II, a rather arbitrary check is performed including all points

in a range of +/- 25 m. For each of those points the distance to the closest obstacles on each side perpendicular to the check axis is calculated. If this distance is found to be 30 m or less for at least half of the points in the consideration, the original point is reclassified to be regime III.
For a point in regime III only the wind speed perpendicular to the wall is changed. Additionally a vertical speed is computed as a point symmetric volume with a maximum at the obstacles walls and a minimum at the center of the space between them. This pace is considered to be a street canyon. The equations for the modification inside a street canyon in x and y direction are given in tab. 4.2. Thereby $u_{rt}$ and $v_{rt}$ (both in m/s) are defined as the components of the wind speed at the top of the higher building, $d_l$ is the metric distance of a specific point on the lattice to the upstream building, and $W$ is the width of the street in m.

**tab. 4.2:** Specification of the wind field modification inside a street canyon (regime III). $u_{rt}$ and $v_{rt}$ are defined as the components of the wind speed at the top of the higher building in m/s, $d_l$ is the distance of a specific point on the lattice to the upstream building in m, and $W$ is the width of the street in m. Modified after Röckle (1990).

| Street canyon in | x direction | y direction |
|---|---|---|
| x component | $u^0 = u(z)$ | $u^0 = -u_{rt}\left(\frac{d_l}{W}\right)\left(\frac{W-d_l}{W/2}\right)$ |
| y component | $v^0 = -v_{rt}\left(\frac{d_l}{W}\right)\left(\frac{W-d_l}{W/2}\right)$ | $v^0 = v(z)$ |
| z component | $w^0 = \lvert\frac{u_{rt}}{2}\left(1-\frac{d_l}{W/2}\right)\rvert\left(1-\frac{W-d_l}{W/2}\right)$ | |

**Divergence** The wind field calculated by the given functions most likely contains a lot of divergence. Assuming incompressible air, this divergence has to be minimized in order to get a valid wind field, as air must not disappear or be created anywhere. Mathematically this is performed by minimizing the functional for the scalar H in eq. 4.36.

$$H(u,v,w) = \iiint \left(\alpha_h^2(u-u^0)^2 + \alpha_h^2(v-v^0)^2 + \alpha_v^2(w-w^0)^2\right) dxdydz \qquad (4.36)$$

In eq. 4.36 $\alpha_h$ and $\alpha_v$ are horizontal stability factors in s/m, $u$, $v$ and $w$ are the stream components in m/s, $u^0$, $v^0$ and $w^0$ are representing the initial stream components in m/s while $dx$, $dy$ and $dz$ are the grid spacings in m.
The functional is to be solved respecting the side condition that the considered medium is incompressible, given in eq. 4.37 .

$$G(u_x, v_y, w_z) = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0 \qquad (4.37)$$

Applying the Lagrangian multiplication method the side condition 4.37 can be written into the functional eq. 4.36 resulting in the functional 4.38 including the Lagrangian multiplication factor $\lambda$ (s).

$$E(u,v,w,u_x,v_y,w_z,\lambda) = \iiint \left(\alpha_h^2(u-u^0)^2+\alpha_h^2(v-v^0)^2+\alpha_v^2(w-w^0)^2+\lambda\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}\right)\right)dxdydz \tag{4.38}$$

This functional can be solved using the Euler-Lagrange equations (eqs. (4.39) to (4.41), Sherman 1978).

$$u = u^0 + \frac{1}{2\alpha_h^2}\frac{\partial\lambda}{\partial x} \tag{4.39}$$

$$v = v^0 + \frac{1}{2\alpha_h^2}\frac{\partial\lambda}{\partial y} \tag{4.40}$$

$$w = w^0 + \frac{1}{2\alpha_h^2}\frac{\partial\lambda}{\partial z} \tag{4.41}$$

To be able to solve the Euler-Lagrange equations the air has to be considered incompressible (eq. 4.42), what is a valid assumption regarding the lowest decameters of the atmosphere as long as thermo-dynamics are not considered (Röckle 1990).

$$0 = \frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z} \tag{4.42}$$

At the borders, the border condition given in eq. 4.43 is to be complied to. So either the development in the velocity normal to the border in m/s ($\delta u_n = u_n - u_n^0$ ) or the Lagrangian multiplication factor $\lambda$ has to equal 0 at a model border. The first case will result in a closed border, as no flow through this border is accepted. This is used to respect solid borders like buildings. The second case allowing for flow through the border is used for the permeable limits of the model domain (Sherman 1978).

$$\lambda\delta u_n = 0 \tag{4.43}$$

Assuming that the weighting factors $\alpha_h$ and $\alpha_v$ are constant within the model area, the Euler-Lagrange equations (4.39 - 4.41) can be differentiated, and substituted into eq. 4.42. The result is a Poisson equation (eq. 4.44).

$$\frac{\partial^2\lambda}{\partial x^2}+\frac{\partial^2\lambda}{\partial y^2}+\left(\frac{\alpha_{h^2}}{\alpha_v^2}\right)\frac{\partial^2\lambda}{\partial z^2}=-2\alpha_h^2\left(\frac{\partial u^0}{\partial x}+\frac{\partial v^0}{\partial y}+\frac{\partial w^0}{\partial z}\right) \tag{4.44}$$

Eq. 4.44 has no analytical solution. However, it can be solved numerically for discrete grid points by iteration if it is converted into a finite difference equation. The ABC model uses the successive over-relaxation method (compare to section 4.3.4.2) by Roache (1982) (see Press et al. 2007, p. 1062 ff) to approximate a solution. It provides grid point values for $\lambda$ that can be used together with the Euler-Lagrange equations (4.39 - 4.41) to solve them. This approximates a non divergent flow field.
The ABC model uses an equidistant grid for the three dimensions $(x,y,z)$, so $\triangle x$, $\triangle y$, and $\triangle z$ are constant. Under this precondition a discrete form of eq. 4.44 can be written (eq. 4.45).

$$\begin{aligned}&\frac{\lambda_{i+1jk}-2\lambda_{ijk}+\lambda_{i-1jk}}{(\triangle x)^2}+\frac{\lambda_{ij+1k}-2\lambda_{ijk}+\lambda_{ij-1k}}{(\triangle y)^2}+\\&\left(\frac{\alpha_{h^2}}{\alpha_v^2}\right)^2\left(\frac{\lambda_{ijk+1}-2\lambda_{ijk}+\lambda_{ijk-1}}{(\triangle z)^2}\right)=-2\alpha_h^2\varepsilon_{ijk}^0\end{aligned} \tag{4.45}$$

In eq. 4.45 $\varepsilon^0$ represents the divergence of the wind field at grid point $ijk$.
For not having to differentiate over the distance of two grid cells, the ABC model calculates divergence using a staggered grid for the air velocity (eq. 4.46).

$$\varepsilon_{ijk}^0=\frac{u^0_{i+\frac{1}{2}jk}-u^0_{i-\frac{1}{2}jk}}{\triangle x}+\frac{v^0_{ij+\frac{1}{2}k}-v^0_{ij-\frac{1}{2}k}}{\triangle y}+\frac{w^0_{ijk+\frac{1}{2}}-w^0_{ijk-\frac{1}{2}}}{\triangle z} \tag{4.46}$$

This guarantees consistency of mass at each point of the $\lambda$ grid. On the other hand the velocity at the center of a cell now has to be calculated by averaging the velocity of its borders. Other difficulty is created by walls and model borders that now are no longer defined at cell borders, but at the cell centres of the velocity grid.

**Border Conditions** The ABC model considers two kinds of border conditions. Closed borders are borders, where the flow velocity has to be zero, as the air must not flow through them. Examples for closed borders are the ground and building walls. Technically the flow velocity normal to closed borders is set to zero. Later compliance with the border condition given in eq. 4.43 ensures the flow

velocity through the border will stay zero.
Open borders, in contrast, allow air to flow through them. Examples for open borders are the vertical domain borders and the model top. At these borders $\lambda$ has to equal zero.
Both types of borders need to be respected during the discretization of the Poisson equation (4.44).

**Validation** The ABC model is evaluated by several studies, e.g. Bagal et al. (2004), Röckle (1990), Singh et al. (2008).
The first validation of the ABC model was done by Röckle (1990), who compared the results of his calculations to full-scale measurements taken at seven ground stations. Röckle (1990) concludes that the model is capable to calculate wind speed and direction with sufficient precision. Also comparisons to the results of other models have been performed that show that the basic structure of the flow field compares very well (Gross et al. 1994). More complex validation has been performed by Singh et al. (2008), who compared model results to data gathered from a wind tunnel experiment for a cubical building array of 11 cubes streamwise and seven cubes crosswise with perpendicular incident wind. They conclude that the results calculated by the ABC model "are in reasonable agreement with experimental data" (Singh et al. 2008).
In detail, the ABC model is found to set the stagnation point on the windward side of the first building rather precise (compare to fig. 4.9, Singh et al. 2008). Also the length of the lee-side stagnation zone compares well, but seems to be little to long (Bagal et al. 2004). However, the calculated velocities do not match the measured ones very well (Bagal et al. 2004). The recirculation zone windward of the obstacle shown by the physical model becomes a "large unorganized cavity with no well-defined features" in the results by the ABC model (Singh et al. 2008).
As the ABC model does not contain a rooftop recirculation scheme, the streemwise velocity on top of the buildings is found to be overestimated by Singh et al. (2008).
The length of the wake zone in the lee of the obstacle of approximately 1.5 times the obstacles height, is found to compare well to full scale measurements (Gross et al. 1994).
Also the vortex calculated for the first street canyon seems to be overestimated. Its downdrafts and backwards flow are stronger than those measured in the wind-tunnel experiment. As the upwards drafts are overestimated two, the modelled vortex rises higher, than the measured one in the physical model (Singh et al. 2008).

Analysis of the wind patterns computed for the first street canyon by the ABC model shows a strong vortex that reaches, horizontally, up to the ends of the street canyon. Comparison with the measurements gathered from the wind tunnel experiment, the calculated vortex is little too strong and too wide (Compare to fig. 4.10) (Singh et al. 2008).

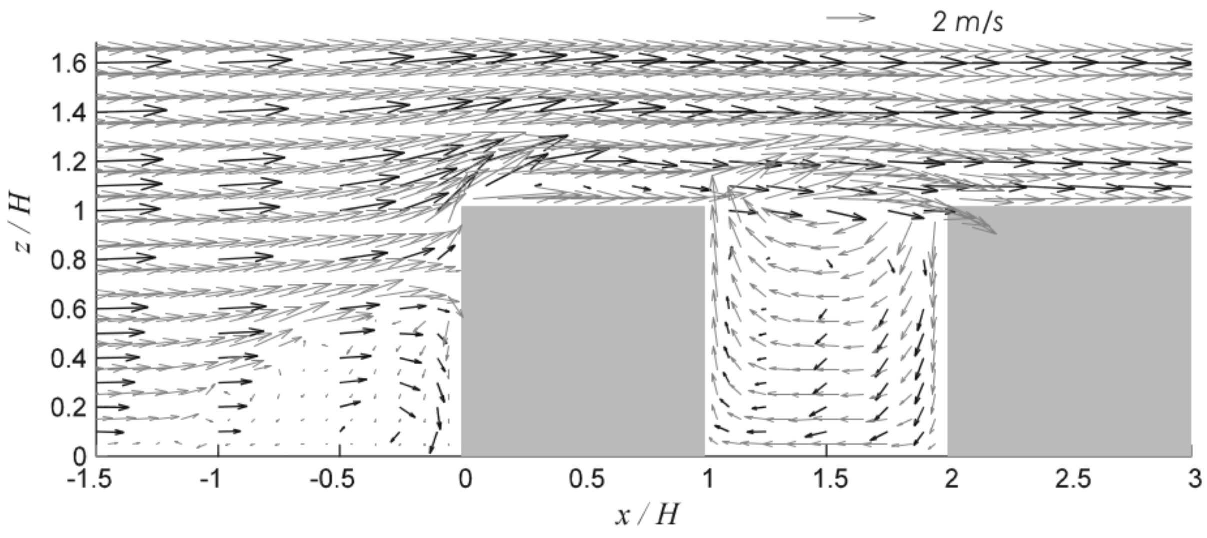


**fig. 4.9:** Comparison between the results of the ABC model calculations (grey arrows) to experimental measurements achieved during a wind-tunnel experiment (black arrows). The figure shows a vertical cut parallel to the x axis through the center of a building array (Singh et al. 2008)

#### 4.3.4.4 TALdia

TALdia is a diagnostic wind field model that is mainly applied in the AUSTAL2000 model for the purpose of particle dispersion calculation. TALdia is able to consider buildings, as well as complex terrain in the calculations. Therefore a first run for the influence of the terrain and a second run for the influence of the buildings is performed.
Assuming a model area with both, terrain and buildings, the following steps are taken to compute the flow field (Janicke Consulting 2011).

- A homogeneous incident flow field is calculated using a terrain following coordinate system.
- The influence of the terrain is calculated based on the mean terrain roughness concerning atmospheric stability. Divergence is minimized afterwards.
- The logarithmic wind profile of the Prandtl-layer is applied to the flow field.
- Divergence is minimized again.
- For the calculation of the influence of buildings, the flow field is projected to a plane coordinate system. The flow around the buildings is calculated on the plane wind field.
- Divergence is minimized for the third time and the resulting wind field is reprojected to the terrain following coordinate system.

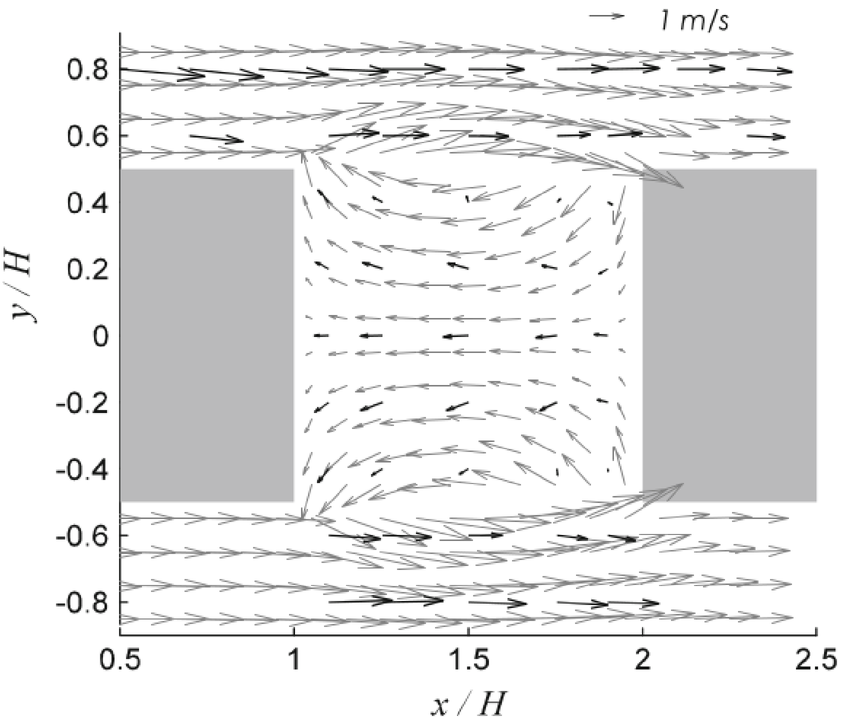


**fig. 4.10:** Comparison between the results of the ABC model calculations (grey arrows) to experimental measurements achieved during a wind-tunnel experiment (black arrows). The figure shows a horizontal cut through the first street canyon (Singh et al. 2008)

**Modification by the terrain** In TALdia, the modifications to the wind field by the terrain are calculated on a terrain following grid addressed as a non-Cartesian coordinate system. Therefore a coordinate $s$ is calculated that is used instead of the absolute height $z$. The horizontal components u and v are considered relative to $s$, resulting in the horizontal fluxes to be parallel to the ground (Janicke Consulting 2011).

The terrain modification to create the initial wind field are basically determined by calculating a potential flow for a terrain grid with a super elevation according to the stability parameter $\alpha_v$. This potential flow is then reprojected to the main grid. An air current will now flow around a hill stronger than over it (for a high value of $\alpha_v$), or the other way round (Janicke Consulting 2011).

The local divergence at a certain grid cell is determined on an ARAKAWA-C-grid (Arakawa and Lamb 1977) according to the Gaussian divergence theorem, stating that the local divergence is the flux through a grid cell, divided by the cells volume (Janicke Consulting 2011).

For the minimisation of the divergence created by the modification of the wind field due to concerning terrain, the Alternate Directions Implicit (ADI) method is used (Press et al. 2007, p. 872 f). This method is selected for it is more stable to variable grid size, resulting from steep terrain (Janicke Consulting 2011). The disadvantage of the ADI method is that convergence is not guaranteed

(Janicke Consulting 2011).

### 4.3.4.5 QUIC-URB

The QUIC (Quick Urban & Industrial Complex) model is a dispersion model based on the calculation of the diagnostic wind field model "QUIC-URB" that can handle complex urban environments (Pardyjak et al. 2004). It is mainly based on the approach of Röckle (1990) and Kaplan and Dinar (1996), but also contains some additional parametrizations that are simple and physically based (Singh et al. 2008). For example a new upwind cavity parametrization (Bagal et al. 2004) and a shelter model introduced by Taylor and Salmon (1993) was integrated.

**Improved upwind cavity** Bagal et al. (2004) compared the results of the calculations using the front eddy parametrization after Röckle (1990) to measurements gathered from wind tunnel experiments. They found that the behaviour of the parametrization for different obstacle heights and widths was fine, but the results seemed to be offset. They therefore modified eq. 4.27 to eq. 4.47.

$$\frac{L_f}{h} = \frac{1.5\left(\frac{wi}{h}\right)}{1+0.8\left(\frac{wi}{h}\right)} \tag{4.47}$$

Unlike in the original parametrization after Röckle (1990), the ellipsoid of the front eddy zone is not only initialized by overwriting the wind component perpendicular to the obstacle by zero, but it is filled with a modified power law profile. The profile contains a factor that reduces the wind component perpendicular to the obstacle (compare to eq. 4.48).

$$\frac{\bar{u}(z)}{U_H} = 0.4 \cdot \left(\frac{z}{H}\right) \tag{4.48}$$

Also a vortex zone was included into the front eddy zone. It is of the same shape as the stagnation zone, but the perpendicular flow is not only set to zero, but a vortex is initiated. The length of the vortex zone is calculated by eq. 4.49.

$$\frac{L_{fv}}{h} = \frac{0.6\left(\frac{wi}{h}\right)}{1+0.8\left(\frac{wi}{h}\right)} \tag{4.49}$$

The ellipsoid of the vortex zone is then filled by a parametrization gathered from fitting experimental wind-tunnel data. The trigonometric eqs. (4.50) and (4.51) are used to calculate the vortex components in the horizontal (eq. 4.50) and the vertical (eq. 4.51) direction.

$$\frac{\bar{u}(z)}{U_H} = \left(0.6cos\left(\frac{\pi h_v}{0.5H}\right)+0.05\right)\cdot\left(-0.6sin\left(\frac{\pi l_v}{L_{fv}}\right)\right) \tag{4.50}$$

$$\frac{\bar{w}(z)}{U_H} = -0.1cos\left(\frac{\pi l_v}{L_{fv}}\right)-0.05 \tag{4.51}$$

$l_v$ and $h_v$ are defined as the current length and height of the vortex.

**Shelter model by Taylor and Salmon** Taylor and Salmon (1993) introduced improved parametrizations for the velocity deficit zone in the lee of an obstacle. The shelter model by Taylor and Salmon (1993), that is also adopted in the QUICK-URB model calculates a Gaussian velocity deficit pattern using equations 4.52 to 4.54.

$$\frac{u_d(x,y,z)}{U(H)} = \Gamma C_D\left(\frac{W}{H}\right)\left(\frac{x}{H}\right)^{-1.5}F(\eta)G(\zeta) \tag{4.52}$$

$$F(\eta) = \left(\frac{1}{(2\pi)^{0.5}a_f}\right)^{\left(\frac{-\eta^2}{2a_f^2}\right)} \tag{4.53}$$

$$G(\zeta) = (c_a\zeta)^{(-a_g\zeta^{1.5})} \tag{4.54}$$

$u_d$ represents the velocity deficit in the lee of an obstacle in m/s, x, y, and z are the stream coordinates in x-, y-, and z-direction, W the width, and H the height of an obstacle in m, U(H) the mean wind speed at the top of the obstacle in m/s based on the upstream power law profile, and $C_D$ is the drag coefficient. $\Gamma$ is defined as $0.6\cdot c_a^2$. The similarity coordinates $\eta$, and $\zeta$, as well as the vertical coefficients $c_a$, and $a_g$ are calculated according to the following equations:

$$\eta = \left(\frac{z}{H}\right)\cdot\left(\frac{x}{H}\right)^{-0.5} \tag{4.55}$$

$$\zeta = \left(\frac{y}{H}\right)\cdot\left(\frac{x}{H}\right)^{-0.5} \tag{4.56}$$

$$c_a = \sqrt{\frac{\left(\frac{ln(H+z_0)}{z_0}\right)}{2\cdot\kappa^2}} \tag{4.57}$$

$$a_g = 0.8 \cdot c_a^{1.5} \tag{4.58}$$

$\kappa$ in eq. 4.57 is the von Kármán constant of $5.67 \cdot 10^{-8} \frac{W}{m^2 \cdot T_s^4}$.
The main advantage of the integration of the shelter model by Taylor and Salmon (1993) is the streamwise velocity deficits that can be calculated more accurately than in the original ABC model. Still they are slightly overestimated compared to measurements (Pardyjak et al. 2004). For their analysis Pardyjak et al. (2004) applied an improved equation for the determination of the recirculation length $a_x$ (or $a_y$ respectively) at level $z$ in m (eq. 4.59)

$$a_x = L_R \sqrt{1.0 - \left(\frac{z}{h}\right)^2} - \frac{l}{2} \tag{4.59}$$

In eq. 4.59, $h$ is the height of the obstacle in m. Its length in flow direction is specified as $l$ (m). $L_R$ describes the maximum length of the recirculation in m.

**Modified street canyon model** As a response to the overestimation of the width of a street canyon vortex shown by wind tunnel experiments (compare to section 4.3.4.3), a modified street canyon model was implemented into the QUIC-URB model (Singh et al. 2008). A basic difference to the street canyon model after Röckle (1990) is the vertical wedges at both ends of the street canyon that are considered separately. A turbulent diffusion region is calculated there to consider transport of momentum into the street canyon or out of it (blue triangles in fig. 6.6).
The vertical wedges are modified after eq. 4.60, where $d_{sc}$ represents the distance from the upwind building.

$$\frac{u^0}{u_{rt}} = \frac{tanh\left(\frac{0.2 \cdot d_{sc} - H}{0.2 \cdot d_{sc}}\right)}{tanh(1)} \tag{4.60}$$

Also the criteria to detect a street canyon has been modified. It is now based on the upstream buildings recirculation zone. Whether a street canyon is set or not is distinguished according to eq. 4.61.

$$\frac{L_R}{H} = \frac{1.8 \cdot \frac{W}{H}}{\left(\frac{L}{H}\right)^{0.3} \left(1 + 0.24 \cdot \frac{W}{H}\right)} \tag{4.61}$$

For the central part of a street canyon Singh et al. (2008) propose a streamwise speed modification according to eq. 4.62.

$$\frac{u^0}{u_{rt}} = -0.3 \cdot \frac{d_{nW}}{0.5 \cdot d_{SCw}} \cdot \left(\frac{d_{SCw} - d_{nW}}{0.5 \cdot d_{SCw}}\right) \cdot F_{SC} \tag{4.62}$$

Eq. 4.62 calculates the street canyon modification from the distance to next wall $d_{nw}$, the street canyon width $d_{SCw}$ (both in meters) and the function $F_{SC}$, stated by eq. 4.63.

$$F_{SC} = \left(1.0 - \frac{|d_{SCccw}|}{\frac{wi-2.0\cdot(0.2\cdot d_{nw})}{2.0}}\right)^{0.25} \tag{4.63}$$

The only new parameter in eq. 4.63 is the crosswind distance to the street canyons center $d_{SCccw}$ in m.

### 4.3.5 Radiation Modeling

Among the four most important meteorological parameters in human thermal biometeorology ($T_a$ , RH or VP, v, and $T_{mrt}$), the ones with the strongest variability in time and space are v and $T_{mrt}$. They both also show strong impact on human thermal perception and, thus, are most important for modelling thermal conditions. While the different approaches of modelling wind speed and direction are already discussed in section 4.3.1, some approaches of modelling radiation fluxes are summarized in the following sections.

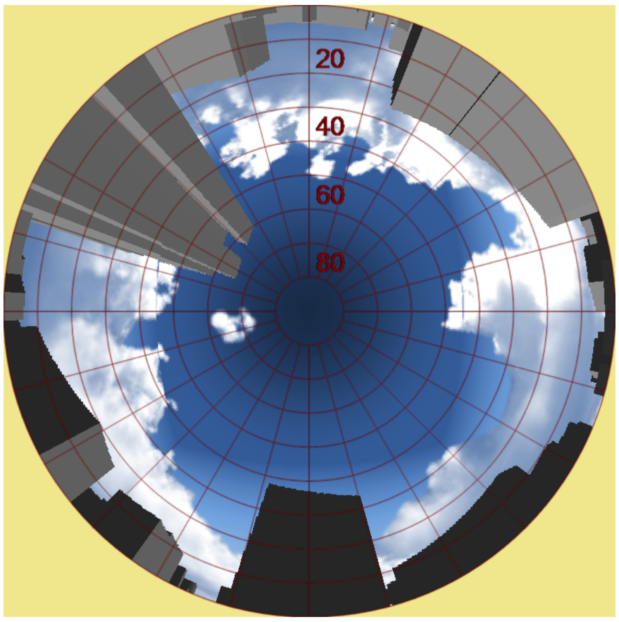
#### 4.3.5.1 Sky View Factor

The sky view factor (SVF) is the fraction of the visible sky, seen from a certain point Oke (1995, p. 353). It is dimensionless and ranges from 0 to 1, where 0 means that the sky is totally covered by terrain or obstacles, while 1 stands for a free sky. The SVF can be calculated as the sum of the squared sine of the terrain or obstacle elevation $\beta_i$ in ° per azimuth angle $a_i$ in ° (eq. 4.64).

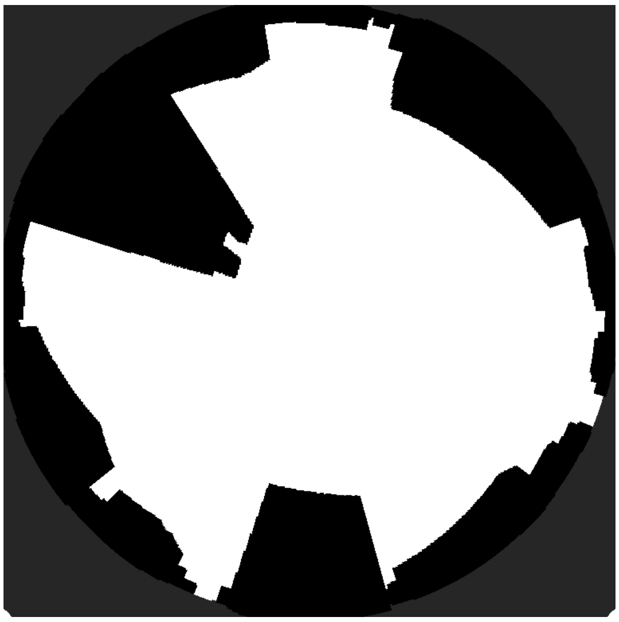
$$\Psi_S = \sum_{i=1}^{n} sin^2 \cdot \beta_i \cdot \left(\frac{a_i}{360°}\right) \tag{4.64}$$

There is two different ways of weighting (to provide planar and spheric SVF) for different purposes (Hämmerle et al. 2011a;b; 2014). The planar SVF is most suitable for all questions about flat surfaces (e.g. the calculation of material heating and reflection) while for objects of interest, that have a vertical extension, e.g. a human being, that can be better represented by a cylinder (Johnson and Watson 1984, Höppe 1984, Watson and Johnson 1988), the spheric SVF is more suitable (Hämmerle et al. 2014).

**Fish-eye image** For numerical models, there is different ways to determine SVF. A popular approach of approximating SVF is rendering a Fish-eye image (compare to fig. 4.11a). SVF can now be determined by distinguishing between the area covered by obstacles and the area showing the sky. This can be done by inserting an overlaying polar grid and counting free and covered cells (Steyn 1980, compare to fig. 4.11a, e.g. applied in Hämmerle et al. (2011b)). Another approach is based on distinguishing white and coloured pixels in the image itself (e.g. applied by the SkyHelios model (section 4.4.2)). Following this approach, the white pixels are usually counted as free sky, while all others are considered as covered by obstacles (compare to figs. 4.11a and 4.11b).
As, in the real environment, the Fish-eye is a half-sphere, not all of the pixel should have the same influence on the SVF. Therefore a dimensionless weighting factor $\omega_{proj}$ (eq. 4.65) is used to consider the projection that adjusts the impact of a pixel by the sine of the zenith angle $\varphi$ (°).



**(a)** Fish-eye image rendered for visualization.



**(b)** Fish-eye image rendered for the calculation of SVF.

**fig. 4.11:** Fish-eye images generated by the SkyHelios model (section 4.4.2) for visualization purpose (left) and the calculation of SVF (right).

$$\omega_{proj} = sin(\varphi) \cdot \left(\frac{\varphi}{90^\circ}\right)^{-1} \tag{4.65}$$

This results in a spheric SVF. If the planar SVF is desired, another correction by $\omega_{planar}$ (dimensionless) needs to be performed (compare to eq. 4.66). It increases the impact of objects close to the ground by the cosine of the azimuth angle (counted from the ground to the top).

$$\omega_{planar} = \omega_{proj} \cdot cos(\varphi) \tag{4.66}$$

**Horizon-angle** Another approach is the determination of the horizon-angle. This is defined as the angle between the ground and the highest elevation or obstacle. The horizon-angle is determined in 1° to 22.5° steps (according to the desired precision) around the point of interest. Finally, eq. 4.64 is used to calculate SVF (e.g. Gal et al. 2009, Unger et al. 2010).

**Shadow-casting method** The SOLWEIG model applies another approach for the calculation of SVF (Lindberg et al. 2008). Lindberg et al. (2008) are using a shadow-casting algorithm (Ratti et al. 2006) creating a number of light sources in a hemisphere around the point of interest. SVF is then determined by counting the number of light sources, that are covered by obstacles or terrain.

According to Hämmerle et al. (2011a) all approaches can deliver results with good accuracy. However, due to input data and settings, uncertainties in the results remain.

#### 4.3.5.2 Shortwave radiation

Global radiation ($G$) describes the shortwave irradiation a surface receives from the upper hemisphere. It consists of the direct solar irradiation ($I$) and the diffuse shortwave irradiation ($D$) (all energy flux densities in $\frac{W}{m^2}$, Matzarakis et al. 2010, Oke 1995, p.14). Both of them are dependent on several parameters and therefore need to be modelled.
For a perfect clear sky condition (with no clouds and no horizon limitation), $G$ can be calculated directly using eq. 4.67 proposed by Jendritzky et al. (1990), VDI (1994; 2008). This is also used to derive an initial global radiation ($G_0$) for further processing.

$$G_0 = 0.84 \cdot I_0 \cdot cos(\varphi) \cdot e^{\left(\frac{-0.027 \cdot \frac{pr}{pr_0} \cdot T_L}{cos(\varphi)}\right)} \tag{4.67}$$

Eq. 4.67 requires the initial direct solar radiation $I_0$, an energy flux density in $\frac{W}{m^2}$, the solar zenith angle $\varphi$ (°), the actual air pressure $pr$ in hPa, as well as the one at sea level for a standard atmosphere ($pr_0$ = 1013 hPa) and the Linke turbidity factor $T_L$.

**Direct solar irradiation** To obtain a better assumption under more complex conditions, $I$ and $D$ need to be estimated separately. According to Jendritzky et al. (1990), $I$ can be estimated as a function of $I_0$ ($\frac{W}{m^2}$), $\varphi$ (°), $T_L$ (dimensionless), the relative optical air mass $r_{opt}$ (dimensionless), the vertical optical thickness of a standard atmosphere $\delta_{opt}$ (dimensionless as well), $pr$ in hPa, and cloud cover $cc$ in octas (0 = clear sky to 8 = overcast sky, eq. 4.68).

$$I = I_0 \cdot cos(\varphi) \cdot e^{\left(-T_L \cdot \delta_{opt} \cdot r_{opt} \cdot \frac{p}{p_0}\right)} \cdot \left(1 - \frac{cc}{8}\right) \tag{4.68}$$

This of course only holds for unshaded conditions. Under shaded conditions, $I$ is usually assumed to equal 0 $\frac{W}{m^2}$.
The relative optical air mass ($r_{opt}$) can be estimated by (eq. 4.69, Kasten and Young 1989).

$$r_{opt} = \left(sin(\beta_s) + 0.50572 \cdot (\beta_s + 6.07995°)^{-1.6364}\right)^{-1} \tag{4.69}$$

In eq. 4.69 $\beta_s$ describes the solar elevation angle in °. Using $\beta_s$ and $r_{opt}$, $\delta_{opt}$ can be estimated following an approach by (eq. 4.70, Kasten 1980).

$$\delta_{opt} = \frac{1}{0.6 \cdot r_{opt} + 9.4} \tag{4.70}$$

**Diffuse shortwave irradiation** Assuming a clear sky and no horizon limitation, the diffuse shortwave irradiation $D$ ($\frac{W}{m^2}$) or the initial diffuse shortwave irradiation $D_0$ ($\frac{W}{m^2}$) for further processing can be estimated by eq. 4.71 only considering the direct solar irradiation $I$ ($\frac{W}{m^2}$) and the solar elevation angle $\beta_s$ in ° (Brown and Isfält 1974, Bruse 1999).

$$D_0 = I_0 \cdot sin(\beta_s) \cdot \left(\frac{\frac{1}{1+8 \cdot sin(\beta_s)^{0.7}}}{1 - \frac{1}{1+8 \cdot sin(\beta_s)^{0.7}}}\right) \tag{4.71}$$

After Valko (1966), $D$ can also be calculated as the sum of isotropic ($D_{iso}$) and anisotropic diffuse radiation ($D_{aniso}$, both in $\frac{W}{m^2}$). The isotropic part can be calculated by eq. 4.72.

$$D_{iso} = (G_0 - I_{clear}) \cdot \left(1 - \frac{I_{clear}}{I_0 \cdot cos(\varphi)}\right) \cdot \Psi_S \tag{4.72}$$

This equation requires the direct solar irradiation assuming a clear sky with no clouds $I_{clear}$ ($\frac{W}{m^2}$). The anisotropic component $D_{aniso}$ ($\frac{W}{m^2}$, eq. 4.73) can be approximated by a similar equation if the sun is visible:

$$D_{aniso} = (G_0 - I_{clear}) \cdot \frac{I_{clear}}{I_0 \cdot cos(\varphi)} \tag{4.73}$$

For the case of the sun covered by horizon or obstacles $D_{0,aniso}$ becomes $0\frac{W}{m^2}$.
For non clear sky conditions, a linear correction after (Valko 1966) can be applied (eq. 4.74). It considers the cloud cover $(cc)$ in octas.

$$D_0 = D_{clear} \cdot \left(1 - \frac{cc}{8}\right) + D_{overcast} \cdot \frac{cc}{8} \tag{4.74}$$

For a completely covered sky $(cc = 8/8)$, Valko (1966) proposes a simplified equation:

$$D = 0.28 \cdot G_0 \cdot \Psi_S \tag{4.75}$$

In that case, global radiation can be approximated by scaling the initial global radiation $(G_0)$ by 0.28 and the local sky view factor $(\Psi_S)$ (Valko 1966).

### 4.3.5.3 Longwave radiation

All surfaces warmer than 0.0 K are emitting longwave radiation according to the Stefan-Boltzmann law (eq. 4.76). It is calculating the longwave radiation flux density $P_{lw}$ $(\frac{W}{m^2})$ emitted by a perfectly black radiator surface $(s)$ at a given surface temperature $T_s$ (K).

$$P_{lw} = \sigma \cdot A_s \cdot T_s^4 \tag{4.76}$$

Elements within the equation are the longwave radiation flux density $P_{lw}$ in $\frac{W}{m^2}$, the Stefan-Boltzmann constant of $5.67 \cdot 10^{-8} \frac{W}{m^2 \cdot T_s^4}$ $(\sigma)$, the radiating surface area $A_s$ in $m^2$, and its surface temperature $T_s$ in K.
Eq. 4.76 is modified by including an emission coefficient $\varepsilon_{lw}$ (dimensionless, eq. 4.77) to be able to apply for non-perfectly black surfaces $(\varepsilon \neq 1.0)$, e.g. for humans with an $\varepsilon_{lw}$ of approximately 0.97 (p. 151 in Fanger 1972, VDI 2008).

$$P_{lw} = \varepsilon_{lw} \cdot \sigma \cdot A_s \cdot T_s^4 \tag{4.77}$$

Finding the appropriate value for $\varepsilon_{lw}$ can get very hard when it comes to longwave irradiation emitted by the atmosphere (Staiger and Matzarakis 2010). While there are empirical values available for quite some time (e.g. Brunt 1932, Swinbank 1963, Satterlund 1979), the determination of $\varepsilon_{lw}$ for a cloudy sky remains quite complex (e.g. Crawford and Duchon 1999, Iziomon et al. 2003, Duarte et al. 2006, Staiger and Matzarakis 2010).

#### 4.3.5.4 Mean Radiant Temperature

The mean radiant temperature $T_{mrt}$ (°C) is one of the most important input parameters for all sophisticated thermal indices applied in human-biometeorology (compare to section 4.3.6). It is an equivalent surface temperature, that summarizes the effect of all the different short- and longwave radiation fluxes (e.g. Fanger 1972, Jendritzky et al. 1990, Kantor and Unger 2011, Chen et al. 2014). $T_{mrt}$ is defined as the surface temperature of a perfect black and equal surrounding environment, which leads to the same energy balance as the current environment (VDI-Kommission Reinhaltung der Luft 1988, Helbig et al. 1999, VDI 2008, Fanger 1972).

**$T_{mrt}$ calculations based on measurements** $T_{mrt}$ can not be measured directly. However, there are different methods to calculate $T_{mrt}$ from several measured parameters (e.g. Thorsson et al. 2007, Kantor and Unger 2011).

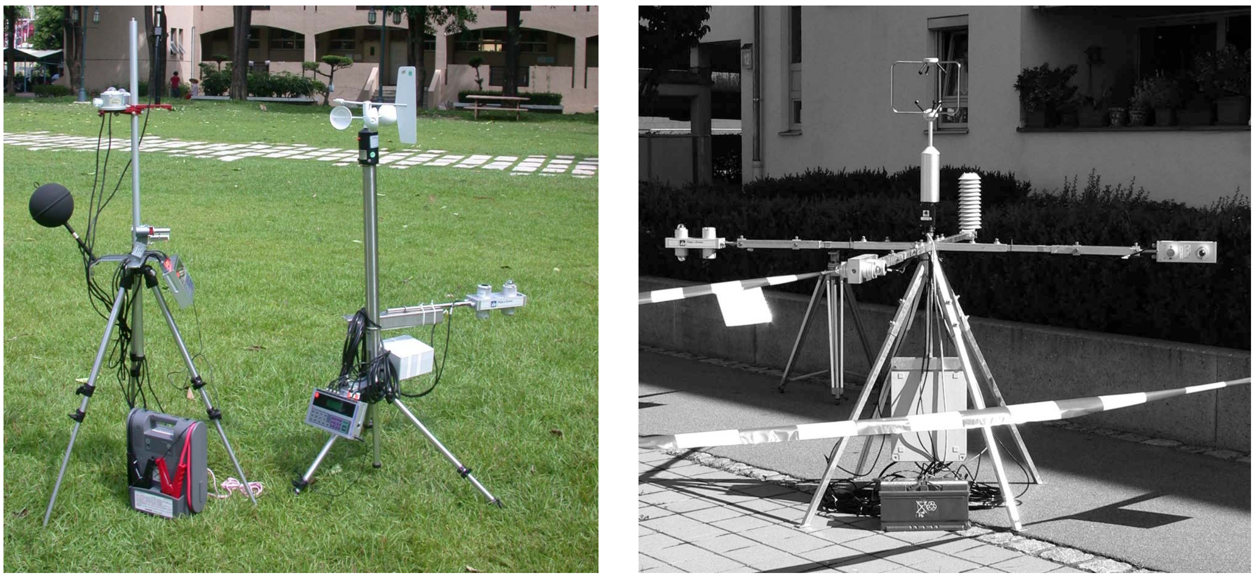

**(a)** Station with globe-thermometer (Lin et al. 2010b). **(b)** Six-directions measurements (Mayer et al. 2008).

**fig. 4.12:** Examples for attempts to retrieve $T_{mrt}$ through measurements. Station with a globe thermometer (left, the black sphere attached to the left station, Lin et al. 2010b) and an urban biometeorological station with six pyranometers and six pyrgeometers for the application of the six-directions method (right, Mayer et al. 2008).

**Globe-thermometer method** A very simple way to calculate $T_{mrt}$ based on measurements is the globe-thermometer method after ISO (1998). It is a very old method (Vernom 1932) originally developed for the application in indoor settings only (e.g. Kantor and Unger 2011). It was later

extended to be applied in outdoor conditions (Clarke and Bach 1971).
The approach is generally based on a black metal sphere with a thermometer in its center. The black metal sphere will absorb all shortwave radiation. This will heat up the sphere, that will now emit longwave radiation according to the Stefan-Boltzmann law (eq. 4.76). The longwave radiation emitted to the inside will then influence the thermometer, that is now recording a globe temperature $T_g$ (°C). $T_{mrt}$ can be calculated from $T_g$ according to eq. 4.78 (ISO 1998, Thorsson et al. 2007).

$$T_{mrt} = \left[ (T_g + 273.15)^4 + \frac{1.1 \cdot 10^8 \cdot v^{0.6}}{\varepsilon_g \cdot diam_g} \cdot (T_g - T_a) \right]^{0.25} - 273.15 \tag{4.78}$$

$\varepsilon_g$ is the globe's absorption coefficient, while $diam_g$ names its diameter in meters.
The globe-thermometer method is applied regularly for it is simple and only requires rather cheap equipment (e.g. Ndetto and Matzarakis 2015). However, it also bears strong limitations due to insufficient thermal conductivity of the globe's material (and, thus, a non-isothermal globe) and insufficient wind speed correction (e.g. Lin and Matzarakis 2011, Chen et al. 2014). Thorsson et al. (2007) also found, that the globe thermometer, dependent on the globe's diameter, needs quite a long time to adapt. Chen et al. (2014) therefore concludes, that it can only be applied for indoor settings, where $v$ (m/s) is constant.

**Six-directions method** Another method of determining $T_{mrt}$ based on measurements is the six-directions method (VDI 1994). The idea behind the approach is to measure the short- and longwave radiation fluxes in six directions (North, East, South, West, up and down) as well as $T_a$. All the Pyranometers and Pyrgeometers applied are considered to consider radiation from within a hemisphere (full 180°) perpendicular to the receptor plate. This leads to the consideration of radiation from all possible directions, thus, of a full three dimensional radiation field. Based on this records, the mean radiation flux density of a person $p$, $\overline{P_h}$ ($\frac{W}{m^2}$) can be calculated by eq. 4.79 (Höppe 1992, VDI 1994).

$$\overline{P_h} = \alpha_{abs,p} \cdot \Sigma_{i=1}^{6} P_{sw,p} \cdot Pr_i + \varepsilon_{lw,p} \cdot \Sigma_{i=1}^{6} P_{lw,p} \cdot Pr_i \tag{4.79}$$

In eq. 4.79, $\alpha_{abs,p}$ is the shortwave absorption coefficient of a person $p$ of approx 0.7 (Höppe 1984, VDI 2008, dimensionless), $i$ is one of the six directions, $P_{sw,p}$ and $P_{lw,p}$ are the short- and longwave radiation flux densities (all in $\frac{W}{m^2}$) measured in direction $i$, while $Pr_i$ is the projection factor for the direction $i$. $Pr_i$ is specified as 0.22 for the horizontal and 0.06 for the vertical directions for a standing person by Fanger (1972), Jendritzky et al. (1990). Based on $\overline{P_h}$, $T_{mrt}$ can be calculated by eq. 4.80 according to VDI (1994).

$$T_{mrt} = \sqrt[4]{\frac{\overline{P_h}}{\varepsilon_{lw,p} \cdot \sigma}} - 273.15 \tag{4.80}$$

$\sigma$ in eq. 4.80 is the Stefan-Boltzmann constant ($5.67 \cdot 10^{-8} \frac{W}{m^2 \cdot T_s^4}$). The main advantage of the six-directions method is that the applied instruments are rather fast and insensitive to wind speed (Thorsson et al. 2007). The main disadvantage is the complex and expensive equipment required for this method (Krueger et al. 2014). It is also very much dependent on the accuracy of the weighting factors (e.g. Chen et al. 2014).

**$T_{mrt}$ calculation by small-scale modelling** In most occasions, $T_{mrt}$ is calculated by small scale models (e.g. Bruse 1999, Matzarakis 2001, Matzarakis et al. 2007; 2010, Lindberg et al. 2008, compare to sections 4.4.1 to 4.4.3). Calculations are generally based on the Stefan-Boltzmann law (eq. 4.76, compare to section 4.3.5.3).
By including the shortwave radiation flux density $P_{sw,s}$ ($\frac{W}{m^2}$) calculated by the diffuse solar irradiation and the diffuse reflected global radiation $D_s$ ($\frac{W}{m^2}$) multiplied by the shortwave absorption coefficient (1.0 - Albedo, $\alpha_{abs,s}$) into eq. 4.77, the total radiation flux density to and from a surface, $P_s$ ($\frac{W}{m^2}$), e.g. the human body can be calculated by eq. 4.81.

$$P_s = \sigma \cdot A_s \cdot T_s^4 + \alpha_{abs,s} \cdot D_s \tag{4.81}$$

Dividing the environment of a person $p$ into a number $n$ of isothermal surfaces $i$ and considering a projection factor $Pr$ to correct the relative surface size of $p$ and $s$ as well as the clothing $clo$ of $p$, $T_{mrt}$ in $\frac{W}{m^2}$ can be calculated following the principle of equal radiation fluxes caused by the actual and the reference environment (eq. 4.82).

$$\varepsilon_{lw,p} \cdot \sigma \cdot \left(T_{mrt}^4 - T_{cl}^4\right) = \Sigma_{i=1}^{n} \left(\varepsilon_{lw,p} \cdot \varepsilon_{lw,i} \cdot \sigma \cdot T_{s,i}^4 + \alpha_{abs,s,i} \cdot D_{s,i}\right) \cdot Pr_{p,i} - \varepsilon_{lw,p} \cdot \sigma \cdot T_{cl}^4 \tag{4.82}$$

Solving eq. 4.82 for $T_{mrt}$ results in eq. 4.83, that is perfectly applicable by numerical micro scale models.

$$T_{mrt} = \left[\Sigma_{i=1}^{n} \left(\varepsilon_{lw,i} \cdot T_{s,i}^4 + \frac{\alpha_{abs,s,i} \cdot D_{s,i}}{\varepsilon_{lw,p} \cdot \sigma}\right) \cdot Pr_{p,i}\right]^{0.25} \tag{4.83}$$

Most numerical models in urban biometeorology, however, are using further simplifications (e.g. Matzarakis et al. 2007; 2010, Bruse and Fleer 1998, Lindberg et al. 2008, refer to section 4.4).

### 4.3.6 Thermal Indices

Human beings do not have any sensor to measure meteorological parameters like the air temperature, but can only feel the integral effect by their skin temperature and their blood temperature in the thermoregulatory system within the hypothalamus (p. 58ff in Tromp 1980, Höppe 1993a). Thermal perception of humans, thus, is based on a huge quantity of parameters (Tromp 1980) and therefore

can not be described through single meteorological parameters, e.g. air temperature ($T_a$) (Höppe 1993a, Błażejczyk et al. 2012, Jendritzky et al. 2012). To approximate human thermal perception, thermal indices have been developed. The more sophisticated ones follow the approach of an equivalent temperature and are based on human energy balance, or heat flux models (e.g. Fanger 1972, Gagge et al. 1986, Höppe 1993a, Błażejczyk et al. 2012).
Three thermal indices, hat are applied in this study are described in the following paragraphs in more detail. Other indices, that are commonly used in the field of urban biometeorology are Fanger's predicted mean vote (PMV, Fanger 1972), the standard effective temperature* (SET*, Gagge et al. 1986) or the new modified equivalent temperature (mPET, Chen and Matzarakis 2014). They all are calculated based on the meteorological input parameters air temperature ($T_a$, °C), relative air humidity (RH, %) or vapour pressure (VP, hPa), wind speed (v, m/s) and the mean radiant temperature ($T_{mrt}$ in °C, refer to section 4.3.5.4). Except from UTCI, they can also deal with different physiological parameters, e.g. hight and weight of the human, its energy production due to activity or its posture (e.g. standing or sitting Watson and Johnson 1988). The indices applied in this study are using °C as unit, facilitating interpretation by people with little knowledge in the field of human-biometeorology (e.g. Höppe 1999).

#### 4.3.6.1 Perceived Temperature

The Perceived Temperature (PT) is an equivalent temperature based on the "Klima-Michel-Model" (Jendritzky et al. 1990), an energy balance model for humans (Staiger et al. 2012). It is designed for people staying outdoors and is defined as "the air temperature of a reference environment in which the thermal perception would be the same as in the actual environment" (Staiger et al. 2012).

**tab. 4.3:** The thermo-physiological meaning of PT results for central Europe as defined by VDI (2008), Staiger et al. (2012).

| PT (°C) | Thermal Perception | Thermo-physiological stress |
|---|---|---|
| $\geq$ +38 | Very hot | Extreme heat stress |
| +32 – +38 | Hot | Great heat stress |
| +26 – +32 | Warm | Moderate heat stress |
| +20 – +26 | Slightly warm | Slight heat stress |
| 0 – +20 | Comfortable | Comfort possible |
| -13 – 0 | Slightly cool | Slight cold stress |
| -26 – -13 | Cool | Moderate cold stress |
| -39 – -26 | Cold | Great cold stress |
| $<$ -39 | Very cold | Extreme cold stress |

By design, PT is a steady-state model to avoid costly iterations. It is calculated for a sample human

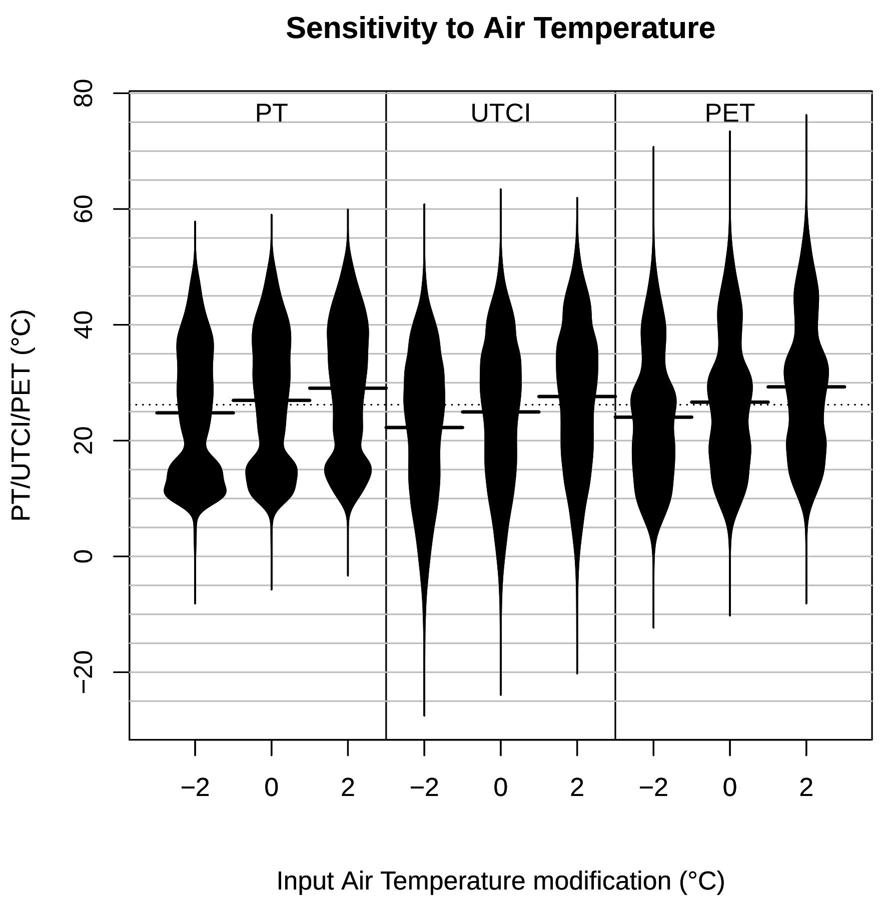


**fig. 4.13:** Distribution of PT before the update (left), UTCI (center) and PET (right) calculated with modified air temperature ($T_a$) for the hot and dry conditions in Doha (Fröhlich and Matzarakis 2016). Meteorological input data is covering the period between March 1999 and January 2014 in three hours resolution. For each index, the distribution of the results is plotted as three beans (Kampstra 2008) for a dataset with $T_a$ reduced by 2 °C (left), the original dataset (center) and a dataset with $T_a$ increased by 2 °C (right).

being (the "Klima-Michel", Jendritzky et al. 1990) with a height of 1,75 m, an age of 35 years, a weight of 75 kg, an internal heat production of 135 W/m² and a walking speed of 4 km/h (Staiger et al. 2012). Assuming this, a simplified human heat balance equation after ASHRAE (2001, p. 134) is applied (eq. 4.84).

$$M - Wo = (C + R + E_{sk}) + (C_{res} + E_{res}) + S_{sk} + S_{cr} \tag{4.84}$$

It compares the energy gain by the metabolic heat production $M$ reduced by the portion of mechanical work $Wo$ to the total heat flux from or to the environment, that is represented by the flux

of sensible heat $C$, radiation $R$, and latent heat $E$. Eq. 4.84 distinguishes between fluxes from or to the skin $_{sk}$, the core $_{cr}$ and via the respiratory system $_{res}$. The heat storage $S$ can be neglected assuming a steady state. All parameters unit is $W$.
As the physiological parameters are fixed and the clothing model is self-adapting, PT can be calculated using the meteorological parameters $T_a$ (°C), $v$ (m/s), $VP$ (hPa), and $T_{mrt}$ (°C) only. Energy gained or lost by the sample person is compared to that of the reference environment. This is defined with parameters $T_{mrt} = T_a$, v = 0.1 m/s and VP equal to VP of the actual environment. If the actual environment would lead to warm and humid conditions, VP is set to a value matching a relative humidity of 50 % (Staiger et al. 2012).
The heat flux from or to the reference environment is compared to a simple two-node (Skin and Core) body model. Heat is exchanged by Skin and Respiration (Staiger et al. 2012). The skin heat transfer includes sensible heat fluxes due to convection and radiation, as well as latent heat fluxes due to sweating (Staiger et al. 2012). The raspiratory heat transfer considers sensible heat, as well as the latent heat flux by the inhaled air (Staiger et al. 2012).
PT contains a clothing model, that is automatically choosing the most appropriate value for the clothing index (clo) according to the prevailing meteorological conditions (Staiger et al. 2012). It automatically tries to maintain thermal comfort. If this can not be achieved, thermal stress is computed. The thermal assessment itself is based on a modification of the Predicted Mean Vote (PMV) index (Fanger 1972, Staiger et al. 2012). It is enhanced by parametrizations of shivering in cold conditions (PMV < -0.11 at clo = 1.75) and sweating, for hot conditions (PMV > 0.5 at clo = 0.5, Staiger et al. 2012). VDI (2008) and Staiger et al. (2012) also published a table allowing for the interpretation of PT results in central Europe (tab. 4.3).
The perceived temperature currently is mainly applied in studies with contribution of the German Meteorological Service (DWD, e.g. 변 et al. 2008b, Schoetter et al. 2013), but is enjoying increasing popularity also in other parts of the world (e.g. in South Korea, 변 et al. 2008a, 이 et al. 2010).
PT (calculated by the RayMan model, refer to section 4.4.1) was found to suffer from some shortcoming in the clothing model, that leads to strangely distributed results (e.g. Fröhlich and Matzarakis 2016, see fig. 4.13). In the course of the dissertation project, the author received an updated version of PT that improves the results (compare old (left part of the beans, light grey) and the updated version of PT (right part of the beans, dark grey) in fig. 4.14). Fig. 4.14 shows that there is some improvement in the lower part of the beans, representing cooler conditions. The overall distribution, however, only shows slight improvement and stays very uneven.

#### 4.3.6.2 Universal Thermal Climate Index

UTCI (Universal Thermal Climate Index) is defined as "the isothermal air temperature of the reference condition that would elicit the same dynamic response (strain) of the physiological model" than the

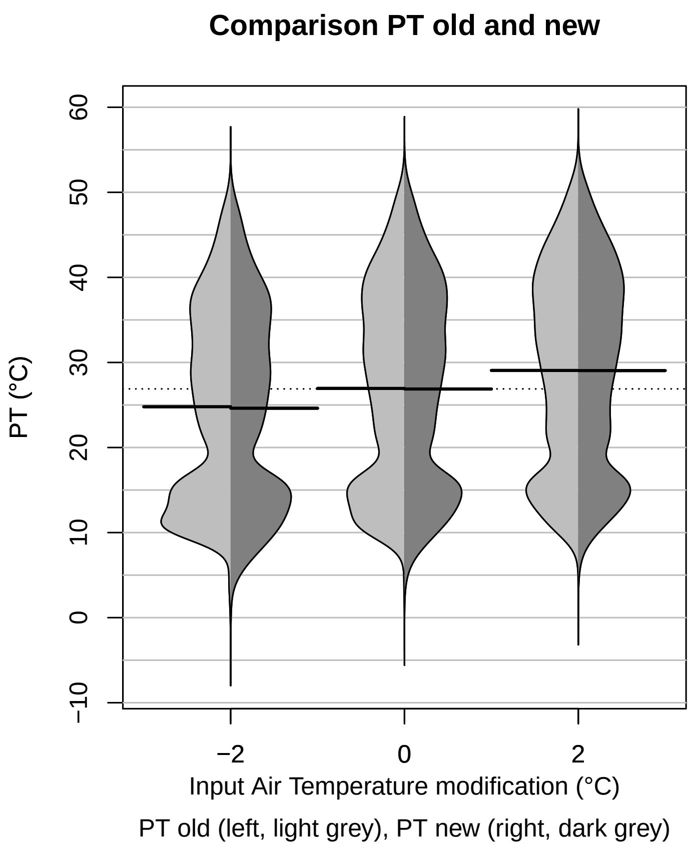


**fig. 4.14:** Distribution of PT before the update (left part of the beans, light grey), and the updated PT (right part of the beans, dark grey) calculated with modified air temperature ($T_a$) for very same dataset as fig. 4.13. The three beans (Kampstra 2008) show the distribution of the results for the dataset with $T_a$ reduced by 2 °C (left), the original dataset (center) and the dataset with $T_a$ increased by 2 °C (right).

actual environment (Jendritzky et al. 2012).

UTCI also follows the concept of an equivalent temperature. The meteorological conditions are compared to a reference environment with 50 % relative humidity, calm air and $T_{mrt}$ being equal to $T_a$ (Jendritzky et al. 2012). The comparison is performed on basis of a heat transfer model (Fiala et al. 2012).

In contrast to other indices, physiological parameters can not be set in UTCI. Besides the self-adapting clothing insulation, a permanent walking speed of 4 km/h (1.11 m/s) and an internal heat production of 135 W/m² are assumed (Jendritzky et al. 2012). UTCI includes a clothing model that automatically adapts to the current conditions (Havenith et al. 2012).

UTCI is not, like PT, or PET, calculated directly. Due to its extremely high complexity, and thus, high computational effort, UTCI can not be calculated for most studies. It can only be approximated using a regression formula, that was abbreviated from sample calculations performed by computing centres (Jendritzky et al. 2012).

The regression function makes the calculation of UTCI computationally cheap, but it also leads to a very narrow range for the input parameters it accepts. Only the meteorological parameters $T_a$ (°C), VP (hPa), $v$ (m/s), and $T_{mrt}$ (°C) can be set. All physiological parameters are considered to be determined automatically. Limitations for the applicability due to the restriction for $T_a$ of -50.0°C to +50.0°C, as well as for the valid range of wind speed ranging from 0.5 m/s to 17.0 m/s, are to be expected. This can lead to a tendency in the results as especially heat stress conditions with high $T_a$ and low $v$ are omitted (Fröhlich and Matzarakis 2016). This can be also seen comparing the three beans representing the UTCI results in the central part of fig. 4.13. The increase in $T_a$ by 2°C does not seem to increase UTCI (Fröhlich and Matzarakis 2016). Other uncertainty is to be expected due to vapour pressure of the reference environment, which is limited to 20.0 hPa (Jendritzky et al. 2012).

Within the accepted range, UTCI is very sensitive to wind speed (Chen and Matzarakis 2014, Fröhlich and Matzarakis 2016). Besides $T_a$, also $T_{mrt}$ strongly influences UTCI (Chen and Matzarakis 2014, Fröhlich and Matzarakis 2016).

**tab. 4.4:** Thermal stress classification for UTCI. Modified after Błażejczyk et al. (2013).

| UTCI (°C) | Thermal Stress category |
|---|---|
| $\geq$ +46 | Extreme heat stress |
| +38 – +46 | Very strong heat stress |
| +32 – +38 | Strong heat stress |
| +26 – +32 | Moderate heat stress |
| +9 – +26 | No thermal stress |
| 0 – +9 | Slight cold stress |
| -13 – 0 | Moderate cold stress |
| -27 – -13 | Strong cold stress |
| -40 – -27 | Very strong cold stress |
| $<$ -40 | Extreme cold stress |

For Central Europe, Błażejczyk et al. (2013) published a thermal stress classification to facilitate the interpretation of UTCI results (tab. 4.4). In contrast to the assessment tables for PT (tab. 4.3) and PET (e.g. tab. 4.6), that are focussing on thermal comfort, the UTCI assessment table is a thermal stress classification (Błażejczyk et al. 2013).

### 4.3.6.3 Physiologically Equivalent Temperature

A regularly used index for the assessment of human thermal comfort is the Physiological Equivalent Temperature (PET). It is defined as "the air temperature at which, in a typical indoor setting (without wind and solar radiation), the energy budget of the human body is balanced with the same core and skin temperature as under the complex outdoor conditions to be assessed" (Mayer and Höppe 1987, Höppe 1999, Matzarakis et al. 1999). PET is based on a simplification of the human energy balance model "Munich Energy Balance Model for Individuals" (MEMI, Höppe 1984). In contrast to PT and UTCI, PET does not use a self adapting clothing model. It is therefore free of behavioural components and, thus, "a real climatic index describing the thermal environment in a thermo-physiologically weighted way" (Höppe 1999).
One of the most important determining factors of PET is the mean radiant temperature $T_{mrt}$ (°C, Herrmann and Matzarakis 2012, Charalampopoulos et al. 2013, Chen and Matzarakis 2014). Other important meteorological input parameters for PET are wind speed $v$ (m/s, compare to tab. 4.5) and $T_a$ (°C). Air humidity (vapour pressure VP (hPa), as well as relative humidity RH (%)) only shows very weak impact on PET (Chen and Matzarakis 2014, Fröhlich and Matzarakis 2016).

**tab. 4.5:** Exemplary calculations to demonstrate the influence of wind speed on PET. All results (PET, lower right part of the table) are in °C. Calculations were performed assuming a constant relative humidity (RH) of 60 %, air temperature ($T_a$) constantly equal to the mean radiant temperature ($T_{mrt}$) and wind speed (v) according to the values on the right hand side of the table header. Physiological parameters were also considered constant with an internal heat production of 80 W and a heat transfer resistance of the clothing of 0.9 clo.

| | $v$ (m/s) | | | | |
|---|---|---|---|---|---|
| $T_a$ = $T_{mrt}$ (°C) | 0,5 | 1,0 | 3,0 | 5,0 | 10,0 |
| 0.0 | -2.9 | -4.3 | -6.4 | -7.2 | -8.1 |
| 20.0 | 18.3 | 17.2 | 15.4 | 14.6 | 13.8 |
| 30.0 | 29.9 | 29.4 | 28.4 | 27.7 | 26.7 |

The thermal impact of the actual environment in PET is assessed through a human energy balance equation (eq. 4.85, Höppe 1999).

$$M + Wo + R + C + E_{sk} + E_{res} + E_{sw} + S = 0 \tag{4.85}$$

It consists of the metabolic heat production $M$, mechanical work $Wo$, the fluxes of radiation $R$, sensible heat $C$, and latent heat $E$. In eq. 4.85 $E$ is divided into fluxes from or to the skin $_{sk}$, through sweating $_{sw}$ and via the respiratory system $_{res}$. All parameters mentioned above are in $W$. The heat

storage $S$ is assumed to equal 0 $W$ at any time (assuming a steady state).
The actual environment is transferred to an virtual indoor environment with $T_{mrt} = T_a$, $v$ = 0.1 m/s, and VP = 12 hPa (Höppe 1999). $T_a$ is varied for the indoor environment iteratively until the effect on the human energy balance is the same as for the actual environment. The resulting indoor $T_a$ equals PET (Höppe 1999).

**tab. 4.6:** Thermal sensation classes for human beings in Central Europe (with an internal heat production of 80 W and a heat transfer resistance of the clothing of 0.9 clo (clothing value)) modified after Matzarakis and Mayer (1996).

| PET (°C) | Thermal Perception | Grade of physical stress |
|---|---|---|
| $>$ 41 | Very hot | Extreme heat stress |
| 35 – 41 | Hot | Strong heat stress |
| 29 – 35 | Warm | Moderate heat stress |
| 23 – 29 | Slightly warm | Slight heat stress |
| 18 – 23 | Comfortable | No thermal stress |
| 13 – 18 | Slightly cool | Slight cold stress |
| 8 – 13 | Cool | Moderate cold stress |
| 4 – 8 | Cold | Strong cold stress |
| $\leq$ 4 | Very cold | Extreme cold stress |

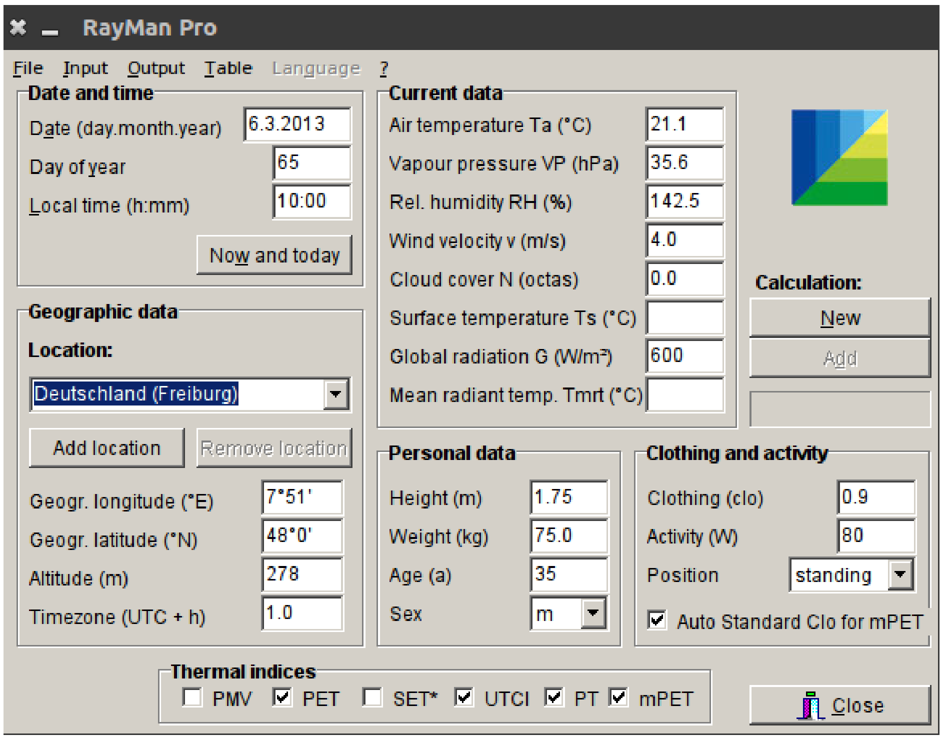
To facilitate the interpretation of PET results, they can be classified using classification tables for the region in question. For Central Europe, PET results were classified into nine classes of thermal perception (tab. 4.6) by Matzarakis and Mayer (1996). For other regions there are local classifications available, that have been adapted to the prevailing climate conditions (e.g. Taiwan: Lin et al. 2013).
PET is currently one of the most commonly used indices for human thermal comfort (e.g. in Matzarakis and Mayer (1996), Matzarakis et al. (1999; 2009), Lin et al. (2010a), Muthers et al. (2010), Lopes et al. (2011), Hwang et al. (2011), Nastos and Matzarakis (2012), Lin et al. (2013), Ketterer et al. (2013), Fröhlich and Matzarakis (2013)).

## 4.4 Numerical Models

In the recent years, a number of numerical models were developed and applied. In the field of urban human biometeorology, however, only few models exist. The most relevant ones for this study are presented in this section.

### 4.4.1 RayMan

RayMan is a micro-scale model developed at the Chair for Environmental Meteorology, former Chair for Meteorology and Climatology of the Albert-Ludwigs-University Freiburg to calculate radiation fluxes in simple and complex environments (Matzarakis et al. 2007; 2010). This allows the calculation of $T_{mrt}$ (compare to section 4.3.5.4), which is an important input parameter in the calculation of thermal biometeorological indices like (updated) PT, UTCI and PET (section 4.3.6).
RayMan is one dimensional in space (all the calculations are performed for one point). It follows



Date and time
Date (day.month.year): 6.3.2013
Day of year: 65
Local time (h:mm): 10:00

Geographic data
Location: Deutschland (Freiburg)
Geogr. longitude (°E): 7°51'
Geogr. latitude (°N): 48°0'
Altitude (m): 278
Timezone (UTC + h): 1.0

Current data
Air temperature Ta (°C): 21.1
Vapour pressure VP (hPa): 35.6
Rel. humidity RH (%): 142.5
Wind velocity v (m/s): 4.0
Cloud cover N (octas): 0.0
Surface temperature Ts (°C):
Global radiation G (W/m²): 600
Mean radiant temp. Tmrt (°C):

Personal data
Height (m): 1.75
Weight (kg): 75.0
Age (a): 35
Sex: m

Clothing and activity
Clothing (clo): 0.9
Activity (W): 80
Position: standing
☑ Auto Standard Clo for mPET

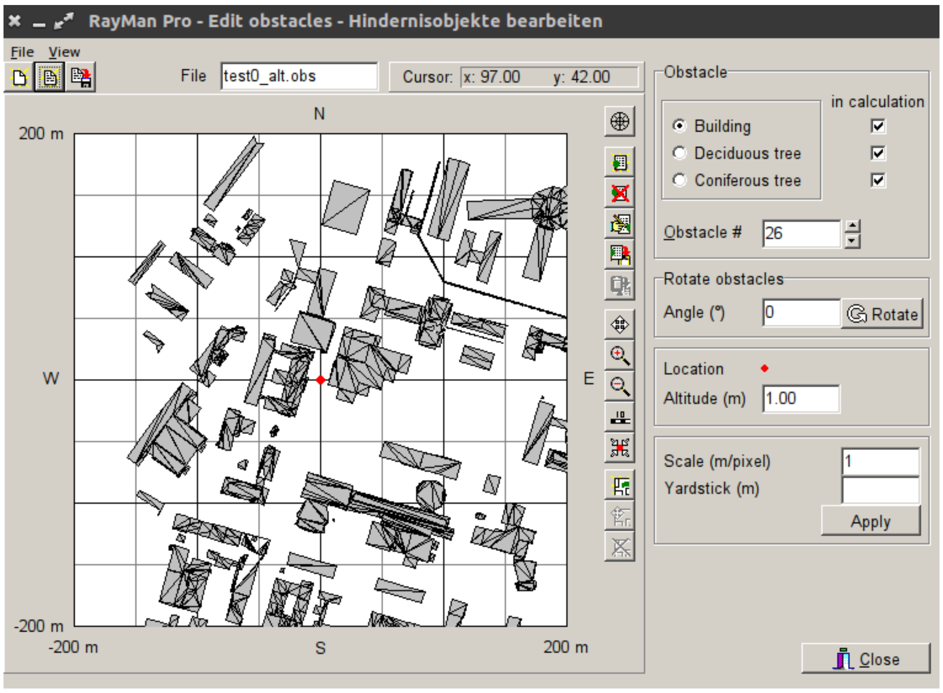
Thermal indices
☐ PMV ☑ PET ☐ SET* ☑ UTCI ☑ PT ☑ mPET

**fig. 4.15:** RayMan: Screenshot of the main window.

the diagnostic approach to be time-independent. The focus during the development was layed on usability (all functions and settings can be controlled through the graphical user interface (GUI, compare to fig. 4.15)) and performance. The latter was to enable the user to run calculations for long datasets covering several years in high temporal resolution (e.g. Fröhlich and Matzarakis 2013). Another principle of RayMan is to only require a limited number of meteorological input, and only common parameters (Matzarakis et al. 2010).

**SVF** One of the main features of RayMan is the determination of SVF (compare to section 4.3.5.1). This can be done based on Fish-eye images (compare to section 4.3.5.1), free drawing of the

horizon limitation, a topographic raster, or an obstacle dataset (compare to fig. 4.16, Matzarakis et al. 2007). RayMan obstacle files are a special type of spatial vector files based on semicolon delimited text files, that can be created manually using the RayMan obstacles editor (fig. 4.16) or automatically based on shapefiles by using the Quantum GIS (QGIS Development Team 2016) plugin "Shp to Obs".
For either of the input possibilities a binary Fish-eye graph is rendered holding the values 1 for free sky and 0 for obstructed pixels. Using this, SVF can be calculated according to section 4.3.5.1.



File: test0_alt.obs
Cursor: x: 97.00 y: 42.00

[✓ Building] ✓
Deciduous tree ✓
Coniferous tree ✓

Obstacle # 26
Angle (°) 0
Location ◆
Altitude (m) 1.00
Scale (m/pixel) 1
Yardstick (m)

fig. 4.16: RayMan: Screenshot of the obstacles input window.

**Global radiation** Global radiation $G$ ($\frac{W}{m^2}$) can either be specified in the GUI as a fixed input parameter, it can be part of a meteorological datafile (refer to section 6.7), or it can be calculated from time, date, geographic position and a cloud cover observation in octas. From these parameters, an initial global radiation $G_0$ ($\frac{W}{m^2}$) is calculated (refer to eq. 4.67, Jendritzky et al. 1990), that is later corrected by SVF and shading.
If a global radiation measurement is provided by the user, it could be set to be considered as the local global radiation $G$ recorded at the very location of interest, or to be considered as a background measurement $G_0$. In the latter case, it will be corrected by the local SVF and shading prior to any further calculations (eqs. (4.68) and (4.72) to (4.75)).

**Mean radiant temperature** The mean radiant temperature $T_{mrt}$ (°C) can be specified by the user. In this case, the value will be considered valid for the location of interest and it will be used for further calculations without any correction. $T_{mrt}$ can also be calculated by RayMan. This is done based on meteorological parameters and the location of interest (compare to section 4.3.5.4).

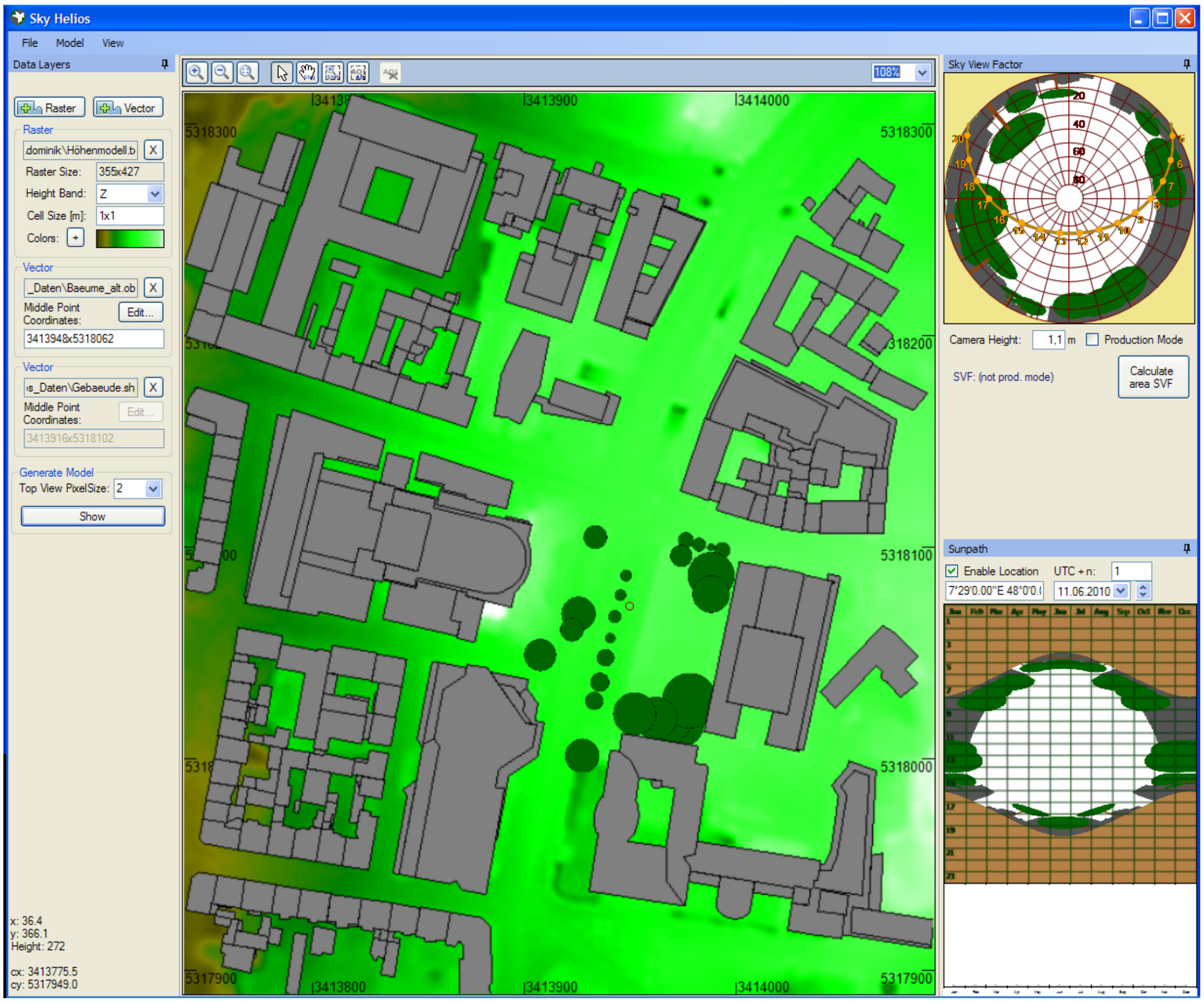
RayMan is one of the most successful models in urban biometeorology and is applied in many studies all around the world (e.g. Charalampopoulos et al. 2013, Abreu-Harbich et al. 2014, Ndetto and Matzarakis 2015, Yang and Matzarakis 2016). Several validation studies attest RayMan good accuracy in approximating $T_{mrt}$ and PET (e.g. Matzarakis et al. 2007, Hwang et al. 2011). The models main advantage is the low computational effort, as well as the good usability. The most severe shortcomings of the RayMan model is the absence of a wind model, as well as the limitation to one point of interest.

### 4.4.2 SkyHelios model

Initially, SkyHelios (compare to fig. 4.17) was a model for the rapid estimation of Sky View Factor (SVF, section 4.3.5.1) and the sunshine duration (Matzarakis and Matuschek 2011). In contrast to other models used in urban climatology, it applies the OGER graphics engine, developed for video games, to create a three dimensional environment from the spatial input data in order to calculate scientific data based on that. This new approach allows faster calculations, as well as calculations using rather cheap hardware (Matzarakis and Matuschek 2011).

**Spatial input options** The SkyHelios model accepts a rather wide range of spatial input formats. These are to be divided into raster and vector formats (compare to fig. 4.18). Raster formats consist of an equidistant grid with some value (e.g. elevation) for every grid cell. SkyHelios accepts various common raster file formats through including the "Geospatial Data Abstraction Library" (GDAL, GDAL Development Team 2016). Vector formats are based on specifying the spatial position of certain points (vertices). Several vertices can form polygons, e.g. buildings. The most common vector file format are shapefiles. These, besides many others, are accepted by the SkyHelios model through OGR (OpenGIS Simple Features Reference Implementation), that is part of GDAL (GDAL Development Team 2016). Another important vector file format that can be used as spatial input is RayMan obstacle files (see section 4.4.1).

**SVF** SkyHelios is able to calculate SVF using 2 different ways of weighting according to eqs. (4.65) and (4.66) (to provide spheric and planar SVF) for different purposes (Hämmerle et al. 2011a;b). The calculations for both, planar and spheric SVF, are based on a fast and simple method. It first



dominik\Höhenmodell.b
Raster Size: 355x427
Height Band: Z
Cell Size [m]: 1x1

_Daten\Baeume_alt.ob
Middle Point Coordinates: 3413948x5318062

is_Daten\Gebaeude.sh
Middle Point Coordinates: 3413916x5318102

Top View PixelSize: 2

x: 36.4
y: 366.1
Height: 272

cx: 3413775.5
cy: 5317949.0

108%

Camera Height: 1,1 m
Production Mode
SVF: (not prod. mode)

Enable Location
UTC + n: 1
7°29'0.00"E 48°0'0.0
11.06.2010

**fig. 4.17:** SkyHeios: Screenshot of the old main window.

distinguishes between white and coloured pixel of a generated Fish-eye image. The white ones are then counted.

**Shading** The SkyHelios model is also able to display the shadows of obstacles using the graphics engines shadow algorithm. Shadow calculations can be done for regular grids based on a ray-casting algorithm. A ray is sent from the grid cell's center to the sun's position. If an obstacle is hit in between, the cell is considered to be shaded.

SkyHelios is a fast and easy to use model, but currently only has very limited functionality. The model is intended to be used in urban human-biometeorological studies in the future. However, for the spatial calculation of biometeorological indices like PET or UTCI, also routines calculating the

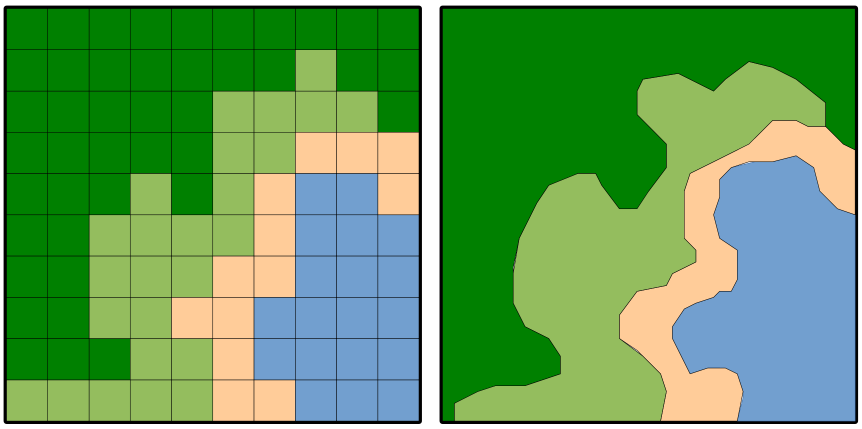
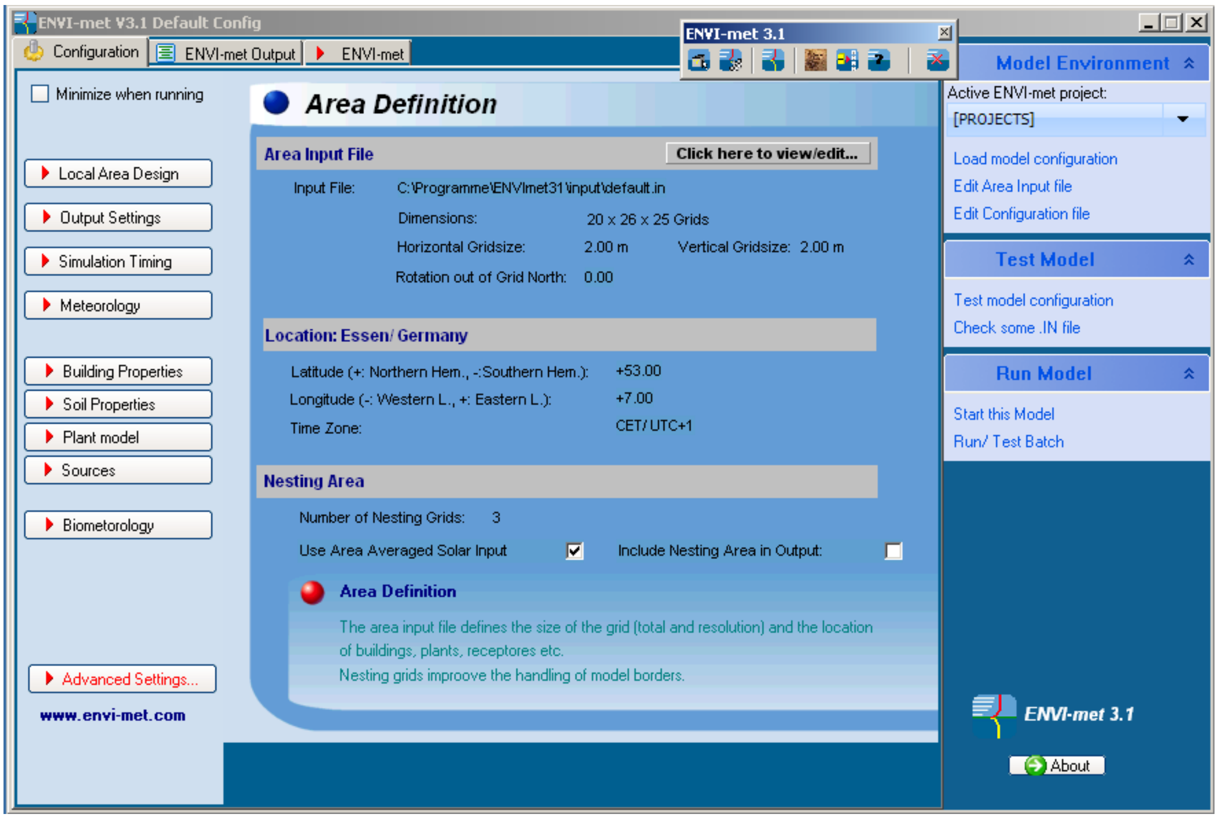


**fig. 4.18:** Sketch of a sample beach area represented as regular raster (left) and vector based polygons (right). In both graphs the sea is blue, the beach is brown, meadows are light green and forest is coloured dark green. Both graphs are showing the same area to show the differences between raster and vector formats.

different radiation fluxes, as well as a model calculating the spatial distribution of wind speed have to be implemented. While parts of a radiation model are already being integrated, a wind model was implemented in the course of this study.

### 4.4.3 ENVI-met

One of the most commonly used prognostic models in the field of urban biometeorology is the ENVI-met model (Bruse 1999, Bruse and Fleer 1998, Huttner 2012, compare to fig. 4.19). Its last official beta release, the version 3.1 is free to use. There are different more recent versions (e.g. 3.5, 4.0 and 4.1) that are not released officially, but used in some studies (e.g. Fröhlich and Matzarakis 2011).
ENVI-met consists of four model parts: a soil model, a vegetation model, and an atmosphere model, that are driven by a 1D background model (compare to fig. 4.20). As the atmosphere model is the most relevant part of ENVI-met concerning this study, this model part will be described in more detail than the other ones.
The 1D background model mainly consists of vertical gradients calculated from the meteorological input parameters and considered to be static in time. Except for few parameters ($T_a$ and $VP$ in version 3.5), the 1D model does not change during a model run. Its main purpose is to provide the 3D model core (atmosphere model) with border conditions. They are representing the general state of the atmosphere at a given altitude. The 1D model covers a vertical range from 0 m to 2500 m above ground (Bruse 1999, Huttner 2012).
The 1D background model forces the three dimensional model core. This includes the atmosphere



Area Input File

Input File: C:\Programme\ENVImet31\input\default.in

Dimensions: 20 x 26 x 25 Grids

Horizontal Gridsize: 2.00 m Vertical Gridsize: 2.00 m

Rotation out of Grid North: 0.00

Location: Essen/ Germany

Latitude (+: Northern Hem., -:Southern Hem.): +53.00

Longitude (-: Western L., +: Eastern L.): +7.00

Time Zone: CET/ UTC+1

Nesting Area

Number of Nesting Grids: 3

Use Area Averaged Solar Input ✓ Include Nesting Area in Output: ☐

Area Definition

The area input file defines the size of the grid (total and resolution) and the location of buildings, plants, receptores etc.
Nesting grids improove the handling of model borders.

Active ENVI-met project:
[PROJECTS]

**fig. 4.19:** ENVI-met: Screenshot of the main window (Version 3.1).

and the vegetation model and interacts with a soil model (Bruse 1999, Huttner 2012).

The 3D atmosphere model is physically considered to be located inside the 1D model. It is horizontally split up into up to 250 discrete cells in x-, as well as in y-direction. The applied parametrizations thereby allow a constant cell spacing ranging from 1 m up to about 20 m (Bruse 1999). Vertically up to 50 discrete cells of varying resolution my be defined. This enables the increase of vertical cell spacing with height that reduces the total number of grid points required to cover the model area and, thus, the required computation time. The most lowest layer of cells is split up into four layers of sub-cells to facilitate consideration of processes at the lower model border (Bruse 1999, Huttner 2012).

The ENVI-met atmosphere model, as described in Bruse (1999), is a prognostic three dimensional model for calculations at the micro scale. It numerically approximates the three dimensional incompressible Navier-Stokes equations in the non-hydrostatic form. The air pressure term in the Navier-Stokes equations was therefore eliminated using the Boussinesq-Approximation (Boussinesq 1897). Time integration is done applying the simple Euler-Forward scheme.

The vegetation model is part of the 3D model core. It calculates simplified plant physiology as well as temperature, radiation and humidity fluxes. Up to version 3.5, all adjacent cells within the ENVI-met

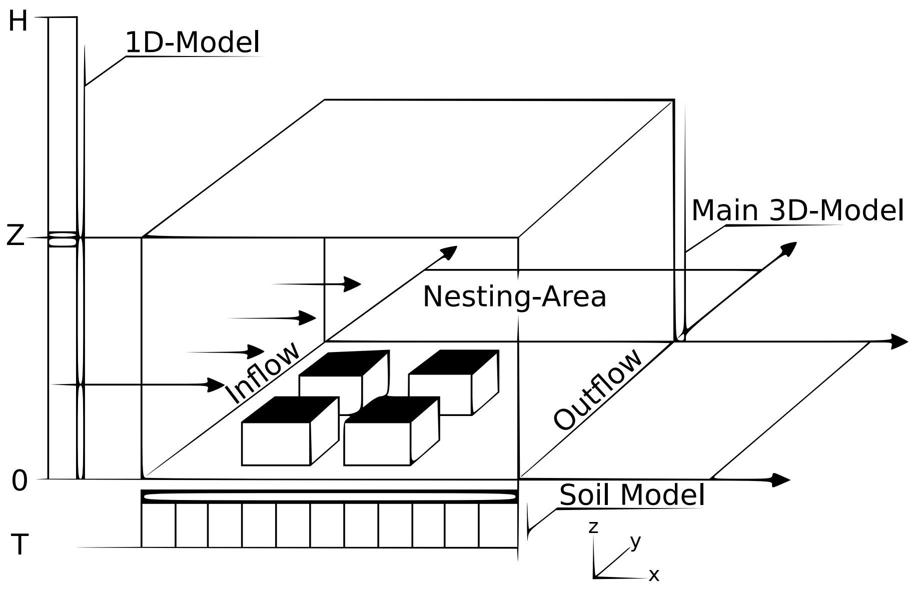


**fig. 4.20:** Overview over the ENVI-met model parts. Modified after Bruse (1999). The main 3D-Model consists of the vegetations model and the atmosphere model.

model area marked as the same plant type are treated as one plant with common physiological states (Bruse and Fleer 1998, Huttner 2012). ENVI-met version 4.0 and higher is able to distinguish individual plants.
The soil model calculates all processes below the ground surface. This mainly concerns water and heat transport. The horizontal resolution matches that of the model core. The vertical extension is fix (-1.75 m to 0.0 m). The soil model also calculates energy conversions at building walls and provides the model core with internal border conditions (e.g. surface temperature and evaporation Bruse 1999, Huttner 2012, Yang et al. 2013).
The ENVI-met model is commonly applied in the field of urban biometeorology (e.g. Fröhlich and Matzarakis 2011; 2013, Yang et al. 2013). However, it has some shortcomings that need to be considered. Through being a prognostic model, ENVI-met is computationally costly. Running a single model day in ENVI-met can take up to several days computational time. This leads to only rather short case studies can be performed (e.g. Fröhlich and Matzarakis 2013). Also no savepoints are created leading to the need to re-run the whole model in case of an error. E.g. stagnation in low wind speed conditions can easily crash the model after several hours computational time with a divide by 0 exception. The model can not resume from there, but needs to be re-run right from the start. For technical reasons (delphi application, 32 bit) ENVI-met input areas are limited to a maximum of 255 on 255 on 40 grid cells (Bruse 1999). This limits the size of model areas or requires a rather low model resolution of several meters, leading to inaccuracy in the results (e.g. for $T_{mrt}$, Chen et al. 2014).

# 5 Approach of this study

In the field of urban biometeorology, there are only few models available, that are allowing for the estimation of thermal indices (see section 4.3.6) considering the urban environment, e.g. RayMan (section 4.4.1) and ENVI-met (section 4.4.3). All of the pre-existing models have both, advantages and shortcomings. While RayMan is rather fast, vector based and supports all important thermal indices (section 4.3.6), it is limited to one individual point of interest and does not provide a wind model. The ENVI-met model supports spatial calculations, but can only calculate some thermal indices (PMV, starting from version 4.0 also PET) based on a raster model area in rather low resolution and requires rather long time to run.

Therefore, non of the models available is able to calculate thermal indices for larger areas in sufficient resolution. To meet these needs, the development of a new model is required.

For not having to start from scratch, the implementation of new modules into the SkyHelios model (section 4.4.2) was considered the most promising approach. The SkyHelios model was selected for being developed following the idea of fast calculations and simple operability. It has a simple graphical user interface (GUI) and accepts a very broad range of spatial input data through including the GDAL library (refer to section 4.4.2). SkyHelios furthermore already contained parts of the necessary functionality. As it was developed at the Chair for Environmental Meteorology Freiburg (former Chair for Meteorology and Climatology Freiburg, former Meteorological Institute Freiburg), it was available as source code in Visual C# language. Visual C# allows for the integration of unmanaged Dynamically Linked Libraries (dll), and therefore the extension of model parts by both, Visual C# and unmanaged C++ code.

To enable SkyHelios calculating thermal indices spatially, the model needs to be able to provide all input variables for any point within the model area. As the spatial variation of $T_a$ and $RH$ / $VP$ within rather short distances is usually very small, they can be considered constant within the model area. While the error due to this simplification is considered to be quite small, this does not hold for parameters with strong spatial variations like $T_{mrt}$ and $v$. SkyHelios therefore is to be extended by two model parts: a radiation model part and a wind model part. As parts of a radiation model already exist in SkyHelios, this only needs to be completed. For the wind model, in contrast, a brave new implementation was necessary.

Before the implementation of new model parts, the main model needs some redesign. The internal treatment of spatial input data is different for all the different file types supported and rather

unsystematic. This leads to compatibility issues as soon as several spatial input files are loaded. To avoid this issues, the model part loading and processing the spatial input needs to be mostly rewritten. The new version is intended to be able to load several spatial input files considering differing geographic projections to consider each of them in the right relative orientation.
The SkyHelios model currently displays input data and calculates the sky view factor based on the MOGRE graphics engine in 32 bit architecture. This is sufficient for very small input areas, but leads to huge inaccuracy and display errors if larger input areas are used. The MOGRE engine therefore needs to be compiled in 64 bit architecture and implemented into the SkyHelios model.
The result of any spatial calculation performed by SkyHelios is currently written to a text file. For more sophisticated calculations, SkyHelios is to be extended by a results manager, that holds the results of the previous calculations and is able to provide them as input for other model parts. This also avoids double calculations decreasing the computational effort.
Run-time is considered one of the most important criteria for modelling in urban-biometeorology, as long data series are very common in this field for statistical reasons. Following the idea that a diagnostic model can run way faster than a prognostic one, a diagnostic wind model developed in a previous study using the statistical environment R (R Development Core Team 2008) is modified and compiled into a C++ dll to be integrated into the SkyHelios model.
The diagnostic wind model can deliver three dimensional wind data for a lot of independent sets of input data without losses in performance. This is important if several different cases (e.g. several heat waves with some years of time in between) or data with very large timesteps is used (e.g. daily means, or data for 13:00 every day). These datasets are quite common in urban biometeorology (e.g. Matzarakis et al. 2007, Fröhlich and Matzarakis 2013). A prognostic model would have to initialize a new wind field for every set of data and iterate until the wind field was stable, what would result in unacceptable long computation time. As a diagnostic wind model is time independent, it calculates a new wind field for every set of data anyway. It is, thus, way more suitable for cases like the ones mentioned above.
The model integrated into SkyHelios is designed after the work of Röckle (1990). As proposed by Röckle (1990), it consists of modules to calculate luv-side stagnation zones, lee-side recirculations, lee-side velocity deficit zones and, if there are obstacles in a specific configuration, vortices (compare to section 4.3.4.3). The sum of all the wind field deformations is used as an initial wind field containing lots of divergence, that has to be reduces numerically. As the parametrization described in Röckle (1990) is partly outdated, it is replaced by new parametrizations allowing for better accuracy as proposed by e.g. Singh et al. (2008). The modular design of the wind model is intended to facilitate the replacement of single parametrizations by updated ones as soon as they are becoming available.
Apart from the wind model, also a radiation model is required for a fully applicable model to be used in urban biometeorlology. To complete the radiation model, the methodology already applied in the RayMan model (see radiation part in section 4.4.1) was used. This methodology was selected

for having proven to provide quite precise results in a very short time (e.g. Matzarakis et al. 2007, Hwang et al. 2011).
The radiation model is able to calculate $T_{mrt}$ based on two different combinations of input parameters, that need to be supplied by the main model. One option is the calculation based on an initial global radiation calculated from cloud cover, geographic position and the true local time. This is corrected by the influence of the model area (e.g. SVF and sun visible (true or false)) to obtain the local global radiation at the point of interest. The second option is the determination of the local global radiation based on an input global radiation provided by a meteorological datafile (refer to section 6.7). The initial global radiation provided is considered to be recorded in rooftop level (without any influence of the area of interest). It is, thus, corrected by the influence of the model area at the given point of interest to obtain the local global radiation. The different options were implemented to ensure a maximum of flexibility and applicability.
Due to its vector-based implementation, the radiation model is able to determine $T_{mrt}$ for any point within the model area. If raster data is required, $T_{mrt}$ is calculated for the center of any cell. Together with the flexibility in the input parameters, this keeps the radiation model in line with the main developing paradigms of simplicity and usability.
A module calculating the perceived temperature (PT, see section 4.3.6.1) will be developed and implemented into the SkyHelios model. Using the results provided by the wind model and the radiation model, SkyHelios will then be able to calculate PT and, thus, to be used for spatial thermal comfort analysis within urban areas.

# 6 Method

In the course of this dissertation project the SkyHelios model (section 4.4.2) was improved and enhanced by several features. The most important improvements are listed below.

6.1 Implementation of a wind model:

- A wind model based on the diagnostic approach by Röckle (1990), that was developed during a previous study was ported to C++ and compiled into a dynamic link library (DLL).
- The wind model was improved by implementing recent parametrizations.
- The improved wind model was linked to SkyHelios. The graphical user interface (GUI) was enhanced by settings for wind calculations. The SkyHelios backend was extended to supply the wind model with input data and to trigger the calculations.

6.2 A module calculating thermal indices, e.g. the Perceived Temperature (section 4.3.6.1) was developed and included.

6.3 The three dimensional part of the model was restructured and partly redesigned to allow for the consideration of several spatial input files in correct relative orientation, as well as for larger model areas.

6.4 The internal organisation of spatial input data was reorganized and partly rewritten for better reliability and enhanced performance.

6.5 Support for a meteorological data input file was included.

6.6 The new functionality was tested for two study areas in Freiburg and compared to measured data from a previous study.

## 6.1 Implementation of a diagnostic wind model

The wind model implemented during this study is based on a previously developed diagnostic wind model running on R (R Development Core Team 2008). R scripts are very suitable for development and testing of new models, as any variables can be checked at any time. Furthermore R takes

care of memory management and type casting by itself avoiding many hard to debug programming flaws. On the other hand R scripts are running very slow, as they are interpreted line by line. This slows down computation a lot when it comes to iterations. R scripts are also hard to use by external programs as the exchange of variables is limited. For productive use, the model therefore needed to be translated to a pre-compiled language. It was ported to C++ for best performance and packaged into a dynamic link library (dll) for compatibility with the SkyHelios model.
The wind model is structured in small modules that are called by a main container method, as well as by each other. The main method is accessible through the dlls interface. This architecture was found to provide maximum compatibility. The operation of the individual modules, as well al the major improvements is described in detail in the following subsections.

### 6.1.1 Construction of an obstacle list

The calling application passes a three dimensional grid of obstacles to the wind model. To consider the obstacles contained by this grid during the calculation of the initial wind field, a special table containing different parameters of the obstacles is needed. This table is created by a module named "SepBuild". The function also splits obstacles of complex shapes into many rectangular obstacles to facilitate the calculations. A U-shaped building of equal height, for example, will be split up into three single, rectangular obstacles. Their impact on the initial wind field will be calculated separately.
The functionality of the "SepBuild" module can be described as follows: It iterates over every column, row, and level of the given three dimensional model area grid. The grid contains a transmissivity factor for every cell that may vary from 0 (air may flow though this cell without any resistance) to 1 (the cell contains a solid obstacle). As soon, as a cell with a transmissivity index larger than 0 is found, a check is run to obtain the obstacles lower, left corner, as well as its extension in all directions. All these values for one obstacle are stored into a common container, a C++ std::vector<double>. It contains the items described in tab. 6.1.

Now the obstacles list, a vector of vectors with obstacle data (std::vector< std::vector<double> >), is checked. If the obstacle is found to not already be part of the obstacles list, and not to be totally contained by another obstacle with larger or equal porosity factor, it is added to this table.
To identify all the obstacles, the whole three dimensional input area grid needs to be searched. As this process is rather time consuming, computations are done in parallel by different threads starting from different y-coordinates. This increases computation speed by almost the number of available CPU-cores.
The obstacles table always stays the same for one model area. To avoid unnecessary computational effort and to safe computation time, the obstacles table is written to a temporary file after it is calculated for the first time. If the wind model is called again for the same area, the obstacles data is read from the temporary file instead of recalculating. The calling application on the other hand

**tab. 6.1:** The content of a vector containing information about one obstacle. Seen from left to right, the columns show the position inside the vector, the parameters name, the unit (if any) and a short description of the parameter.

| Position | Parameter | Unit | Description |
|---|---|---|---|
| 0 | Id | | obstacle number |
| 1 | startx | cells | smallest x position in grid |
| 2 | starty | cells | smallest y position in grid |
| 3 | bottom | cells | smallest z position in grid |
| 4 | x-length | m | obstacles extension in x direction |
| 5 | y-width | m | obstacles extension in y direction |
| 6 | height | m | obstacles extension in z direction |
| 7 | transmissivity | | transmissivity of the obstacles (see above) |
| 8 | LR | m | maximum recirculation length (will stay empty until written during recirculation calculations, see section 6.1.4) |

now has to make sure the temporary file is deleted as soon as a new model area is present. In SkyHelios, this is done when the "Show" button is pressed.

### 6.1.2 Initialisation of velocity grids

The module calculating a vertical profile and initializing the three velocity grids is called "SetWD". The module first creates a vertical reference profile from the user input passed by the calling application. In a former version the logarithmic velocity profile according to eq. 4.2 was used. As this was found to be insufficient, a modified urban canopy profile (compare to section 4.3.1.1, Macdonald 2000) was implemented.
It is first applied to approximate the vertical profile for the station providing the wind data. Therefore, the sensor height, the roughness length $z_0$ (m), the displacement height $d$ (m), wind speed $u_{ref}$ (m/s) and wind direction $WD$ (°) at the station are required.
For a most accurate estimation of the wind field, the wind model supports spatially resolved $z_0$ and $d$. Therefore, two grids defining $z_0$ and $d$ for any two dimensional position (x/y) inside the model area is to be passed by the calling application. If both grids are present, the wind model uses the station profile to estimate wind speed and direction at the upper model border. Now for any x and y an individual vertical profile is calculated according to the local conditions (compare to fig. 6.1).
According to the individual local profiles, the module "SetWD" will set up three three dimensional arrays to store the velocity fields in the x, y, an z direction. All of them contain one additional set of cells in the flow direction, as they are addressed as half-grids (staggered grids, refer to section 4.3.4.3). The grid for the flow in x direction (called "ugrid"), and the grid for the velocity in

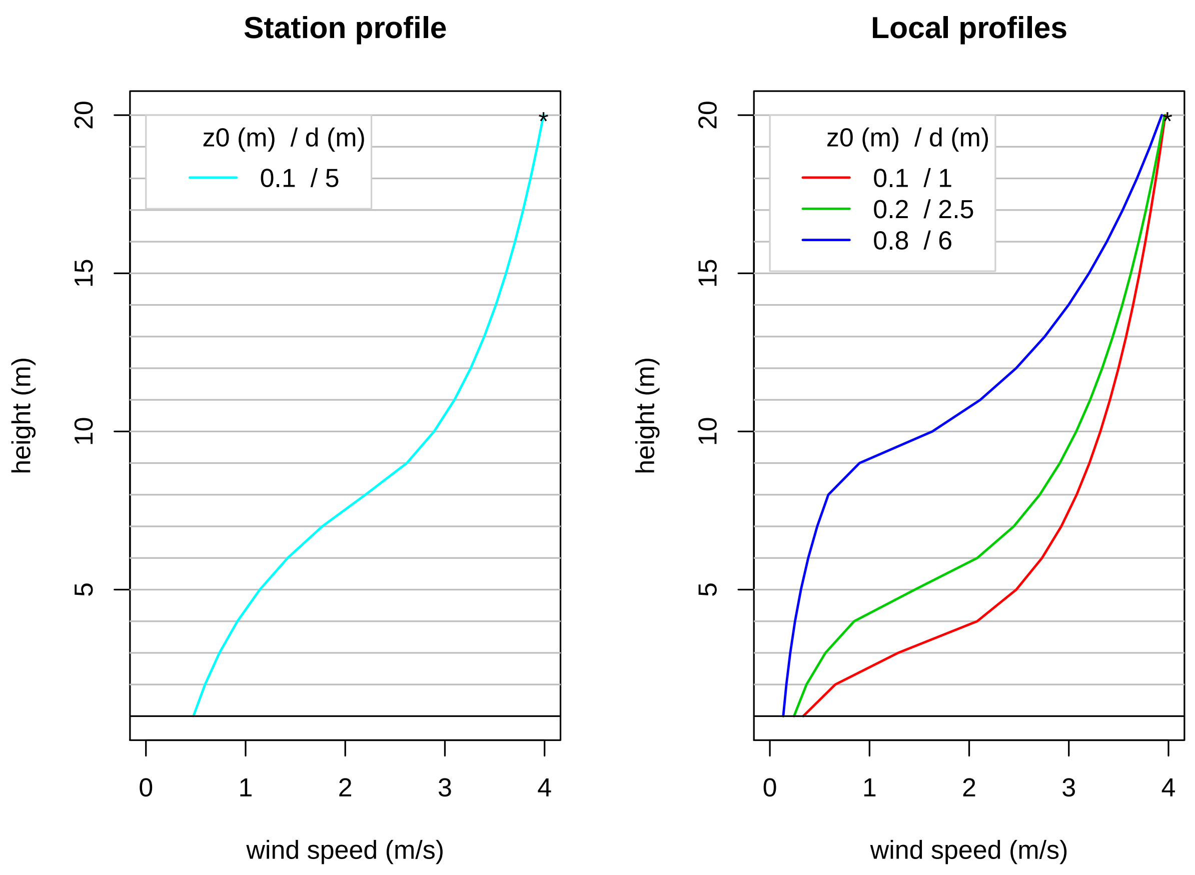


**fig. 6.1:** Example for the determination of local vertical wind speed profiles: First a station profile is calculated from sensor height, roughness length $z_0$ (m), displacement height $d$ (m), wind speed $u_{ref}$ (m/s) and wind direction $WD$ ($^\circ$) at the stations location (light blue, left). This is used to determine wind speed at the model top (*). * can now be used to calculate local vertical profiles (red, green and blue, right) with different $z_0$ and $d$.

y direction (named "vgrid") are filled with an initial wind speed according to the calculated local profiles. Therefore wind speed has to be split up to its x- and y-component using eq. 4.26. The initial z-component stored in "wgrid" is initialized with 0.0 m/s at any position.

### 6.1.3 Calculation of the windward stagnation zone

To calculate the stagnation zone in wind direction of any obstacle, the module named "FrontEddyBagal" is called (for the code of the module refer to section 10.1.1). It first determines the position and size of the windwards stagnation zone (compare to fig. 6.2) and then writes a wind field modification

after Bagal et al. (2004).
In a first step, the module sets up three temporary arrays to store the position and wind field

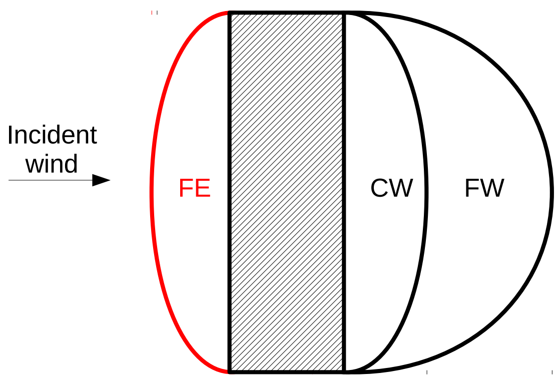


**fig. 6.2:** Outline indicating the position of the windwards stagnation zone (front eddy zone (FE), red) relative to an obstacle. The black rectangle filled with diagonal lines represents a solid, rectangular obstacle. Incident wind is assumed perpendicular from the left.

modifications of the front eddy zones for all the obstacles. One of them, called "u_temp" matches the size of ugrid. The other two, named "v_temp" and "w_temp" those of v- and wgrid accordingly (compare to section 6.1.2). Alike all the flux grids used in this model, they are of type float (single precision) and are initialized with a fail value of 9999.0. Calculations would be possible to perform using flux grids in double precision, but numbers are quite small and memory space is limited. Single precision therefore was considered to be most adequate.
The main program part is organized as a loop iterating over the internal table of obstacles (tab. 6.1). This ensures all the obstacle are considered one after the other.
For any obstacle, the four semi-axis ($a_x$ and $a_y$ for the front eddy zone in x- and y-direction) for the ellipse-shaped front eddy zones (in x- and y-direction) is calculated (compare to tab. 4.1). At the same time, the dimensions of a vortex inside the front eddy zone are determined according to eq. 4.47. The vortex is also assumed to be half-ellipse shaped extending windwards from the obstacles wall.
In the next step a check on the wind direction is run. If wind direction ranges in between 0 and 180 degrees, a front eddy zone for an incident flow from the +x direction is calculated. If wind direction is in between 90 and 270 degrees, a front eddy zone for an incident flow from the -y direction is calculated. The other two possible ranges of incident wind direction (-x and +y direction) are treated respectively. This makes sure that, except for perpendicular incident flow, always two front eddy zones are calculated per obstacle. One for the x-, and one for the y-component.
For every wind direction to be considered, the width and length of the obstacle are determined. For the basic shape of the front eddy zone is a half ellipse, its center point has to be calculated. Using

the half axis and the center point, a check is run whether a specific point is inside the front eddy ellipse, or not. For not having to check whether all of the grid points in the model domain belong to the front eddy zone, a check-range is calculated to select only points that could theoretically be inside the front eddy ellipse for the specific wind direction. These points are selected by creating a three dimensional bounding box using the length of the two half axis and the obstacles position and dimensions to consider the ellipses maximum possible extension. Based on the bounding box a check-range is set. E.g. for incident wind from -x direction, the x-check-range would be in between startx (refer to tab. 6.1) plus x-length of the obstacle, and startx plus x-length plus the previously calculated semi-axis in x-direction.
Inside the calculated x-, y- and z-range, every point is tested to be inside, or outside the ellipse of the front eddy zone. Therefore the basic equation for the construction of an ellipse is used in a slightly modified form (eq. 6.1).

$$\frac{(x - centerx)}{a_x} + \frac{(y - centery)}{a_y} <= 1 \tag{6.1}$$

The modifications done to the equation are to correct the position of the center of the ellipse that is not equal to the coordinate systems origin. Furthermore the result does not have to be equal to one, but also smaller, as the point may be part of the outline of the ellipse, or inside of it, to be considered as part of the front eddy zone.
All the points that are found to be inside the front eddy zone are tested to be part of the inner vortex zone using the same methodology, but with a smaller half-axis in wind direction, calculated according to eq. 4.49. For all points found to be inside the inner vortex zone, a wind field modification is calculated according to eqs. (4.50) and (4.51). For points inside the front eddy zone, but outside the inner vortex zone, the air flow perpendicular to the obstacles wall is set to be 0.0 m/s. The air flow parallel to the obstacles wall will remain unchanged. The front eddy zone for an incident flow from $270°$ will therefore only show up in the "u_temp" grid, while the inner vortex zone will show up in the u-, v-, and "w_temp" grids.
Finally u-, v-, and "w_temp" will be returned as the result of this module. For the sake of performance, they will need to stay in the systems main memory until the initial wind field is assembled by the "AssembleInitialGrid" module (see section 6.1.7).

### 6.1.4 Position and initial conditions of lee-side recirculation

In the lee of an obstacle the air flow disattaches from the obstacles lee-side roof edge and will not re-attach to the ground until some point further windward (compare to fig. 4.7). The space in between is experiencing a wind direction rather opposite to the incident wind direction, that will be strongest close to the obstacle and close to the ground. It forms a recirculation on the lee-side

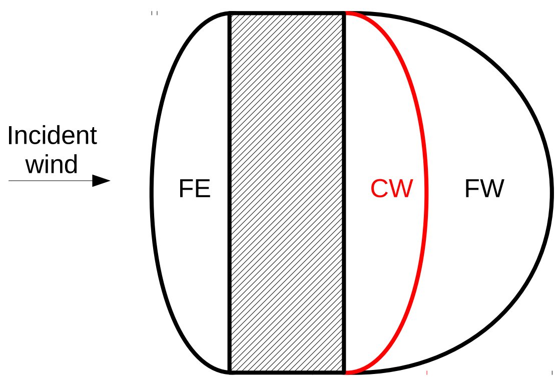


**fig. 6.3:** Outline indicating the position of the lee-side recirculation zone (close wake zone (CW), red) relative to an obstacle. The black rectangle filled with diagonal lines represents a solid, rectangular obstacle. Incident wind is assumed perpendicular from the left.

of an obstacle (Hosker 1985). To determine the modifications by a lee-side recirculation to the initial wind field, the module "CloseWake3" is called (see section 10.1.2 for the code). The module calculates size and position (compare to fig. 6.3), as well as the wind field modification caused by a recirculation on the lee-side of an obstacle. Together with the module "FarWake2", that is described in section 6.1.5 it calculate the wake zone of an obstacle. The "CloseWake3" module creates a half-ellipse shaped zone at the lee side of an obstacle, within which the air current is set in opposite direction to the air current at rooftop level, decreasing to zero along height and increasing distance to the obstacle.
To achieve this, two new temporary arrays are created to temporarily store the results. One of them, "u_CW", matches the size of ugrid, the other one, "v_CW" matches the dimensions of vgrid. As the module "CloseWake3" not only calculates the position, but also the modifications, the temporary arrays have to be of type float to match u- and vgrid. They are initialized with a fail value of 999.0 m/s at all cells.
The module distinguishes two different conditions: Incident flow perpendicular to the obstacles and non-perpendicular incident flow. As the obstacles are represented by a regular, Cartesian grid, incident wind from 90°, 180°, 270°, or 360° is considered perpendicular. Any other incident wind direction represents non-perpendicular incident wind.
The influence of the close wake is, analogue to the front eddy, calculated for one obstacles after the other. The basic shape of the close wake zone is, still in agreement to the front eddy zone, elliptic. First, the module therefore needs to determine the length of the ellipses first half axis ($a_x$, or $a_y$, depending on incident wind direction).
As for a ground based obstacle, the recirculation is strongest in ground level and gets weaker (and, thus, shorter) with height $z$, $a_x$ or $a_y$ need to be calculated for each level separately. This is done on

basis of their maximum possible length $L_R$.
Eq. 4.28 is found to overestimate the maximum length of the recirculation (or the most largest distance between the obstacle and the point of flow re-attachment to the ground) $L_R$ severely in case of an unfavourable combination of obstacles width $wi$, length $l$ and height $h$ (all in m). Furthermore, eq. 4.28 in some cases calculates decreasing $L_R$ with increasing $h$, what must be considered implausible. It therefore was modified and extended by a border statement scaling its maximum size and fixing the inverse dependence on $h$ (eq. 6.2).

$$L_R = \frac{1.8 \cdot \frac{wi}{h_{LR}}}{\left(\frac{l}{h_{LR}}\right)^{0.3} \cdot \left(1 + 0.24 \cdot \frac{wi}{h_{LR}}\right)} \cdot h_{LR} \cdot (1 - Por) \tag{6.2}$$

Eq. 6.2 includes the obstacles porosity (dimensionless factor) and the height scaling factor $h_{LR}$ (dimensionless), that is calculated according to eq. 6.3.

$$h_{LR} = 5.0 \cdot (1.0 - exp(-0.2 \cdot h)) \tag{6.3}$$

Eq. 6.2 is found to calculate more stable values for $L_R$ than eq. 4.28 avoiding huge $L_R$ in spite of small $w$ (compare to fig. 6.4). The new equation takes into consideration, that at some point of increasing obstacle height the air will only flow around the obstacle and $L_R$, thus, must not increase any further. In the example shown by fig. 6.4, eq. 6.2 limits $L_R$ to 7.26 m even for an obstacle height of 50 m. At the same point eq. 4.28 calculates 17.54 m.
$L_R$ is finally stored within the obstacles parameters (compare to tab. 6.1) to stay available for the modules "FarWake2" (section 6.1.5) and "StreetCanyonSingh" (section 6.1.6). Using $L_R$, the half axis parallel to the air flow $a_x$, or $a_y$ can be determined using a slightly modified version of eq. 4.59 proposed by Pardyjak et al. (2004, eq. 6.4). The second half axis length is set according to the half width of the obstacle.

$$a_x = L_R \cdot (1 - Por) \cdot \sqrt{1.0 - \left(\frac{z}{h}\right)^2} - \frac{l}{2} \tag{6.4}$$

Also other parameters apart from $a_x$, or $a_y$, e. g. the center of the close wake zone ellipse, that needs to be set to the corresponding side of the obstacle are determined according to the incident wind direction.
To limit the number of points to be considered in the further calculations and, thus, to save computation time, a x- and y-range is calculated where to check for the close wake zone of this obstacle. Also a z-range is set according to the vertical extension of the obstacle. This is done in the same way as for the front eddy zone. The ranges sometimes contain points that are outside the model domain. This points have to be excluded to avoid errors in the further calculations.

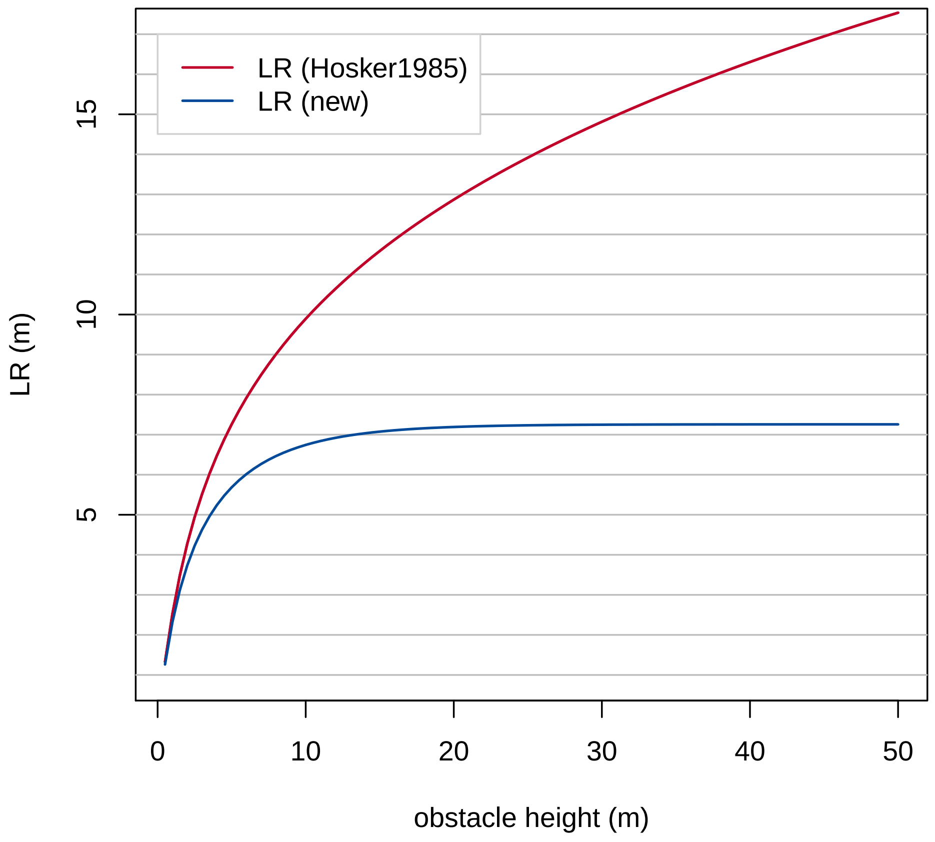


**fig. 6.4:** $L_R$ (m) calculated by eq. 4.28 (red) and eqs. (6.2) and (6.3) (blue) for an obstacle of 5 m width, 5 m length and a height varying from 0.5 m to 50.0 m

For every point inside the x-, y-, and z-range, a test is performed to check whether the point is inside or part of the ellipse, or not. This is done similar to the check performed during the calculations to set the position of the front eddy zone (eq. 6.1).
If a point is found to be part of the close wake zone, its distance to the obstacle ($d_l$), as well as the distance between the obstacle and the end of the close wake zone ($d_W$) is calculated. Using this parameters, the modification for the initial wind field at the certain point is calculated according to eq. 4.30. The modifications are then stored into the corresponding temporary grids.
For non-perpendicular incident flow, the calculations are only slightly more complex. As suggested by Röckle (1990), the position of the close wake, as well as its modification to the initial wind field are calculated using a rotated coordinate system. The close wake is, thus, calculated for a rotated and modified obstacle with perpendicular incident flow from $270°$. The results then are re-projected

and written to the original grid (compare to section 4.3.4.3).
To achieve the rotation, a two dimensional rotation matrix is created according to the incident wind direction. re-projection to the original grid is done using a backwards rotation matrix. For every obstacle, all four corners are saved to an array. Every entry of the array is now multiplied by the rotation matrix and added to the array that now contains four original and four rotated points.
To apply the method described in section 4.3.4.3, the rotated obstacle (that now faces incident flow from -x direction) has to be reshaped in width $wi$ and length $l$ to meet the obstacle width relative to the non-perpendicular incident flow (compare to dashed obstacle in fig. 4.8). To get the relative width, the rotated point with the lowest y-value is subtracted from the rotated point with the highest y-value. The length of the projected obstacle is calculated by dividing the area of the original building by the relative width. Also other parameters defining the obstacle (e.g. start-x and a start-y, refer to tab. 6.1) have to be calculated for the projected obstacle using the rotated points. Besides the ID and the $Por$ also the bottom cell and the vertical extension of the obstacle will stay the same as the rotation is two-dimensional.
Using the variables for the projected building, the half axis of the ellipse are calculated analogue to the calculations for perpendicular incident flow from -x direction. The half axis are both calculated for the projected grid, as all the following calculations are taking place based on the rotated coordinate system. An exception are the center of the ellipse, as well as the upper and lower end of the ellipse. Though they are calculated for the rotated grid, a back-rotated copy is saved to meet the original grid.
Another rather complex calculation needs to be done to set the x- and y-range to limit the points to be considered. To avoid rounding errors, it was found to be of advantage not to calculate the close wake for the grid cells of the rotated coordinate system and project those afterwards. In this case, the influence is calculated for any point of the rotated grid. Afterwards, those points are projected to the original grid and their values are written to the closest point on the original grid. As the rotated and the original grid don't match very well, there will be points on the original grid that receive double, or even triple modification. At the same time there are points that do not match at all and, thus, will not be modified. To avoid this issue, the points to be considered are taken from the original grid directly that have to be projected to the rotated grid first.
As the points are taken from the original grid, also the x- and y-ranges need to be set up on the original grid. They are calculated depending on wind direction using the re-projected upper and lower ellipse ends, as well as the re-projected ellipse center and the half axes calculated on the projected grid. As for perpendicular incident flow, all points that exceed the domain limits will be deleted from the ranges.
For all points inside the z-, y-, and x-range, a projected point is calculated. This projected point is tested to be part of the close wake zone, as described for the perpendicular incident flow. If the point is found to be located inside the close wake, its distance to the obstacle, as well as the obstacles distance to the end of the close wake are calculated on the projected grid. Only if both

are larger than zero, a modification is calculated.
To calculate the modification, first, the wind direction has to be considered again. This is necessary to calculate the modification from the correct cell of the correct speed grid. E.g. for incident flow from within $180°$ to $360°$, the modification has to be calculated from the x+1 cell of ugrid (as ugrid is a staggered grid). Again, the modification will only be written to u_temp, if there is not already a stronger modification by the close wake of another obstacle present at this cell.

### 6.1.5 Position and initial conditions of the velocity deficit zone

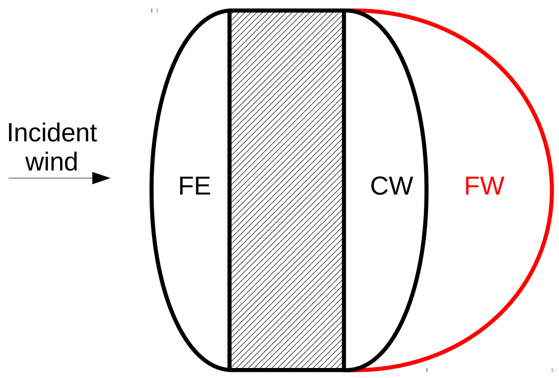


**fig. 6.5:** Outline indicating the position of the lee-side velocity deficit zone (far wake zone (FW), red) relative to an obstacle. The black rectangle filled with diagonal lines represents a solid, rectangular obstacle. Incident wind is assumed perpendicular from the left.

In the lee of an obstacle, behind the recirculation, there is a zone with a wind direction just like the incident wind direction, but a reduced wind speed (refer to fig. 6.5, Röckle 1990). This zone is called lee-side cavity, velocity deficit zone, or far wake. The determination of the position and modifications of the cavity in the lee of an obstacle is quite similar to the method used to calculate the recirculation zone (refer to section 6.1.4). As for the recirculation zone, all the necessary methods are organized in an own module named "FarWake2". Still in agreement to the calculations for the recirculation zone, another two temporary speed grids ("u_temp" and "v_temp") are created to store the modifications to the u- and the vgrid within the velocity deficit zone. Analogue to the the module "CloseWake3" (see section 6.1.4), perpendicular, and non-perpendicular incident flow is distinguished. In both cases (perpendicular and non-perpendicular incident flow), a loop over all the obstacles is run to consider one obstacle after the other. Within the loop, at first, the static parameters of the obstacle are calculated, e.g. the obstacles lower and upper border, the startx and starty coordinates, but also the vertical range.

The width and length of the obstacle are set according to the incident wind direction as described in section 6.1.4. The maximum length of the first half axis $L_R$ is already calculated by the "CloseWake3" module by the time the "FarWake2" module is called and therefore can be loaded from the obstacles properties table (tab. 6.1). The maximum length of the wake zone is then multiplied by three, to set it to be three times the length of the recirculation zone (Röckle 1990). It is then used replacing $L_R$ in (eq. 6.4, modified after Pardyjak et al. 2004) to calculate the length of the velocity deficit zone at any level $z$. The width of the wake zone (the second half axis) is set according to the obstacle's width. According to the incident wind to be in +, or - direction, the center of the ellipse, as well as the x- and y-range are set to the according side of the obstacle. For any point within the ranges, a check is performed, whether the point is part of the wake zone, or not. This is done in the same way as for the recirculation zone. Also the distance of the point to the obstacle, as well as the distance of the obstacle to the end of the wake zone are set in the same way as those, for the recirculation zone. If both of them are larger than zero, a modification is calculated by eq. 4.32 proposed by Röckle (1990).

In the case of non-perpendicular incident flow, the same workaround as for the recirculation zone is applied. The model area is first rotated to a grid with a perpendicular incident flow from -x direction. Then, the modifications are calculated for projected points taken from the original grid. The detailed procedure is described below.

For the rotation of the model domain, a rotation angle is calculated from the initial wind direction. Also a rotation matrix and a backwards rotation matrix are created to facilitate the conversion from one grid to the other.

As everything else has to be done for all the obstacles separately, all steps described below are placed inside a for-loop iterating over the obstacles table. As in the section for non-perpendicular flow in the "CloseWake3" module, the coordinates of the four edges of the obstacle are written into a "points" table. The points are then rotated by multiplying by the rotation matrix to meet the rotated grid and stored at the end of the "points" table that afterwards consists of four original, and four rotated points.

To be able to access them faster, also the obstacles lowest z-coordinate and its height are stored to internal variables. Using this variables, the zrange for this obstacle is set. The width of the projected obstacle is calculated as the y-distance between the (rotated) point with the largest y- and the one with the smallest y-value. The area of the building is calculated, to determine the rotated obstacles length (the length of the obstacles walls parallel to the incident wind) that is calculated from the projected obstacles width and its area. As the obstacle has been rotated, also the startx and the starty value are, most likely, different and need to be determined for the rotated grid. Thereby startx is calculated as the length of the projected obstacle, subtracted from the highest x-value of the rotated points. The new starty is set to be the smallest y-value of the rotated points.

The maximum length of the first half-axis is set as for perpendicular incident flow using the variables calculated before. The maximum length of the first half axis $L_R$ is read from the obstacles properties

(tab. 6.1) and a copy multiplied by three is stored locally. First, it is used to calculate the half axis for all levels in the vertical range of the obstacle applying eq. 6.4 (with $L_R$ multiplied by three). Later it is used during the calculation of the x- and y-ranges. The calculation of the horizontal ranges requires also the position of the center of the ellipse. It is calculated analogue to the one of the ellipse that forms the recirculation zone. As the ranges have to be calculated for the original grid, the coordinates of the ellipse's center have to be reprojected.
After all this variables are set, the modifications are calculated. This is done in a very similar way as the calculations for perpendicular incident flow and those of the recirculation zone: To consider all necessary points three for-loops are nested into each other. The first one is iterating over all values of z in the vertical range, the second one is to consider all y-values, and the third, the inner one, to respect all values in xrange. Except the calculation of a squared value of the two half axis for a specific value of z, and the calculation of a variable storing the current first half axis for the specific z, divided by three and squared afterwards, all calculations are performed inside the inner loop.
Inside the inner loop, at first, the projected coordinates are calculated for the current point. They are then used to check, whether the point is inside the +x part of the ellipse of the wake zone. This is done in the same way as for the front eddy or the recirculation zone, using eq. 6.1.
If a point is found to be inside the positive part of the ellipse, its distance to the projected obstacle, as well as the length of the wake zone for the specific y-value is calculated. After a check, whether both distances are larger than or at least equal to one, the modifications are calculated according to eq. 4.32. They again are only written to the temporary grid, if their absolute value exceeds the value of the same cell on the temporary grid that may result from a previously considered obstacle.

### 6.1.6 Position and modification by street canyons

If obstacles are located in a certain configuration relative to each other, they can not be considered individually (Hosker 1985). If there are two obstacles close after each other in flow direction, there is not enough room for a fully featured wake zone. The air current will then not re-attach to the ground, but will flow over both obstacles in rooftop level causing a vortex in between them (compare to fig. 6.6, Röckle 1990). This will happen, if (viewn in flow direction) two obstacles are located within a distance of the first obstacles recirculation zone and if they are, in crosswind direction, overlapping (Singh et al. 2008).
The modification by a possible street canyon, as well as the detection of its position inside a model domain is performed by the module "StreetCanyonSingh". It is created after the suggestions made in Singh et al. (2008). The module works very different than the modules "FrontEddyBagal", "CloseWake3" and "FarWake2" and differs strongly from the methodology proposed by Röckle (1990), that is found to be insufficient.

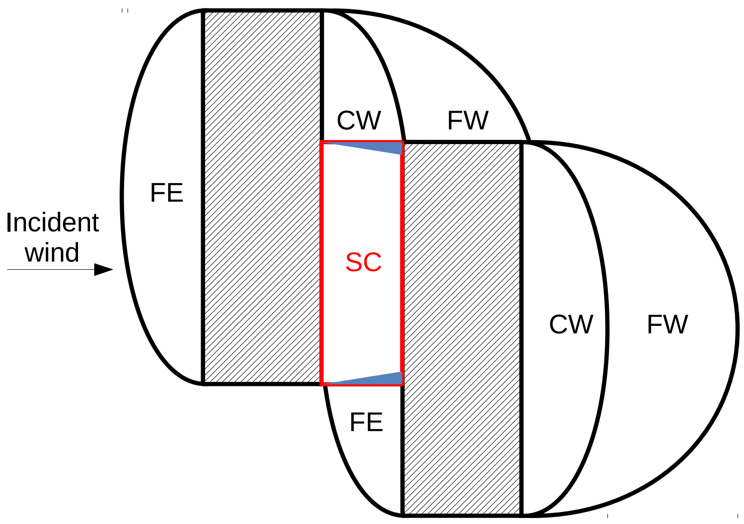


**fig. 6.6:** Outline indicating the position of a street canyon vortex (street canyon (SC), red) between two obstacles. The black rectangles filled with diagonal lines represent solid, rectangular obstacles. Incident wind is assumed perpendicular from the left. The blue triangles are indicating the position of the triangular transition zones at the ends of the street canyon vortex as proposed by Singh et al. (2008).

In the newly developed "StreetCanyonSingh" module, first, three more temporary speed grids are created to store the modifications by possible street canyon vortices. They are initialized on 9999 at all cells, to be able to distinguish between modified and original cells easily later.

In a next step, the direction of possible vortices between two obstacles is determined. To achieve this, the main flux grids _ugrid and _vgrid (at this stage only holding the undisturbed vertical profiles) are tested. If e.g. the most lowest cell in _ugrid holds a value unequal to zero m/s (indicating air movement in x-direction), there might be a street canyon vortex in y-direction. By checking if the value is positive or negative, also the direction of a street canyon vortex can be pre-determined.

The following will be done (depending on the incident wind direction) for a street canyon in x- and in y- direction. As the procedure is basically the same, only the calculations for the street canyons in y- direction are described here.

For any obstacle the module will test if there is a second obstacle present in a distance allowing for a street canyon vortex. To only consider relevant obstacles, they are sorted according to the air flow direction. Obstacles that are not solid or are smaller than one cell in height are skipped. Also obstacles not starting from the ground (e.g. the middle part of some bridge-like obstacle) are neglected. Finally, the maximum recirculation length $L_R$ is read from the obstacles properties (tab. 6.1). If $L_R$ is shorter than one cell, the obstacle is skipped for it can not possibly cause a vortex large enough to be considered by the model. If the obstacle is found to be relevant, the position of its rear wall in stream direction is determined.

From the rear wall of the first obstacle on, all obstacles in the lee are investigated to be in the right

orientation. All obstacles that are permeable to wind, that are not high enough or disconnected from the ground are skipped. For all the remaining obstacles, the front wall position and its distance to the rear wall of the first obstacle is determined. If that distance is larger than one cell, but smaller than the maximum recirculation length $L_R$ of the first obstacle a test is performed to see if the obstacles are overlapping in crosswind direction (compare to fig. 6.6).
The overlapping length is determined by comparing the obstacles corners. If this length is larger than one cell a street canyon vortex is assumed. According to Singh et al. (2008) this consists of a central vortex part and two triangular transition zones at the edges of the vortex (compare to fig. 6.6). Each of this vertical wedges is considered to extend over the whole distance between the obstacles in wind direction. In crosswind direction, they are assumed to cover 0.2 times the street canyons length. The vertical height is set by the lower one of the two obstacles in consideration.
For any cell within the street canyon between two obstacles a wind field modification needs to be calculated. Therefore the wedges and the main vortex area need to be distinguished. As the wedges are right-angled triangles, this can be done by basic trigonometry. For any point within the wedges, a streamwise modification is calculated based on eq. 4.60. The vertical speed is set to 0.0 m/s (Singh et al. 2008).
The wind speed modifications for the central part of the street canyon is calculated using eqs. (4.62) and (4.63) for the horizontal streamwise component as proposed by Singh et al. (2008). The vertical component is assessed according to the equation for the vertical stream modification in tab. 4.2 (Röckle 1990).

### 6.1.7 Assembling the initial wind field

The wind field modifications calculated by the modules "FrontEddyBagal", "CloseWake3", "FarWake2", and "StreetCanyonSingh" described in the sections 6.1.3 to 6.1.6 need to be merged with the local initial profiles generated by the "SetWD" module (section 6.1.2) to obtain the initial wind field (compare to section 4.3.4). The initial wind field is calculated by the module "AssembleInitialGrid". Besides the results provided by the modules stated above, the module requires the number of grids in the three directions, the obstacles table (tab. 6.1), and the three dimensional model domain grid holding the porosity $Por$ for each cell.
For writing all the modifications to the initial grid, the right order in the consideration of the modifications is very important. The lowest priority has to be copied first, the highest the last. The modification with the lower priority will, thus, be overwritten.
The modifications are written to the main initial grid in the following order:

- Initial local profile ("SetWD")
- Far wake zone ("FarWake2")

- Front eddy zone ("FrontEddyBagal")
- Lee-side recirculation ("CloseWake3")
- Street canyons ("StreetCanyonSingh")

Finally, all the cells that are filled by solid obstacles ($Por$ = 0), are overwritten by $0$ m/s to make sure there is no flow through solid matter.

### 6.1.8 Successive over-relaxation

The initial wind field calculated by the previous modules already is a much better assumption of the wind field under the given conditions than a simple profile (Röckle 1990). Still, it contains a lot of divergence. Assuming that the air inside the model domain is incompressible, the initial wind field has to be modified, until the "budget" of every cell is closest possible to zero. This means that the flux into a grid cell has to equal its outflow. To reduce the divergence, the fluxes into and out of a cell are modified iteratively, until the total remaining divergence reaches a number that is defined to be tolerated. The lower this number, the better the result, but the more iterations are required and, thus, the more computation time is needed.
The black and red successive over-relaxation method with Chebyshev acceleration after Press et al. (2007, p. 1062 ff) is used by this model according to section 4.3.4.2. This method is implemented in the "SOR" module to solve the Poisson equation for a central difference stencil (for the source code, refer to section 10.1.3). The module takes the grid spacing, the initial grids, the obstacles table, the model domain, the number of grids in the three directions, the stability parameters ($\alpha$, $\alpha_h$, $\alpha_v$), as well as the maximum number of tolerated iterations into account.
In the "SOR" module, a lot of parameters are set prior to the main iteration, as every operation becomes expensive that has to be performed over and over again.
The target divergence, the absolute divergence per cell to be accepted in the final wind field, is set to the number of $1 * 10^{-5}$ m/s inside the module. Also five new arrays are introduced. Three of them match the sizes of the three velocity grids and are used to store the result of the iterations. The other two are set to match the size of the model domain. They are used to store the Lagrangian multiplication factor $\lambda$ and the local divergence. All five arrays are initialized with zeros.
For all three directions a cell factor is calculated for the calculation of the local divergence. For the two horizontal directions, it is calculated by one divided by the area of the surface of the cell in this direction. Vertically, it is set to $\alpha$ divided by the vertical surface. As this factors have to be multiplied by two for most of the cells, as there are two permeable surfaces in the same direction (e.g. the left and the right side), a second version multiplied by two is saved for the three parameters. To save even more computation time, a cell factor for all the cells that are no border cells is saved as a constant.

Another variable that is initialized with zero, is the variable "delta" that is used to store the local divergence during the iteration. As they are used many times, three constants are created that represent the ranges of cells inside the model domain in x-, y-, and z-direction. They are used to control the iteration later.
The black and red SOR method requires two special x-ranges to address only the red, or only the black cells of the red-black chequerboard-like domain. Therefore two ranges of x-values, one starting with one, the other one starting with two, are created. Both ranges increase by two until they reach the total number of cells inside the model domain in x-direction. To check whether the first, or the second x-range has to be used, also a y-range, covering all values of y within one and the total number of y-grids, increasing by two, is saved. Finally, a variable is created that contains the current x-range. It is initialized using the x-range starting with one.
The initial local divergence is calculated using eq. 4.46, and set to zero for all cells, containing solid obstacles. The last parameters to be set prior to the main iteration are some variables to be used to calculate the over-relaxation variable $\omega$, that is initialized to be one, and the array containing information about the borders of each cell. This can be loaded directly from a temporary file, if a wind field was calculated for the same area before. If such a file is not present, the "makeborders"-method is called that is described in section 6.1.8.1.
The main iteration of the SOR routine first initializes, or resets a variable to store the sum of the local divergence. It is used to abort the iteration as soon as the remaining divergence becomes smaller than the acceptable divergence, stated on call of the module. A second iteration is run immediately, iterating over a variable "black" that may become FALSE and TRUE. It controls whether the black or the red cells are to be considered. Inside this iteration, two more iterations over the z- and y-range are run. For each value of z and y, a +1, and a -1 version is saved to a variable to save computation time. Inside the y-range iteration, the range of x-values is set according to "black" and the value of y. Finally, the inner iteration over the values of the x-range is started.
As for the values of z and y, there is a +1 and a -1 version of the current x-value saved to a variable. The current cell is now tested, to be located inside an obstacle. If this is the case, the cell will be skipped. If not, the array containing the information about the borders is queried with the current coordinates. If the cell is not located at any borders, the new local divergence for this cell is calculated by eq. 6.5, derived from eq. 4.45.

$$\frac{\lambda_{i-1jk}+\lambda_{i+1jk}}{(\triangle x)^2}+\frac{\lambda_{ij-1k}+\lambda_{ij+1k}}{(\triangle y)^2}+\frac{\lambda_{ijk-1}+\lambda_{ijk+1}}{\left(\frac{(\triangle z)^2}{\alpha}\right)}-\varepsilon^0_{ijk}-\lambda_{ijk}*\omega=\varepsilon^1_{ijk} \tag{6.5}$$

If the current cell is a border cell, the x-, y-, and z-direction have to be treated separately depending on the information given by the array containing the borders.
Applying the new local divergence $\varepsilon^1_{ijk}$, the local multiplication factor $\lambda_{ijk}$ is rewritten. The absolute value of $\varepsilon^1_{ijk}$ is added to the variable containing the sum of the divergence. The new value of lambda

is added to the variable for the sum of the multiplication factors. Now all iterations, except the main iteration are closed.
After having iterated over the whole grid for red and black, the abort criteria after Röckle (1990), the sum of the divergence divided by the sum of the multiplication factors, has become smaller than the rest divergence to be accepted. If this is not the case, a new over-relaxation factor is calculated applying the Chebyshev acceleration after Press et al. (2007, p. 1062 ff). The iteration continues. Else, the main iteration loop is terminated, and the new velocity grids are calculated using the array storing the local multiplication factors.

#### 6.1.8.1 Set cell borders

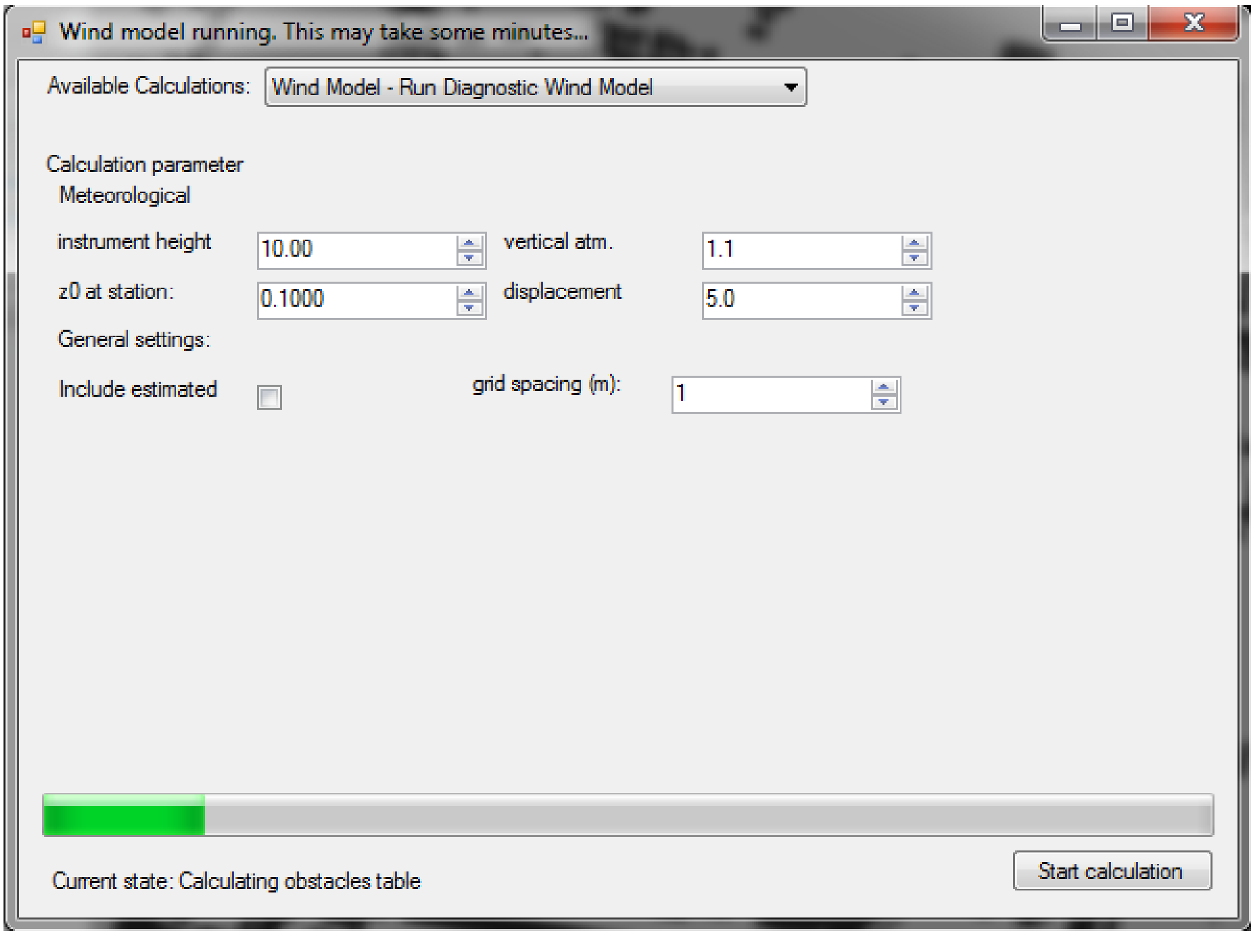
To determine the borders, all the single cells may be located at, the "makeborders"-function is called. It is based on the table containing the obstacles, and the number of grid cells in all the three directions.
The function first creates a new array matching the size of the model domain. The array is of type integer and initialized with zeros. This array is then filled with numbers indicating the borders in the vicinity of every cell. For not having to store tables of twelve boolean values for open and closed borders in the six directions for every cell, all the borders at one cell are represented by one integer number. This number is created using tab. 6.2. The numbers can easily be added, if a cell is located at several borders.

**tab. 6.2:** Table showing the values representing the different border types in the six directions.

| direction | closed border | open border |
|---|---|---|
| -z | 10 | 20 (never used) |
| +z | 1 | 2 |
| -y | 1000 | 2000 |
| +y | 100 | 200 |
| -x | 100000 | 200000 |
| +x | 10000 | 20000 |

According to tab. 6.2 the cells at the border of the model domain are set to be located at open borders in the particular direction. The cells at the -z end of the domain are set to be located at closed borders.
Now for all obstacles, the cells in their vicinity are set to be located at closed borders. The array containing the border information is then saved as a file to be already present if another wind field needs to be calculated for the same domain.

### 6.1.9 Integration in the SkyHelios model


Wind model running. This may take some minutes...

Available Calculations: Wind Model - Run Diagnostic Wind Model

Calculation parameter
Meteorological

instrument height: 10.00
vertical atm.: 1.1
z0 at station: 0.1000
displacement: 5.0

General settings:

Include estimated: ☐
grid spacing (m): 1

Current state: Calculating obstacles table

**fig. 6.7:** The graphical user interface to control the wind model and display the model status within the SkyHelios model.

To integrate the newly developed wind model into the SkyHelios model (see section 4.4.2), several steps need to be taken:

- The wind model needs to be compiled into a dynamically linked library (dll).
- The graphical user interface within SkyHelios needs to be enhanced to control the wind calculations and to display the wind model status while running.
- The SkyHelios backend needs to be extended to call the dll, supply the wind model with the necessary input data, and to receive the wind model output.

Combining two programs is, on the Microsoft Windows platform, most easily done using a dll as a common interface. Visual C# supports two kinds of dlls: Managed, and unmanaged ones. The first

can be included directly by the .NET framework. The latter ones are more universal and slightly faster. The wind model therefore is implemented using an unmanaged dll.
On the wind model side, a method "runWind" is implemented. It is callable through the dll interface and controls the wind model calculations. The method includes a callback function, that is able to send status information as set of one character string type text and a numeric value of type integer, the progress in percentages, back to the calling program.
Data is exchanged through the dll interface by passing pointers to reduce computation time and to save memory space. This especially is important for the huge, multi dimensional arrays, specifically the three dimensional array holding the model domain, but also for the two dimensional arrays containing the roughness length and displacement height as well as the array to return the results.

On the SkyHelios side, the implementation is more complex. The most visual part is the updated GUI. The "Computations" window now allows for the selection of "Wind Model - Run Diagnostic Wind Model" in the upper drop-down menu (fig. 6.7). If selected, the reference height (in m) for the incident wind speed and direction as well as the vertical atmospheric stability (factor from 0.5 to 1.5 for unstable to stable conditions) can be set there. Also the roughness length and the displacement height (both in m) for the reference wind speed and direction can be selected. A checkbox in the central left area allows for the application of the roughness length and displacement height calculated by SkyHelios for all points within the current area of interest. Finally, the desired horizontal grid spacing in meters for the wind data can be set. A button in the lower left corner of the window allows to immediately start the wind field calculations. During the calculations, the status bar and the label below are showing the current progress.

The major part of the modifications necessary for the integration of the wind model into SkyHelios is the work on the SkyHelios backend. First of all, SkyHelios needs to be able to provide (and first of all to generate) the necessary input.
A most crucial part of input data is a three dimensional array holding the cell densities (1 - $Por$), called "_build". To obtain this array, the number of spatial input files and the spatial input data type is checked. If there is only one shapefile holding buildings only, the three dimensional array "_build" can be generated by first applying the shape to raster method within SpatiaLite with the height field as the data attribute. This will create a raster file holding the building height for every cell within the area of interest (compare to fig. 6.8 left). Based on the raster file, the "_build" array can be created by filling the cells from the most lowest layer up to the layer agreeing to the height stated in the raster with 1.0 (this method can only consider buildings, that are solid). All cells above will stay 0.0 (and, thus, open, compare to fig. 6.8 right).
In spite the method described above is elegant and quite fast, it can only be applied if the only spatial input is a shapefile holding solid obstacles only. For any other case a more complex, but

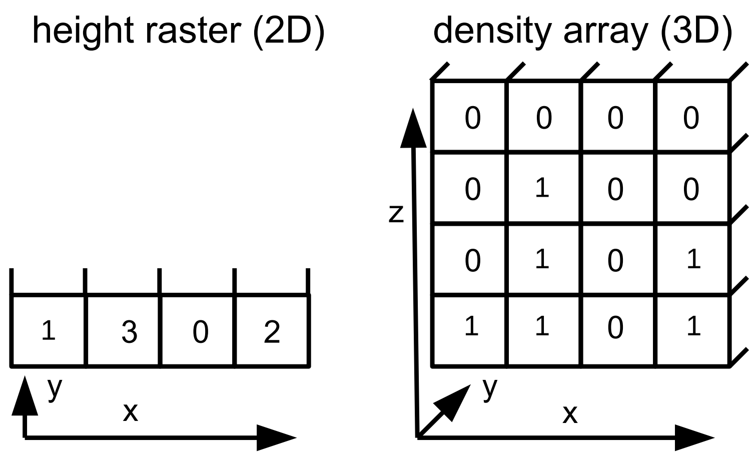


**fig. 6.8:** Relationship between the two dimensional height raster holding the metric heights of the area of interest (left) and the three dimensional density (1.0 - *Por*) array required for the wind model (right).

also more generally applicable method is required.

Starting from the idea, that MOGRE (the graphics engine) will always know the outline of the model

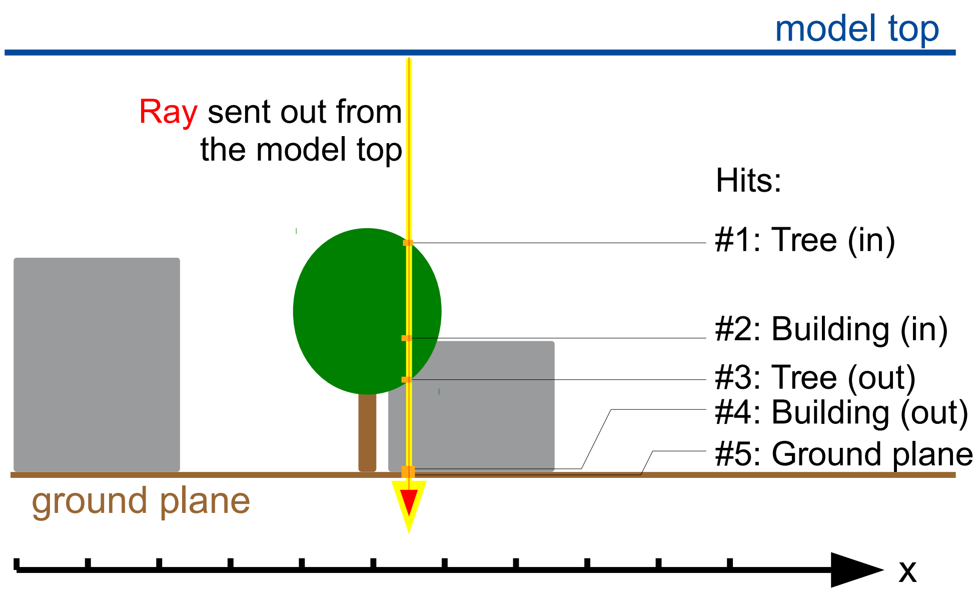


**fig. 6.9:** Sketch showing the principle of the method creating the three dimensional density (1.0 - *Por*) array required for the wind model by ray tracing: A ray is sent out from the model top vertically downwards. On its way down it will hit the border of all objects in its way (at least one, the ground plane) and remember their ID as well as the distance in meters from the model top to the hit. All obstacles are hit twice. First, when the ray is entering the obstacle and second, when it is leaving it.

area currently displayed, it will also know the highest point within this area (technically, the upper limit of the displayed scenes bounding box). Also, the coordinates of all discrete cells are well known. Finally, most graphics engines provide a built-in method called "ray tracing" (so does MOGRE). The method will sent a virtual ray from a certain position within the current scene in a given direction. The ray will now probe the segment from the position it was sent out to the point it hits the outer boundig box of the current scene. If it hits any object in between, it will remember two important things: the ID of the obstacle that was hit, and the distance from the point it was sent out to the location of the hit. Thereby, all three dimensional objects are hit twice. First, when the ray is entering the object and second, when it is leaving it.
Making use of this three things, the SkyHelios model is enhanced by a method using ray tracing to generate the three dimensional density (1.0 - $Por$) array required for the wind model. To do so, for any (horizontal) cell center, a ray is casted from the upper model border perpendicularly downwards (compare to fig. 6.9). For any object, that is hit by the ray, the distance to the ray's origin, as well as the objects ID is saved. The ray also counts the total number of hits. If the total number of hits is larger than one (the ground plane will always count one hit) the heights above ground of the ray entering and leaving the very same object is calculated. This heights are then mapped to cell heights and the appropriate cells of the "_build" array (that is initialized with 0.0 at all cells) are filled by the objects density.
This is not exactly the fastest method, but it is found to be the most appropriate for its compatibility with any kind of spatial input data or any combination of such. The computational time is considered acceptable as the "_build" array only needs to be created once for a model area.

The results calculated by the wind model are mostly used as initial data for other model parts within the SkyHelios model. They therefore are assigned to the results manager (see section 6.2) as soon as they are becoming available.

## 6.2 Results manager

The initial SkyHelios modes as described in section 4.4.2 is performing most calculations instantly displaying the results in the GUI (e.g. a Fish-eye image (e.g. fig. 4.11a), or the local SVF). This is possible for simple calculations that are not depending the results of each other. With increasing complexity, the SkyHelios model needs a more organized way to deal with the computed results.
To address this shortcoming, a results manager is implemented. Prior to any calculation, the results manager checks if it is already holding the result or if there is a temporary file available doing so. If not, the calculation is performed and the result is stored in the results manager.
If a calculation is desired, that requires the result of another, e.g. if a thermal index (see section 4.3.6)

is to be computed requiring the local wind velocity, SkyHelios automatically detects if wind data is already available by the results manager or through a temporary file. If yes, that result will be used to calculate the thermal index immediately. If not, the wind model is called automatically and the calculation of the thermal index is interrupted until the local wind speed is available. Both of the new results, the wind speed as well as the thermal index, will be stored in a temporary file to be available if they are needed later.

## 6.3 Multithreading

The initial SkyHelios modes as described in section 4.4.2 is (from its backend) a single thread application. This means, that there is only one thread doing all the work one after the other. However, there is quite some work, that could (assuming running on a modern multi-core CPU) be done in parallel, e.g. thermal indices could be calculated for several cells at the same time during spatial calculations. Ideally, this could speed up calculations by almost the number of CPU cores.
Of course not everything can be done in parallel. This sometimes is due to technical reasons and limitations, but sometimes also for it is logically impossible. The first is mostly caused by Visual C# being unable to call a dll extension twice at the same time. This makes it impossible to call the wind model several times in parallel. Also MOGRE does not fully support multithreading. Examples for things, that must not be done at the same time are processes modifying general properties (e.g. the current scene) or calculations depending on each others results. Everything, that is intended to run in parallel or is affected by anything that is possibly running in parallel therefore needs to be redesigned in a thread-safe way. This is done by providing the different threads with own deep-copied objects and methods where ever possible and, if not possible protect sensitive objects by using semaphores.
In SkyHelios, multithreading can be applied most efficient during area output calculations. While the wind model can be called only once and needs to complete calculations before other calculations (e.g. for thermal indices) can be performed, most other calculations can be done in parallel. Therefore as many threads are created, as CPU cores are detected (e.g. four threads on a CPU with four cores). Each thread will receive a deep-copy of the scene model and most input data. The area of interest is then split into segments along the y axis (compare to fig. 6.10). The number of segments thereby is determined by the number of threads. Each tread is now able to independently calculate results for its own segment, that are stored in the (common) results manager (see section 6.2).

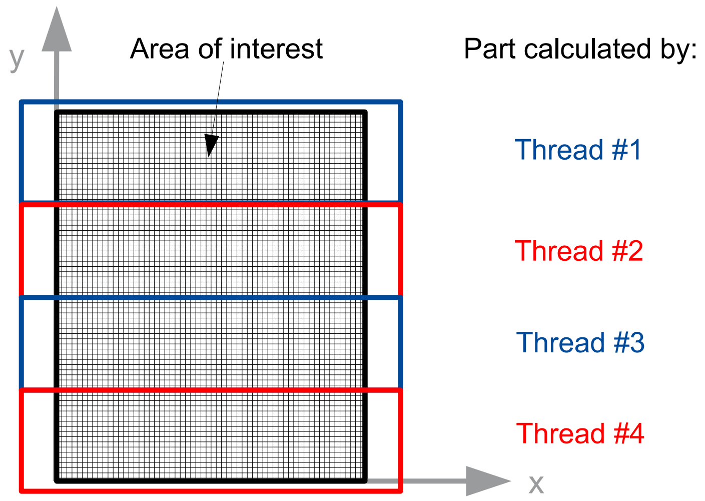


**fig. 6.10:** Area of interest (black grid) split up into four parts to be processed by four threads in parallel (assuming running on a CPU with 4 cores).

## 6.4 Roughness length and displacement height

During the work on this dissertation project a new model part allowing for the calculation of roughness length $z_0$ and displacement height $z_d$ was implemented into the SkyHelios model by Marcel Gangwisch. It allows for the estimation of the roughness length following the three different approaches introduced in section 4.3.2 (Ketterer et al. 2016). Technically, the roughness calculations are applying the same "computations" environment as the wind model.

For any roughness calculations, first of all, reference areas are required. As rather arbitrary reference areas like lot areas or regular grids are found inappropriate, the reference areas are determined based on a voronoi diagram in SkyHelios (refer to section 4.3.2). The voronoi diagram is computed using a sweepline algorithm after Fortune (1987). The method is found to be quite fast. However, it only supports point data. All obstacles therefore are split up into sample points consisting of the polygon vertices and additional sample points on the outline of the polygons.

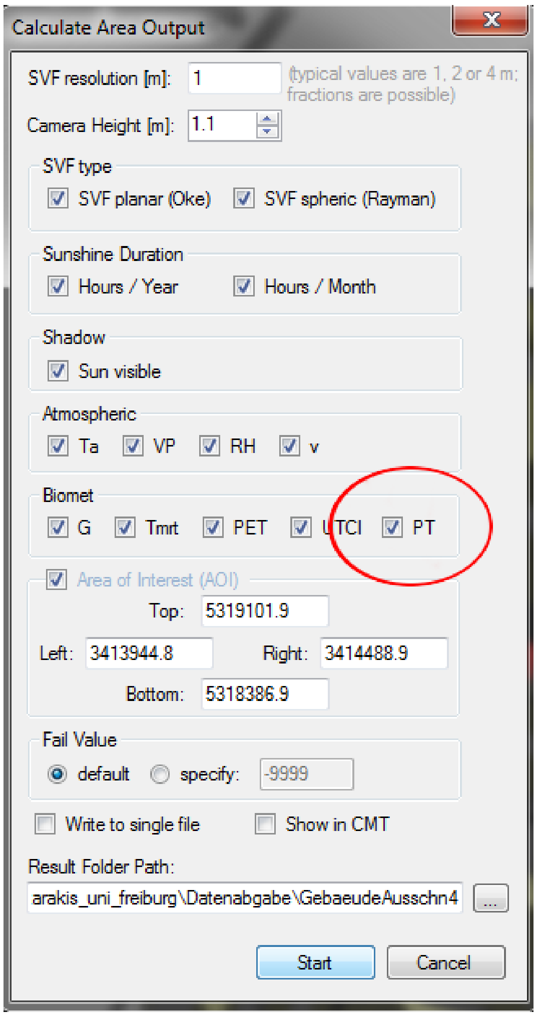
Calculating a voronoi diagram for the sample points results in a huge number of voronoi cells. For the voronoi calculations, the cells are merged depending on the obstacle they belong to. Afterwards, they are clipped not to overlap with either an obstacle or with themselves. This results in a subdivision of the entire area of interest based on the obstacles, that can be used as reference areas.

The roughness parameters $z_0$ and $z_d$ can be estimated based on the obstacles geometric properties and the reference areas applying the methodology after Lettau (1969), Matzarakis and Mayer (1992), Bottema (1997), Bottema and Mestayer (1998, compare to section 4.3.2) depending on user selection.

Spatial roughness calculations are found to provide valuable results for analysis (Ketterer et al. 2016). However, they can also be applied as additional input data for the wind model. If the "include estimated roughness" checkbox in the central left part of the wind model's GUI is checked (compare to fig. 6.7), the wind model will ask the SpatiaLite database for spatial roughness length and displacement height after Bottema (1997), Bottema and Mestayer (1998). If they are not present, roughness calculations according to section 4.3.2.2 are performed.
As soon as results for $z_0$ and $z_d$ are available in the SpatiaLite database, the "rasterize" method is used to obtain a grid in the appropriate resolution to serve as input for the wind model.

## 6.5 Thermal indices

A module calculating the thermal indices Perceived Temperature (Staiger et al. 2012, see section 4.3.6.1), Universal Thermal Climate Index (Jendritzky et al. 2012, see section 4.3.6.2) and Physiologically Equivalent Temperature (Höppe 1999, compare to section 4.3.6.3) is developed and included into the SkyHelios model (see section 4.4.2). This is achieved by little work on the graphical user interface (GUI), where a checkbox for PT, UTCI and PET calculations is added (see e.g. for PT fig. 6.11) and by adding all the necessary methods.
As the modules for the calculation of PT, UTCI and PET are basically working the same way, this is only described once using the example of the perceived temperature.
The perceived temperature methods are implemented within a custom class "PerceivedTemperature". The class is inheriting from the custom interface "IThermalIndex", that is also used in order to calculate UTCI (section 4.3.6.2) and PET (section 4.3.6.3).
"IThermalIndex" provides the new class with two custom constructs. One of them is "RayManModel". the other one is called "PersonData". While "RayManModel" provides the class "PerceivedTemperature" with the necessary meteorological information ($T_a$, $VP$, $v$, and $T_{mrt}$), as well as the global fail value. "PersonData" holds a list of physiological parameters (compare to tab. 6.3). Not all properties of tab. 6.3 can be used during the perceived temperature calculations. The only index to support all of the properties within "PersonData" currently is PET. PT only takes the physiological parameters age, weight and height. They are replacing hard-coded values in the original PT code. However, only the persons weight seems to be considered during the calculations. The activity level is considered static at 134.69 W.
The calculation of the perceived temperature itself is based on a slightly modified port of a Fortran version provided by the German Meteorological Service (DWD). The modifications are mainly for technical reasons, as well as to comply to the requirements of the Interface "IThermalIndex". One such requirement is a "Compute" method, that controls the calculation. It first checks if one of the meteorological input variables is equal to the fail value. In this case, the fail value is returned (as


Calculate Area Output

SVF resolution [m]: 1 (typical values are 1, 2 or 4 m; fractions are possible)

Camera Height [m]: 1.1

SVF type
[✓ SVF planar (Oke)] [✓ SVF spheric (Rayman)]

Sunshine Duration
[✓ Hours / Year] [✓ Hours / Month]

Shadow
[✓ Sun visible]

Atmospheric
[✓ Ta] [✓ VP] [✓ RH] [✓ v]

Biomet
[✓ G] [✓ Tmrt] [✓ PET] [✓ UTCI] [✓ PT]

[✓ Area of Interest (AOI)]
Top: 5319101.9
Left: 3413944.8 Right: 3414488.9
Bottom: 5318386.9

Fail Value
[✓ default] specify: -9999

Write to single file Show in CMT

Result Folder Path:
arakis_uni_freiburg\Datenabgabe\GebaeudeAusschn4

**fig. 6.11:** The modified graphical user interface controlling the spatial calculations. The red circle marks the new checkbox for the PT calculations.

all of them are required and nothing useful can be calculated if one of them is missing) and the calculation is interrupted. If all values are valid, some local variables are set and the calculation according to section 4.3.6.1 is started by calling the "pt_basic" method.

## 6.6 Spatial input and display

The three dimensional part of the SkyHelios model is the model part dealing with the spatial input and displaying it using the MOGRE graphics engine (Matzarakis and Matuschek 2011). It therefore creates a three dimensional scene out of the spatial input files. The scene is then passed to the MOGRE engine for display.

**tab. 6.3:** Outline of the properties of the construct "PersonData" together with the corresponding units and data types.

| name | unit (if any) | type |
|---|---|---|
| Age | y | integer (never used) |
| Weight | kg | double |
| Height | m | double |
| Activity | W | double |
| Clothing | | double |
| Sex | | custom type (male or female) |
| PersonPosition | | custom type (standing or sitting) |
| Workload | W | double |

This can be done quite easily as long as there is only one spatial input file containing a fairly small model area. As soon as the area becomes larger, or several spatial input files are to be considered at once, there are some important things to consider. First of all, their relative orientation and size need to be determined and considered. Second, they need to be transferred into a three dimensional scene optimized for the graphics card to be displayed with minimum inaccuracy.
The relative location, orientation and size of two spatial input files can be determined if their geographic projection is known. The first files projection is thereby used as the "project coordinate system". All files loaded afterwards are automatically reprojected to the project coordinate system, what makes them match perfectly. For spatial input files that can't have a geographic coordinate system (e.g. RayMan obstacle files), an anchor point can be set. It is a point with coordinates in a well known projection. As all supported file types of this kind are holding metric vectors, this is sufficient to add them to the scene correctly.

Spatial input files can have, depending on their geographic projection, very large coordinates. E.g. the coordinates of Freiburg im Breisgau (round about 47° 59' 56.428" N, 7° 50' 31.575" E) in the popular reference system Universal Transverse Mercator (UTM, Zone 32T with base World Geodaetic System 1984 (WGS 84)) are Easting: 413625.14 and Northing: 5316838.57. Graphic cards, however, can only consider coordinates in float point precision leading to rounding errors of several meters in the positions. This can be fixed by normalizing the scene passed to the graphics card by lowest x- and y-coordinate (compare to fig. 6.12) and calculating in double precision where geographic coordinates are required. This leads to three different coordinate systems used in SkyHelios: The original geographic coordinate system, the normalized "internal" one and one coordinate system required for the MOGRE engine using a flipped z-axis instead of the y-axis (compare to fig. 6.12).
The maximum raster size in the original SkyHelios model is very limited. This is due to the MOGRE

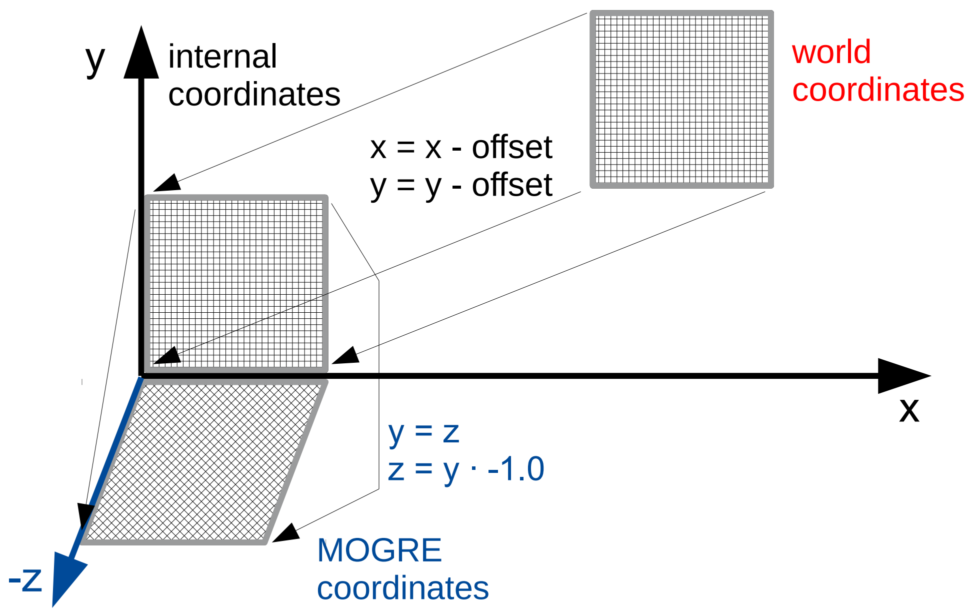


**fig. 6.12:** The three types of coordinates used within SkyHelios: World coordinates (right, red) are the original coordinates of the spatial input file. They're moved (close) to the origin by subtraction by an offset becoming internal coordinates (upper left, black). To be displayed, they need to be moved again, to match the coordinate system used by the MOGRE engine. While the x coordinate stays the same ($x_{MOGRE} = x_{internal}$), the y coordinate is the vertical one now ($y_{MOGRE} = z_{internal}$) and the z coordinate is the negative y one ($z_{MOGRE} = -1.0 \cdot y_{internal}$)

engine only allowing for a certain number of vertices within one mesh object, specifically 65536 vertices, while four vertices are required to represent one grid cell. To allow for raster files exceeding 16384 grid cells (more than e.g. 128 · 128 cells), the spatial input raster module needs to be restructured and partly redesigned. A most promising solution for the limitation was found in splitting the raster into four subrasters (tiles). Each tile is tested to hold less than 16384 grid cells. Larger tiles are split into four again until they are small enough. Now, each tile is passed to the MOGRE engine forming an own sub-mesh. This method is expected to not only allow for larger raster input files, but also to speed up display, as only the raster tiles that are currently visible in the output need to be considered by the GPU.

## 6.7 Meteorological input data file

In SkyHelios calculations can be performed based on meteorological data specified in the graphical user interface (GUI). However, SkyHelios is intended to be applicable not only for case studies with individual sets of data, but for long data series. As it is very inconvenient to enter lots of datasets manually through the GUI, the SkyHelios model was extended by a module to consider meteorological input data files (containing meteorological readings as shown by tab. 6.4).

**tab. 6.4:** Example for data stored in a meteorological input file. The example contains four rows of data with the parameters $T_a$ (°C), $VP$ (hPa), $v$ (m/s), $WD$ in (°) and $G$ in $\frac{W}{m^2}$.

| date | time | Ta | VP | v | WD | G |
|---|---|---|---|---|---|---|
| 07.08.2015 | 08:00 | 25.4 | 17.0 | 1.2 | 248 | 350 |
| 07.08.2015 | 12:00 | 33.5 | 14.2 | 4.5 | 253 | 841 |
| 07.08.2015 | 14:00 | 35.5 | 13.7 | 3.0 | 221 | 817 |
| 07.08.2015 | 16:00 | 36.9 | 11.6 | 2.0 | 269 | 590 |

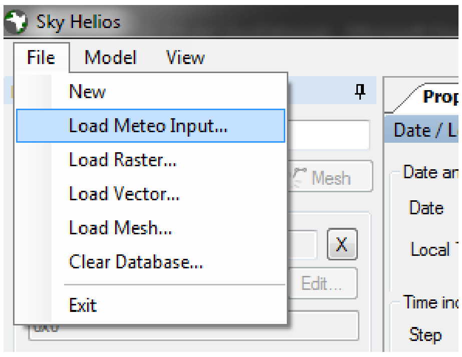
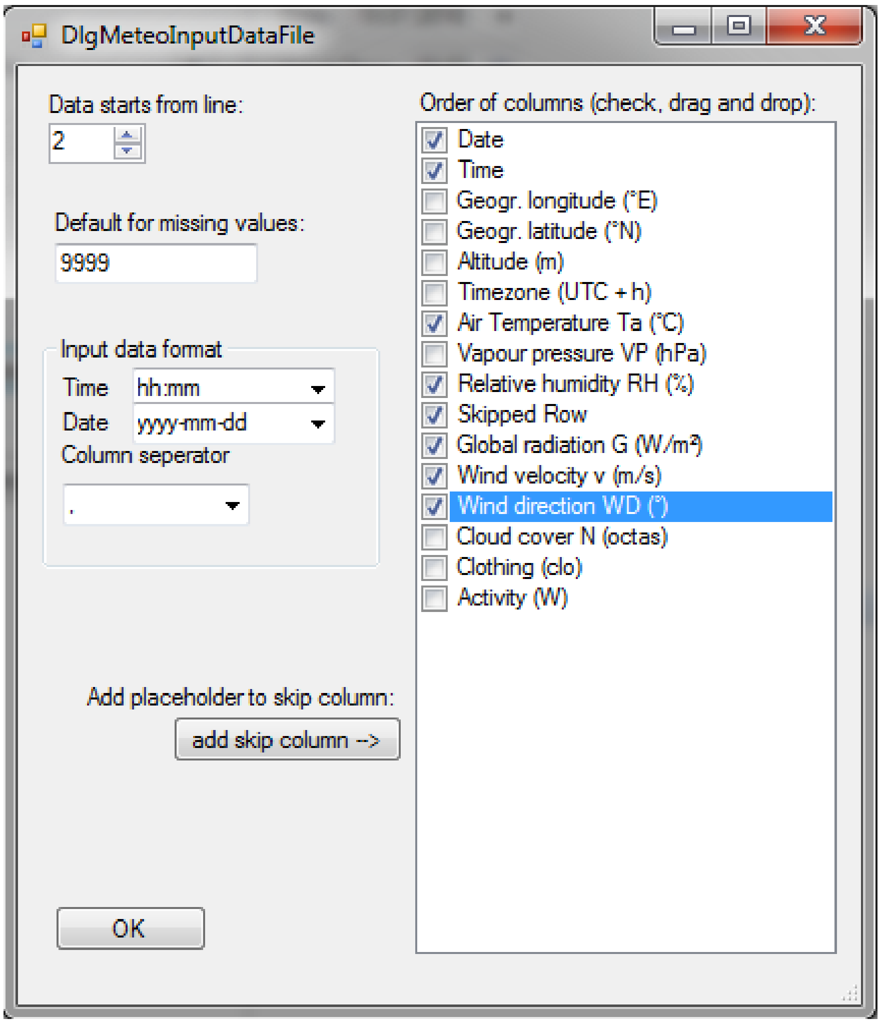


**fig. 6.13:** Extenden SkyHelios main menu allowing for loading a meteorological input data file.

SkyHelios supports any delimited text file as meteorological input data. The order of columns, header rows, the column delimiter and even blanc columns (compare to fig. 6.14), that are to ignore can be configured using the new "Meteo Input File" window accessible through the main menu (refer to fig. 6.13).
Once a meteo input data file is loaded any spatial calculations will be performed for any row of data within the file (four times in the example of tab. 6.4). All calculations are performed targeting maximum efficiency and performance. Results from the first row of data (or even from before) will be used in the further calculations wherever possible to avoid redundant calculations. E.g. the SVF will not change for any meteorological data and is, thus, only calculated once. All results will be stored as temporary grids and will stay in the temp folder to be available if they can be used later on (compare to section 6.2).


DlgMeteoInputDataFile

Data starts from line: 2

Default for missing values: 9999

Input data format
Time: hh:mm
Date: yyyy-mm-dd
Column seperator: .

Add placeholder to skip column:
add skip column -->

OK

Order of columns (check, drag and drop):
- [✓] Date
- [✓] Time
- [ ] Geogr. longitude (°E)
- [ ] Geogr. latitude (°N)
- [ ] Altitude (m)
- [ ] Timezone (UTC + h)
- [✓] Air Temperature Ta (°C)
- [ ] Vapour pressure VP (hPa)
- [✓] Relative humidity RH (%)
- [✓] Skipped Row
- [✓] Global radiation G (W/m²)
- [✓] Wind velocity v (m/s)
- [✓] Wind direction WD (°)
- [ ] Cloud cover N (octas)
- [ ] Clothing (clo)
- [ ] Activity (W)

**fig. 6.14:** New form of the SkyHelios graphical user interface allowing for the configuration of a meteorological input data file.

## 6.8 Test cases

The new functionality is tested for two study areas in Freiburg, south-west Germany. Freiburg is located within the sharp transition of the Upper Rhine Plain and the Southern Black Forest mountain range at the entry of the Dreisam valley (approximately 47° 59' N, 7° 51' E). The city of Freiburg is selected for the test cases for several reasons. First, Freiburg is considered the warmest city in Germany (Nübler 1979, Rudloff 1993). This causes a strong demand for tools assessing thermal comfort and thermal stress in urban planning, as thermal stress is expected to increase due to global climate change (e.g. Matzarakis and Endler 2009; 2010). Second, the municipality claims Freiburg to be a "Green City" and is supporting environmental studies by providing spatial data

(e.g. a vector layer containing buildings, areal images and the tree register). Finally, the Chair of Environmental Meteorology is operating a meteorological background station (Matzarakis et al. 2000) at the top of a highrise building within the "Institutes Quarter" (see section 6.8.2) providing data that can be used as a roof-top reference. The stations records are covering a 13-year period from Sept. $1^{st}$, 1999 to April $30^{th}$, 2013 in hourly resolution. The parameters used in the test cases are air temperature ($T_a$) in °C, vapour pressure ($VP$) in hPa, global radiation ($G$) in $\frac{W}{m^2}$, wind speed ($v$) in m/s and wind direction ($WD$) in °.
The main aim of the test cases is to test the models applicability. This does require the application of a meteorological data input file containing several rows of data. However, as results are spatial, presentation requires huge maps. Presenting a huge amount of results is therefore neglected, as it is considered not necessary for this purpose. To keep the results section as short as possible, the model was tested using the meteorological data input file shown by tab. 6.4. The file is extracted from the urban climate stations readings, providing values for a hot and dry summer day.

### 6.8.1 Place of the old synagogue

The place of the old synagogue is located westwards from the inner city of Freiburg between the main university buildings I and II (KG I and KG II), the university library and the theatre (compare to fig. 6.15). It is a popular, large open space of approximately 100 m on 150 m. The municipality currently intends to redesign it, causing controversial discussions about the new design. The high level of public attention, as well as previous studies (Fröhlich and Matzarakis 2011; 2013) make the place of the old synagogue a very interesting area of interest for this test case. The original place of the old synagogue is divided into four sections by two roads crossing it. The larger one, the "Rotteckring" crossing from north to south, is a major, asphalt covered road claiming more than a third of the place's area. The smaller one is the "Bertholdstraße" crossing from north-west to the east. It is part of the pedestrians-only area and mostly covered by a reddish pavement. The two remaining areas are a smaller one in front of the theatre and a larger one in front of KG II. The smaller one is half covered by lawn and half by dark pavement. The larger one is unsealed and mostly covered by lawn. All in all, there are 22 trees of different species and size scattered all over the place. As they all are deciduous trees, they provide shade in summer and maintain access to direct solar radiation in winter. This leads to a significant impact on local thermal bioclimate (Fröhlich and Matzarakis 2011; 2013).
After the redesign, according to current planning, the place will be sealed completely by large light-coloured stone plates in the central area, framed by small-pebble pavement. Both of the streets, the "Roteckring" as well as the "Bertholdstraße" will become part of the pedestrian precinct and will be integrated in one single place area. Only a small section in front of the theatre will be

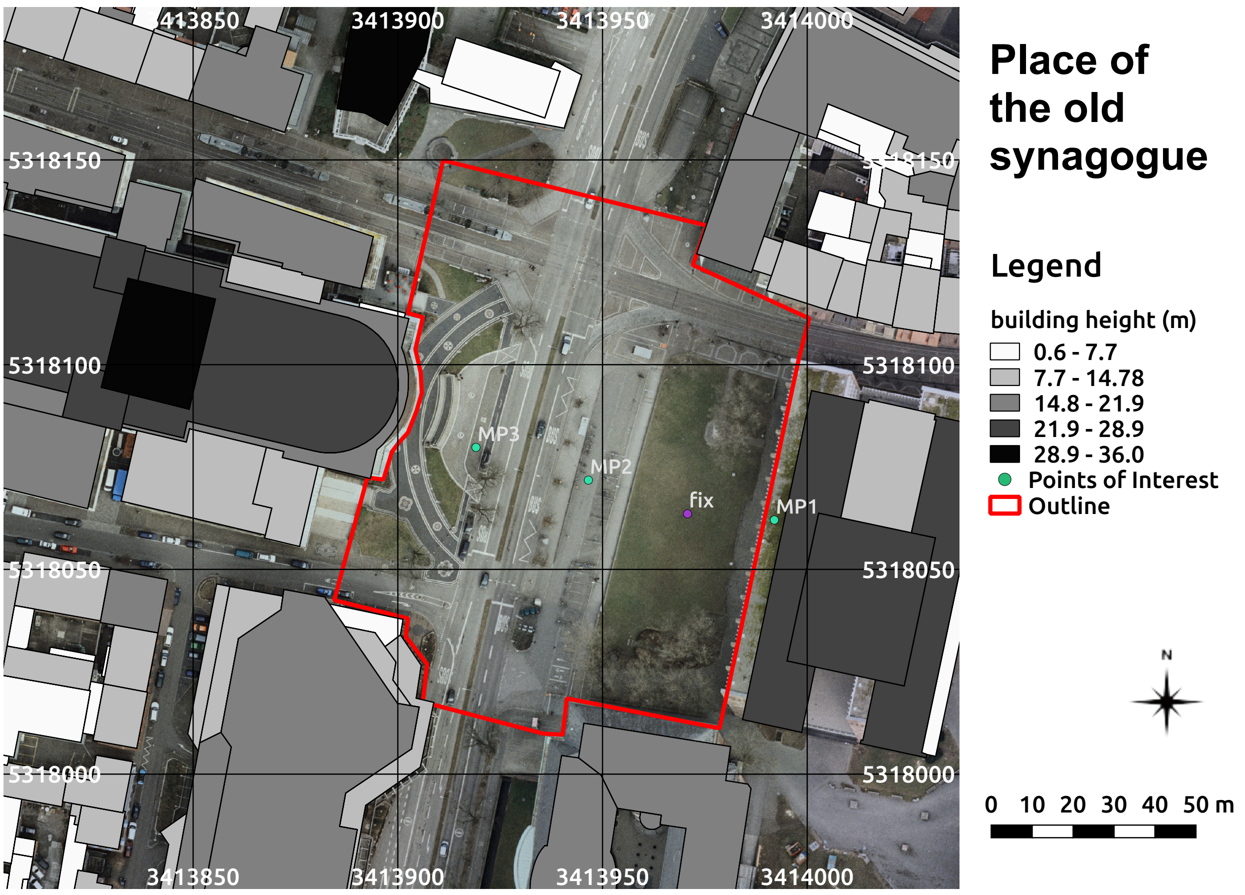


**fig. 6.15:** Overview over the place of the old synagogue in Freiburg, south-west Germany. Building heights above ground (m) and aerial imagery provided by the municipality of Freiburg. The outline of the place of the old synagogue is depicted in red. The green and violet points are showing the location of points for the comparison of measured and modelled results (see section 6.8.3). The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

separated optically by covering it with darker pavement. In front of the theatre as well as in the south-east of the place, shallow water basins are planned. Many of the trees will be removed. Some small crowned deciduous trees will be added in the south and east of the place keeping a large central area exposed to direct solar radiation (Fröhlich and Matzarakis 2011; 2013).
The intention running this test case is to check the applicability of the SkyHelios model for an area of interest formed by several spatial input files. The area of interest is formed by two spatial input files of different types: a (polygon) Shapefile holding the buildings and a RayMan obstacle file containing the trees.

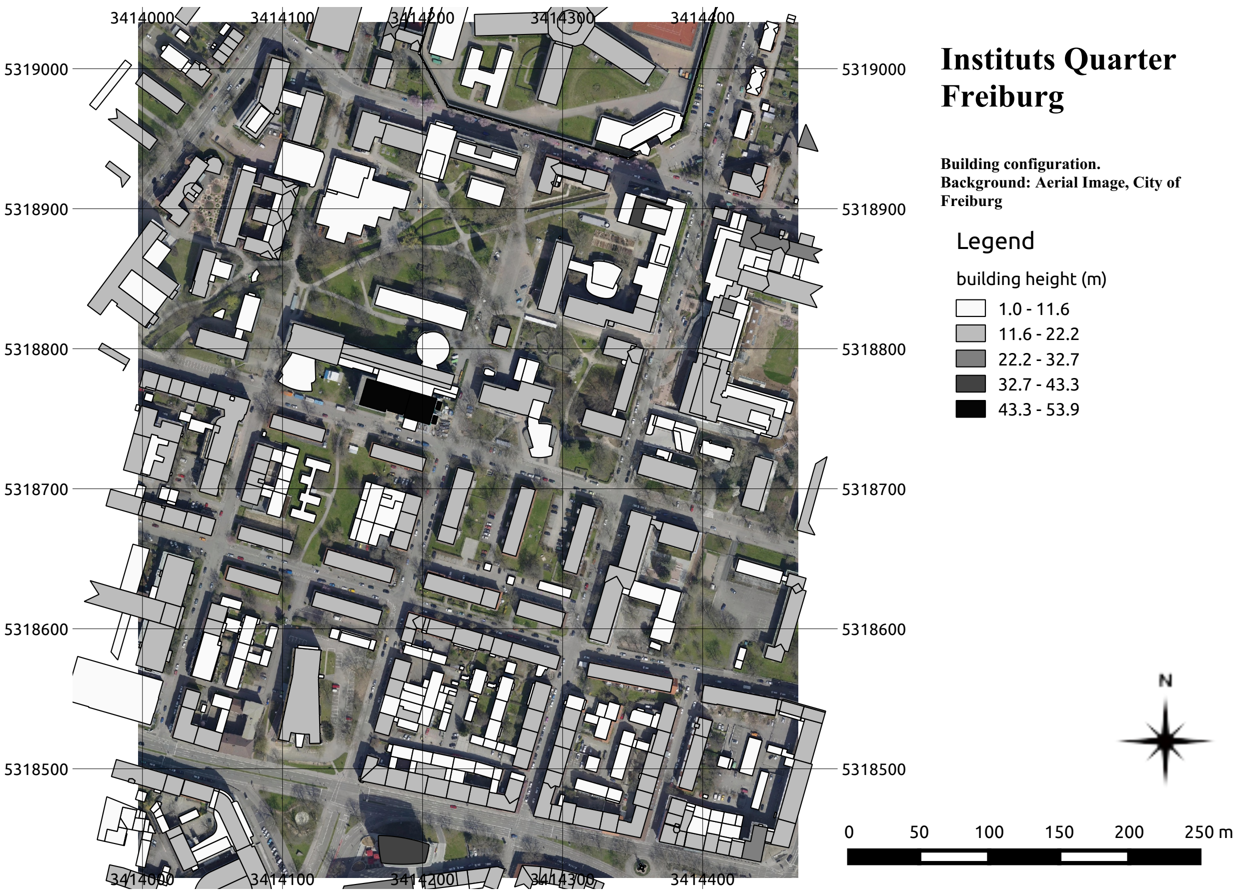


**fig. 6.16:** Overview over the Institutes Quarter in Freiburg, south-west Germany. Building heights above ground (m) and aerial imagery provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

## 6.8.2 Institutes Quarter

The Institutes Quarter in Freiburg, south-west Germany, is a city quarter north of the city center mainly consisting of institute buildings of the Albert-Ludwigs-University Freiburg. It coveres an area of approximately 700 m on 500 m starting from the "Tennenbacher Straße" in the north extending down to the "Friedrichring", a major street in the south. In the East-West direction the Institutes Quarter is located in between "Merianstraße" and "Soutierstraße" in the east, and the "Schnewlinstraße" in the west.
The area was selected to serve as an area of interest for this test case for its size and the perfect availability of data. A spatial vector dataset containing the buildings was provided by the municipality of Freiburg, along with aerial imagery (compare to fig. 6.16). Meteorological background data was available through the Chair of Environmental Meteorology operating the urban climate station

(Matzarakis et al. 2000, Matzarakis and Mayer 2008) on top of the chemistry faculty's high-rise building in the center of the Institutes Quarter (dark grey building in the center of fig. 6.16).
The intention behind this test case is to proof the stability of the SkyHelios model running for a huge model area.

### 6.8.3 Comparison to measurements

Results for the current place of the old synagogue (refer to section 6.8.1) on the $1^{st}$ of July 2008 calculated by the advanced SkyHelios model are compared to measured data from a one day measurement campaign on the place to get an insight on the models accuracy. The data was provided by a local human-biometeorlogical station (for a picture of the station refer to fig. 4.12b) and a mobile station recording at three measurement points ("MP1" to "MP3", see fig. 6.15).
The local climate station, placed in the central part of the place (violet point labelled "fix" in fig. 6.15) provides the parameters $T_a$, $v$, $WD$, $G$, and (abbreviated based on the six-direction method according to section 4.3.5.4) $T_{mrt}$ for the time in between 06:30 and 21:03 in 1 minute resolution. The station details are published by Mayer et al. (2008).
The mobile measurements are available in hourly resolution. They are gathered using a hand cart based system suitable to access the measurement points within the pedestrian only area (green points labelled "MP1" to "MP3" in fig. 6.15). The details of the mobile station can be found in Mayer et al. (2008).
The SkyHelios model is performing the calculations based on the meteorological data provided by the urban climate station Freiburg (Matzarakis et al. 2000) for the same points in time the mobile data is available.
The meteorological parameters calculated by the model are compared to the ones measured on site for the corresponding locations and the times. $T_{mrt}$, as well as the thermal indices PT and PET for the stations on site are calculated by the RayMan model (see section 4.4.1) and compared to the corresponding ones determined by the advanced SkyHelios model.

# 7 Results

The model is applied for four test situations at two different test domains, the place of the old synagogue in the current, as well as in the redesigned form, and the Institutes Quarter in Freiburg (refer to section 6.8). The test situations each consist of four sets of meteorological input conditions represented by the four lines of data specified by tab. 6.4.
The current place of the old synagogue is additionally used for a comparison between modelled values and measurements, that took place on the 1$^{st}$ of July 2008.

## 7.1 Place of the old synagogue

The four test situations are applied for the current (see section 7.1.1), as well as for the redesigned place of the old synagogue (compare to section 7.1.2). Results are compared to show the modifications due to the redesign.

### 7.1.1 Old place of the old synagogue

For the current place of the old synagogue the spatial indices PT, UTCI and PET (see section 4.3.6) are determined by the SkyHelios model. Results are presented along with some of the pre-results they are dependent on for the current place, to show the dependencies and allow for some insight on the interdependencies.

#### 7.1.1.1 Sky View Factor

The spatial distribution of the spheric SVF (see section 4.3.5.1) at the current place of the old synagogue is shown by fig. 7.1. Bright colors are indicating small SVF, while violet colors show areas with only little horizon limitation. The calculations are based on the buildings (grey polygons in fig. 7.1), but also on trees. The trees are not visible in fig. 7.1, but their position is clear to see by their impact on SVF in terms of almost white areas in the center of the place.
Like most of the other calculations, SVF is calculated in parallel using as many threads as CPU

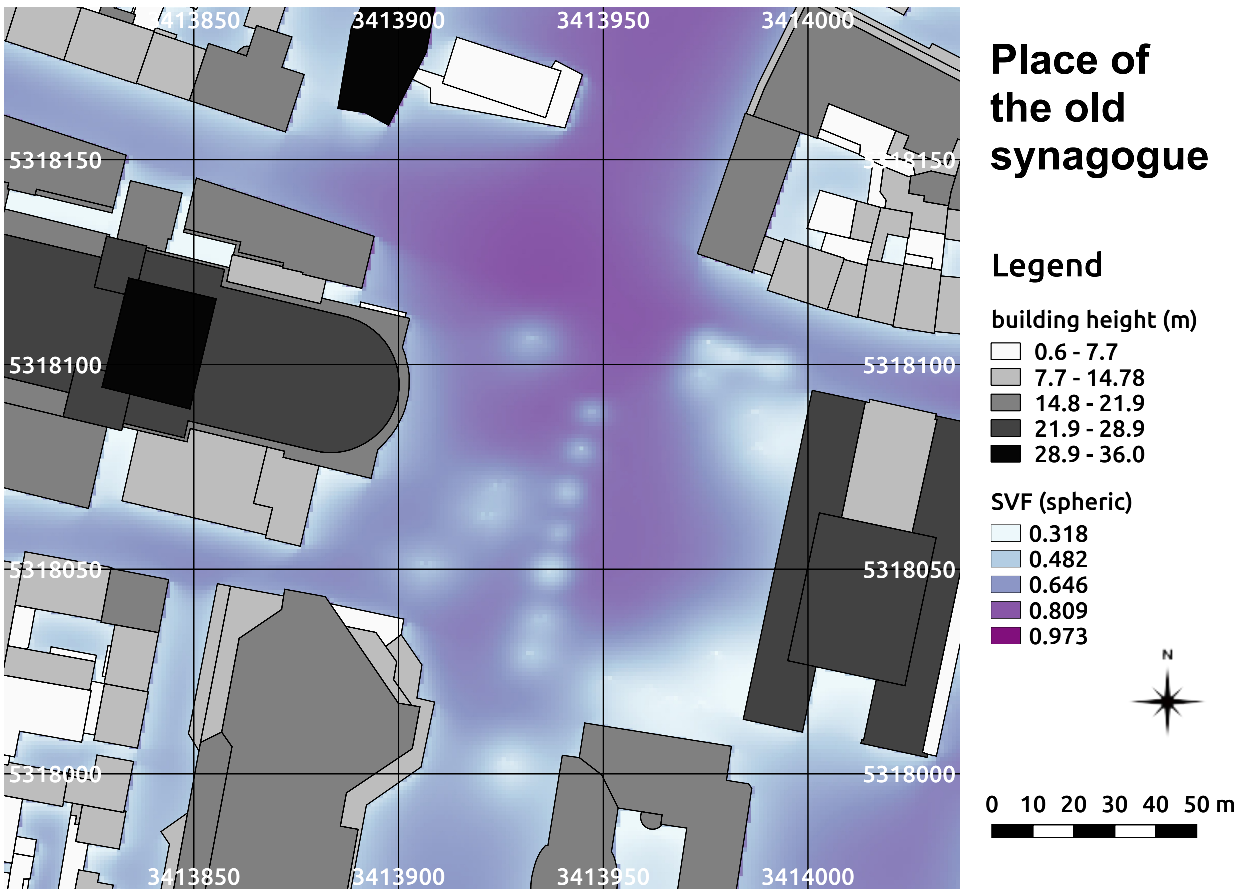


**fig. 7.1:** Spatial distribution of SVF at the place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree properties and positions (compare to fig. 6.15) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

cores are available. The SVF shown by fig. 7.1 therefore is calculated by two independent threads splitting the area of interest into a northern and a southern part. Both parts are merged afterwards. Fig. 7.1 shows, that this does not cause any error or imprecision, that would cause a horizontal line of mismatching values.

SVF results show relatively small horizon limitation in the central part of the place, as well as at the crossroads in the north of the place. SVF is decreased severely below the trees, as well as close to the buildings. In the central area of the place, SVF reaches values of up to 0.97, while it is decreased to approximately 0.50 in the narrow streets west of the university library and further down to values of around 0.40 under the trees.

#### 7.1.1.2 Wind speed

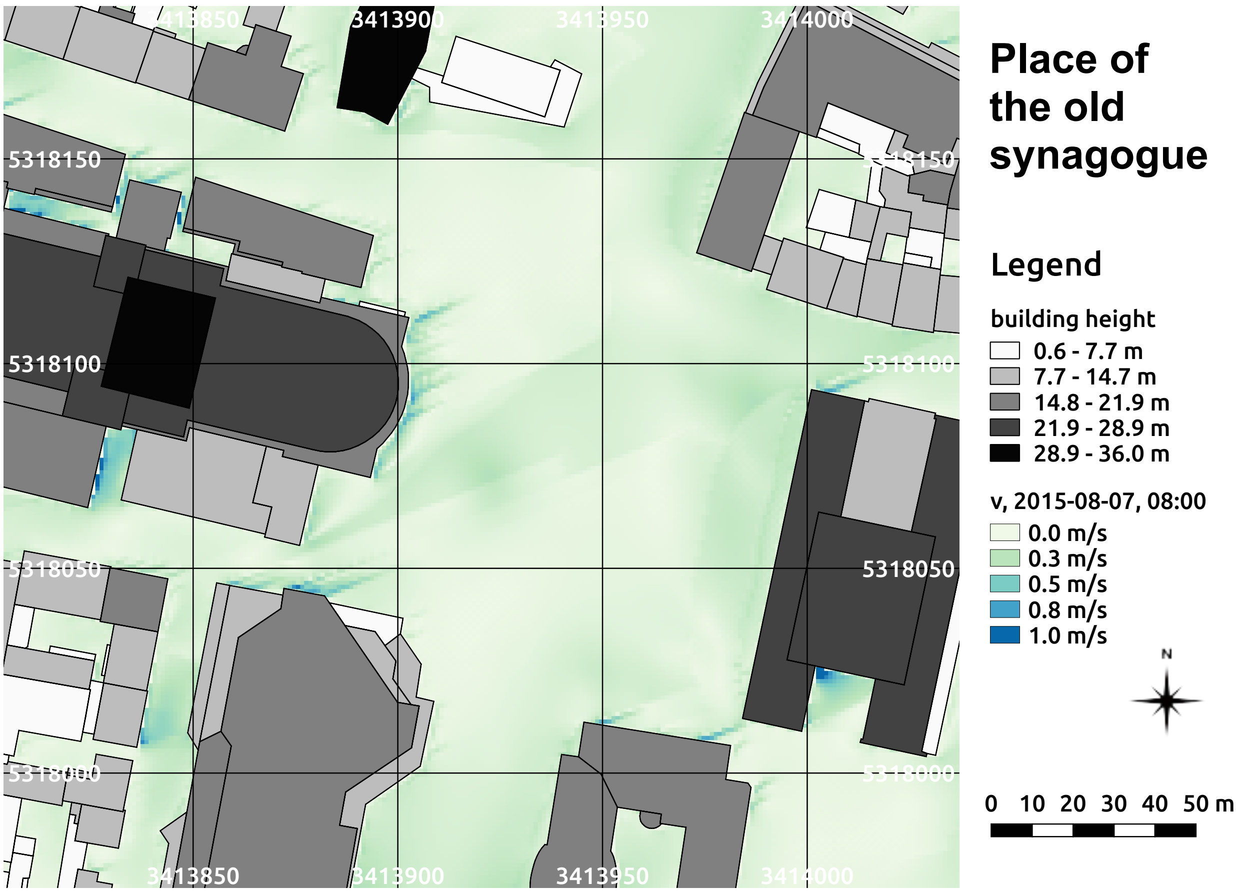


**fig. 7.2:** Spatial distribution of wind speed for the 07th of August 2015 at 08:00 in 1.5 m height at the place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Incident wind in 10 m height is 1.2 m/s from 248°. Building heights above ground (m) as well as tree properties and positions (compare to fig. 6.15) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

The 07th of August 2015 is a hot summer day with relatively low wind speed and south-west to western incident wind direction (compare to tab. 6.4).
At 08:00 in the morning, incident wind speed recorded at the urban climate station Freiburg is 1.2 m/s corrected to 10 m above ground level. Incident wind direction is 248°. The spatial distribution of wind speed at 08:00 shows quite low wind speed of less than 0.3 m/s in the central area of the place of the old synagogue (refer to fig. 7.2). It ranges in between 0.1 m/s and 0.3 m/s. The lowest wind speed of less than 0.1 m/s is calculated on the lee side of the university library and the theatre. The highest wind speed within the whole area is calculated close to the obstacles causing eddies. This can be seen north and south of the theatre, as well as in a small area west of the library. Wind

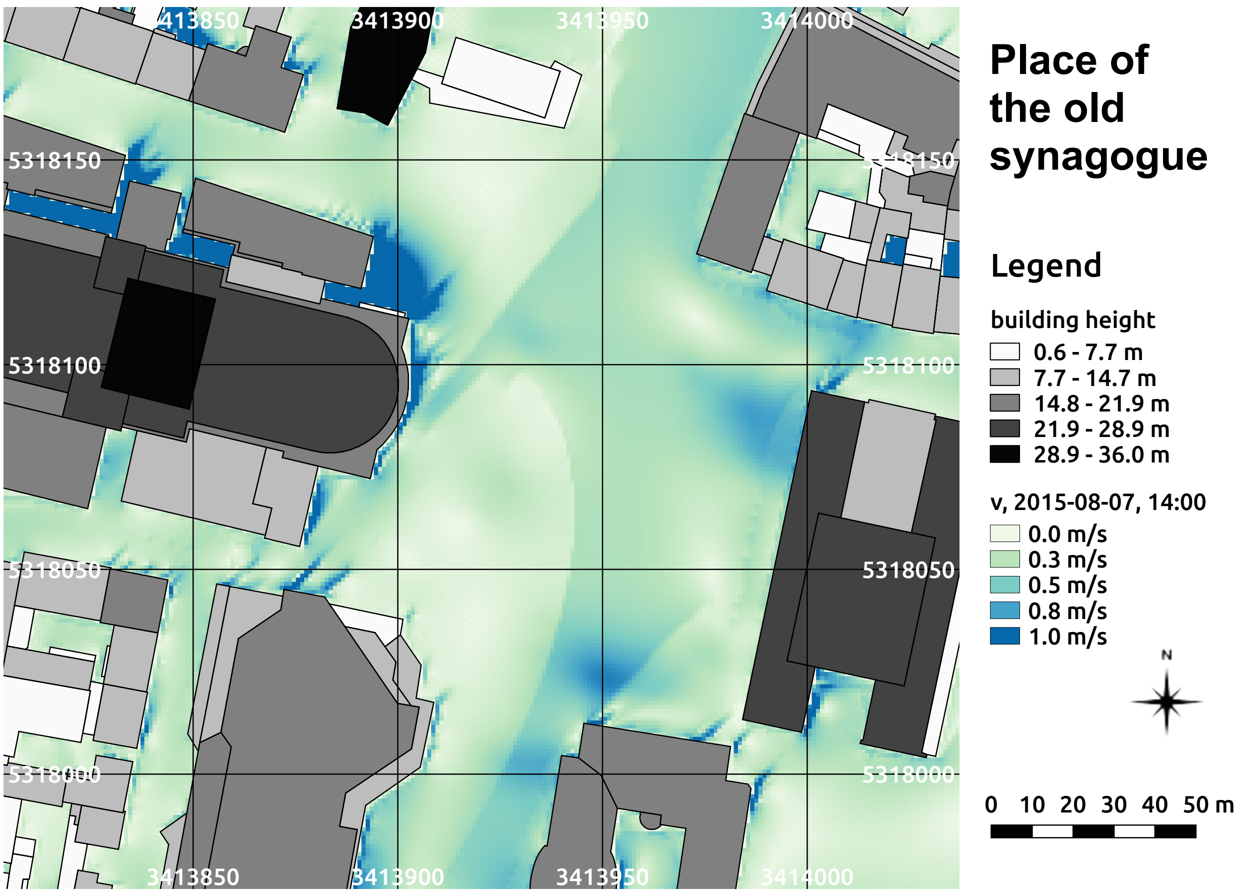


**fig. 7.3:** Spatial distribution of wind speed for the 07th of August 2015 at 14:00 in 1.5 m height at the place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Incident wind in 10 m height is 3.0 m/s from 221°. Building heights above ground (m) as well as tree properties and positions (compare to fig. 6.15) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

speed can reach values of almost 1.0 m/s there.

At 14:00 in the afternoon wind speed is still very low. However, compared to the situation at 08:00 incident wind speed increased strongly, being 3.0 m/s in 10 m height above ground now. Wind direction is still south-west (221°). The increased wind speed can be seen easily in the spatial distribution of wind speed calculated for 14:00 at the place of the old synagogue in 1.5 m above ground (see fig. 7.3).

The variation of wind speed in the central part of the place also increased a lot (compare figs. 7.2 and 7.3). At the south-western corner of the KG I (see fig. 6.15) wind speed exceeds 0.8 m/s in 1.5 m above ground, while it drops below 0.1 m/s west of the KG II. Close to some buildings, especially north of the theatre, some eddies are calculated. Wind speed exceeds 1.0 m/s there slightly.

In neither of the two spatial wind distributions (figs. 7.2 and 7.3) the trees show a lot of effect.

This is due to only the tree crowns are considered in the model. The tree crowns, however, are way higher than the target height of 1.5 m. The air flow below the trees remains, thus, almost unaffected.

### 7.1.1.3 Global radiation

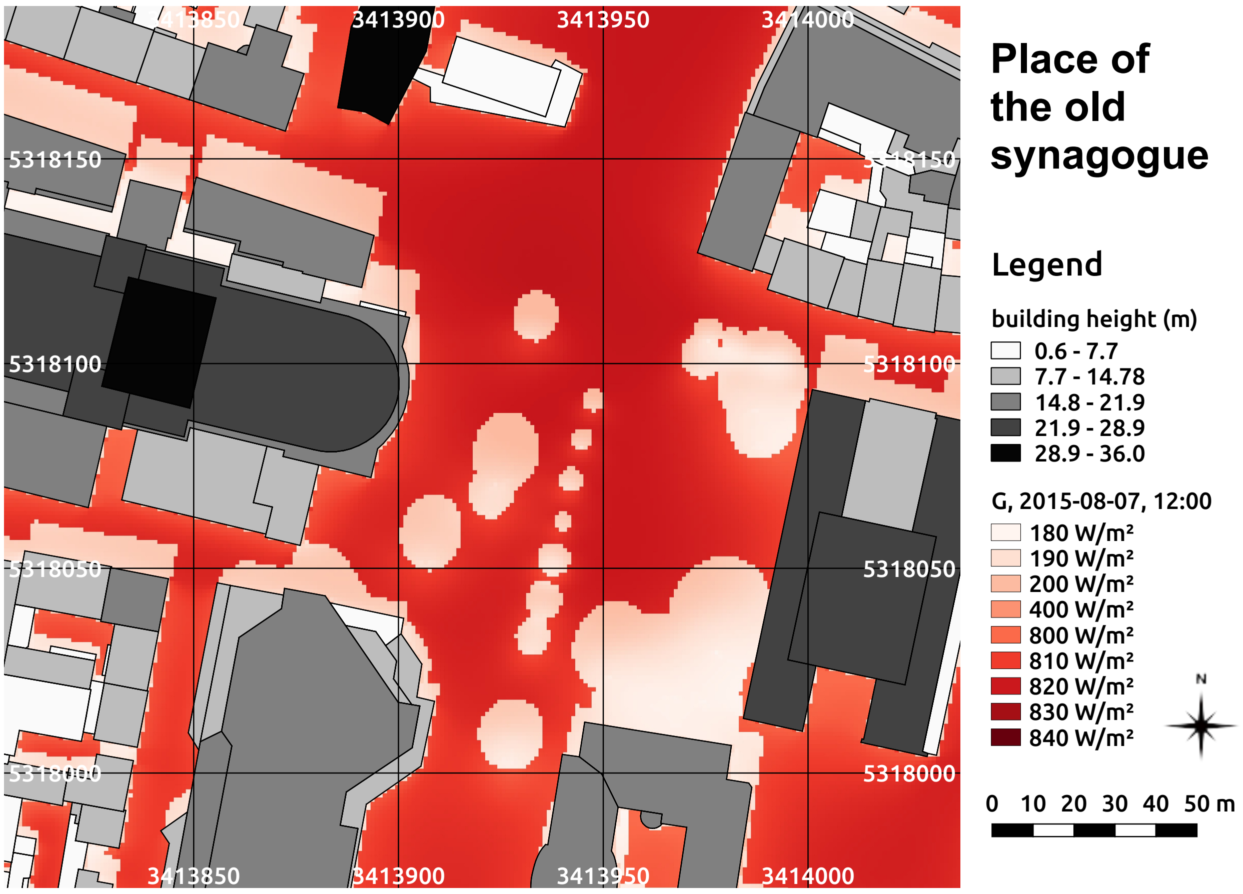


**fig. 7.4:** Spatial distribution of global radiation in $\frac{W}{m^2}$ for the 07th of August 2015 at 12:00 at the place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree properties and positions (compare to fig. 6.15) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

The 07th of August 2015 is a summer day. Global radiation at midday therefore is quite high. Looking a t the spatial distribution of global radiation at the current place of the old synagogue at 12:00 (fig. 7.4), the most dominant influence of shading on global radiation can be seen. Only if the scale is changed from a linear one to a custom scale stressing the ranges from 180 $\frac{W}{m^2}$ to 200 $\frac{W}{m^2}$ and

from 800 $\frac{W}{m^2}$ to 840 $\frac{W}{m^2}$ other effects can be seen in the spatial distribution map at all. There are no values calculated outside the two ranges for any position within the whole area of interest.
The lower range, the one from 800 $\frac{W}{m^2}$ to 840 $\frac{W}{m^2}$ applies for all the shaded locations. This is the locations north of the buildings, as well as under the trees. As trees are considered solid obstacles in terms of radiation in the SkyHelios model, there is no difference between areas shaded by trees and areas shaded by other obstacles. The lowest values of global radiation of about 180 $\frac{W}{m^2}$ are calculated, however, for locations under the trees. Areas shaded by buildings only show values of 185 $\frac{W}{m^2}$ to 195 $\frac{W}{m^2}$.
The unshaded areas show values starting from 800 $\frac{W}{m^2}$ (compare to fig. 7.4). The weaker global radiation of below 815 $\frac{W}{m^2}$ can be found close to obstacles, while the large open areas in the central part and especially in the area of the crossroads in the north of the place show values exceeding 820 $\frac{W}{m^2}$.

#### 7.1.1.4 Mean radiant temperature

The 07th of August 2015 combines both, high air temperature ($T_a$), as well as high global radiation ($G$, compare to tab. 6.4). This of course leads to a very high mean radiant temperature ($T_{mrt}$). At midday, the $T_a$ of 33.5°C and the $G$ of 841 $\frac{W}{m^2}$ (for an unlimited sky) are leading to $T_{mrt}$ of about 50.0°C all over the place of the old synagogue (compare to fig. 7.5).
In the shaded areas, both by trees and by obstacles, $T_{mrt}$ varies around 50.0°C. The unshaded areas are about 10.0°C hotter. $T_{mrt}$ ranges from 58.0°C to about 61.0°C there.
Slight variations within the unshaded areas are, besides differences in SVF leading to variation in $G$, due to wind speed influencing the surface temperature. This effect can lead to a variation in $T_{mrt}$ of up to 5.0°C, as to be seen in the street west of the university library.

#### 7.1.1.5 Perceived Temperature

The spatial determination of individual meteorological parameters, e.g. wind speed, for urban areas in a high resolution can deliver quite useful results by itself. However, the SkyHelios model is developed with the intention to assess human thermal perception. Human thermal perception can be best assessed by applying thermal indices (see section 4.3.6).
One of the commonly applied thermal indices is the Perceived Temperature (compare to section 4.3.6.1). It is determined for the current place of the old synagogue in 1 m resolution for the four times present in the meteorological data input file (tab. 6.4) using the SkyHelios model.
As the 07th of August 2015 was a very hot summer day, the spatial distribution of PT is shown by figs. 7.11 to 7.14. using a custom color scale instead of the one generated after the thermo-physiological assessment table by (Staiger et al. 2012, tab. 4.3).

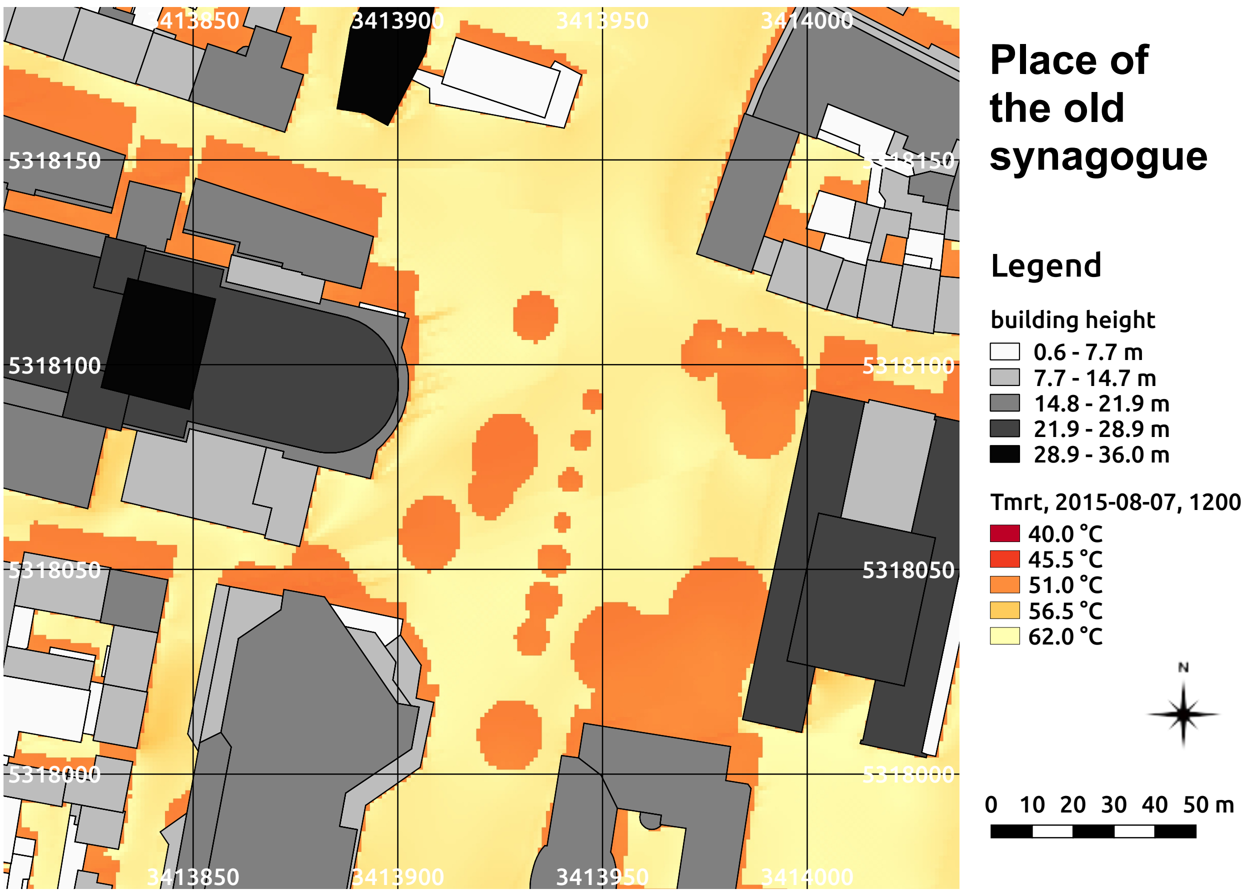


**fig. 7.5:** Spatial distribution of the mean radiant temperature in °C for the 07th of August 2015 at 12:00 at the place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree properties and positions (compare to fig. 6.15) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

At 08:00 in the morning, the current place of the old synagogue is already fairly warm (compare to fig. 7.11). Within the shaded areas PT ranges from 27.2°C to 29.9°C. According to the assessment table after (Staiger et al. 2012, tab. 4.3), this matches the class "warm" (26°C to 32°C), that causes moderate heat stress. Areas exposed to direct sunlight show PT of 33.4°C to 34.4°C. This applies to the class "hot" (32°C to 38°C) representing great heat stress. While both is quite high for 08:00 in the morning, it is in perfect agreement to the result calculated for PET (see below).
The most dominant influence on PT in fig. 7.11 is obviously shading by buildings and trees. As trees are represented by solid matter in the SkyHelios model concerning radiation calculations, there is no difference between areas shaded by trees and such shaded by buildings. The second stronges influence in fig. 7.11 can be identified as wind speed (compare to fig. 7.2).
Until midday, the current place of the old synagogue heated up a lot (compare to fig. 7.12). The

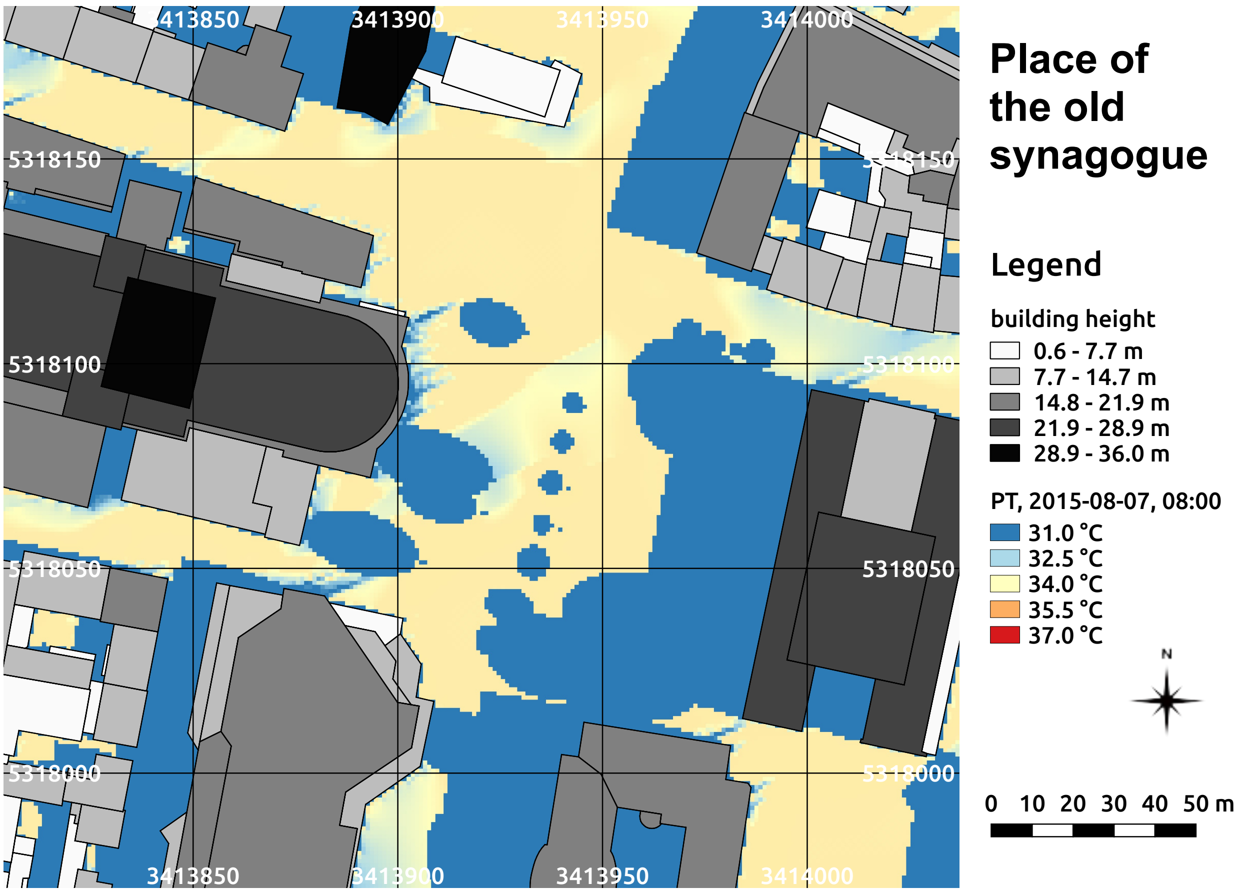


**fig. 7.6:** PT on the 07th of August 2015 at 08:00 at the place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree locations and positions are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

increased wind speed at 12:00 is not able to even out the increased air temperature ($T_a$) and global radiation (compare to tab. 6.4). The shaded areas now show PT in between 32.6°C and 34.3°C. According to tab. 4.3, this stands for "hot" conditions, leading to great heat stress.
More uncomfortable are the unshaded areas. PT ranges from 34.6°C to 36.9°C there. In spite this is more uncomfortable than in the shaded areas, it is still the same grade of thermal perception and thermo-physiological stress, as 38.0°C is not exceeded. The highest class of the thermal perception table is therefore not reached. This disagrees to the results calculated for PET predicting extreme heat stress at 12:00 in some areas exposed to direct sunlight.
Two hours later, compared to 12:00 the global radiation is slightly decreased (compare to tab. 6.4). However, $T_a$ is slightly higher than at 12:00 and wind speed is little decreased as well. This leads to even hotter conditions on the place of the old synagogue than at 12:00 (see fig. 7.13). The shaded

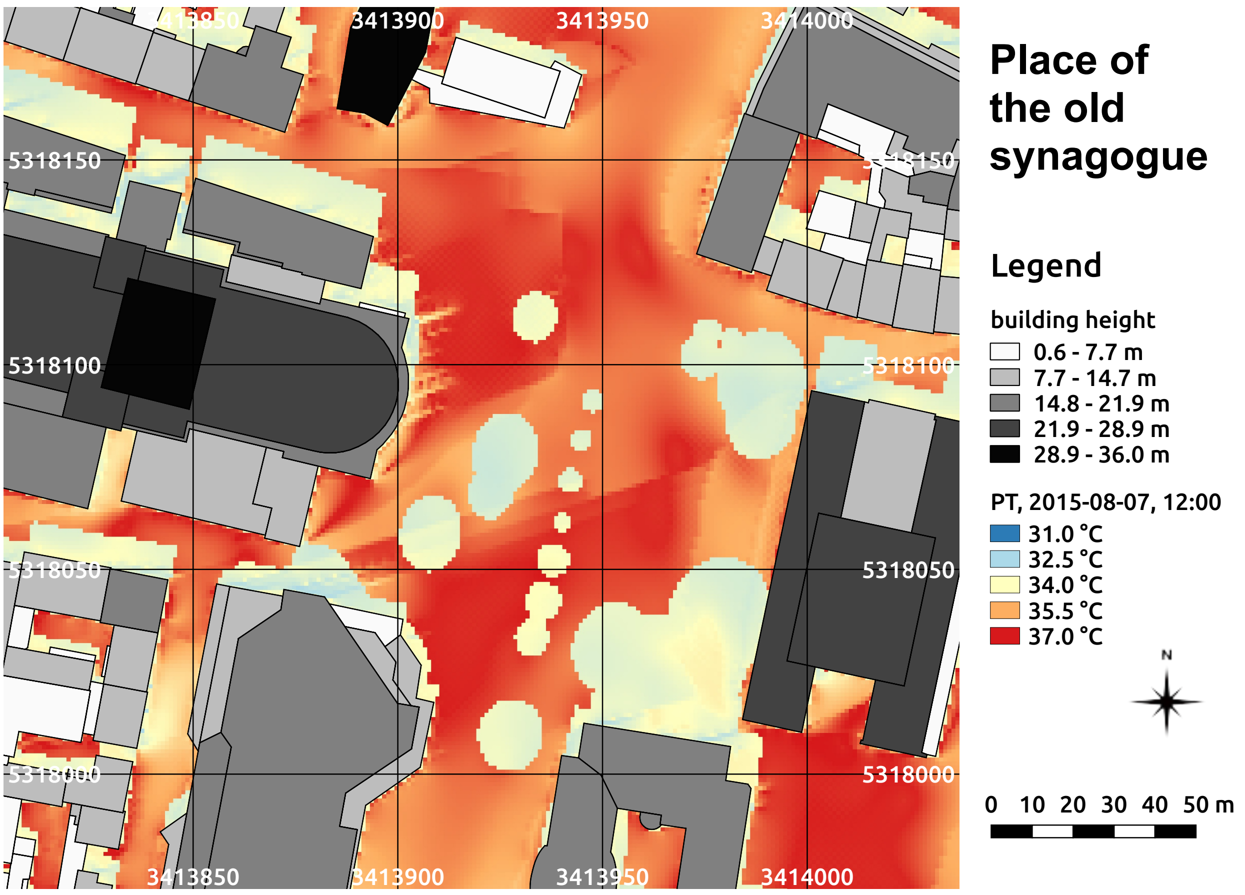


**fig. 7.7:** PT on the 07th of August 2015 at 12:00 at the place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree locations and positions are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

areas are now reaching PT of 33.5°C to 35.0°C. According to the thermo-physiological assessment table (tab. 4.3) the shaded areas provide "hot" conditions causing great heat stress.
The unshaded areas are of course way warmer. In areas exposed to the sun PT ranges in between 35.7°C and 36.8°C. In spite this is way warmer, it is still matching the same class of the assessment table. All locations within the whole area of interest are therefore to be considered "hot" causing great heat stress on the 07th of August 2015 at 14:00.
Another two hours later, the global radiation is way lower (compare to tab. 6.4). In spite $T_a$ is, again, increased and wind speed is decreased a lot, conditions improved regarding PT at the place of the old synagogue at 16:00 (compare to fig. 7.14). The shaded areas cooled down to PT of 31.5°C to 32.7°C. This means, that parts of the shaded areas are now part of the class "warm", only causing moderate heat stress according to tab. 4.3. Shaded locations with lower wind speed are remaining

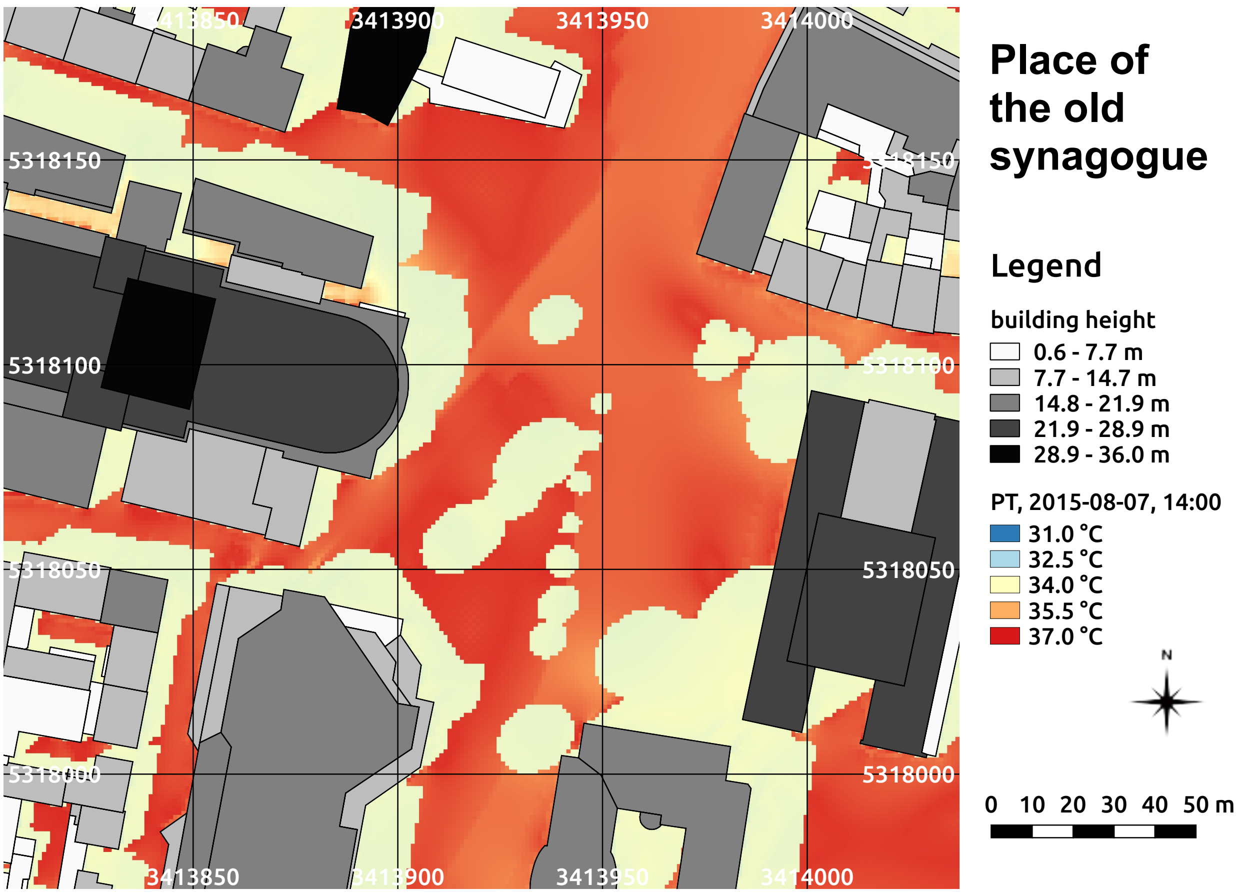


**fig. 7.8:** PT on the 07th of August 2015 at 14:00. at the place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree locations and positions are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

part of the class "hot".
The same holds for all the areas outside the shade. The locations exposed to direct solar irradiation show PT of 33.8°C to 34.0°C and are, thus, causing great heat stress. Surprising about the distribution of PT at 16:00 is the small variation of PT in locations exposed to the sun of only 0.2°C. Compared to the results for PET (see below) it has to be noted, that results for PT according to tab. 4.3 represent thermo-physiological stress that it one class more comfortable than predicted by PET using the according classification (tab. 4.6).

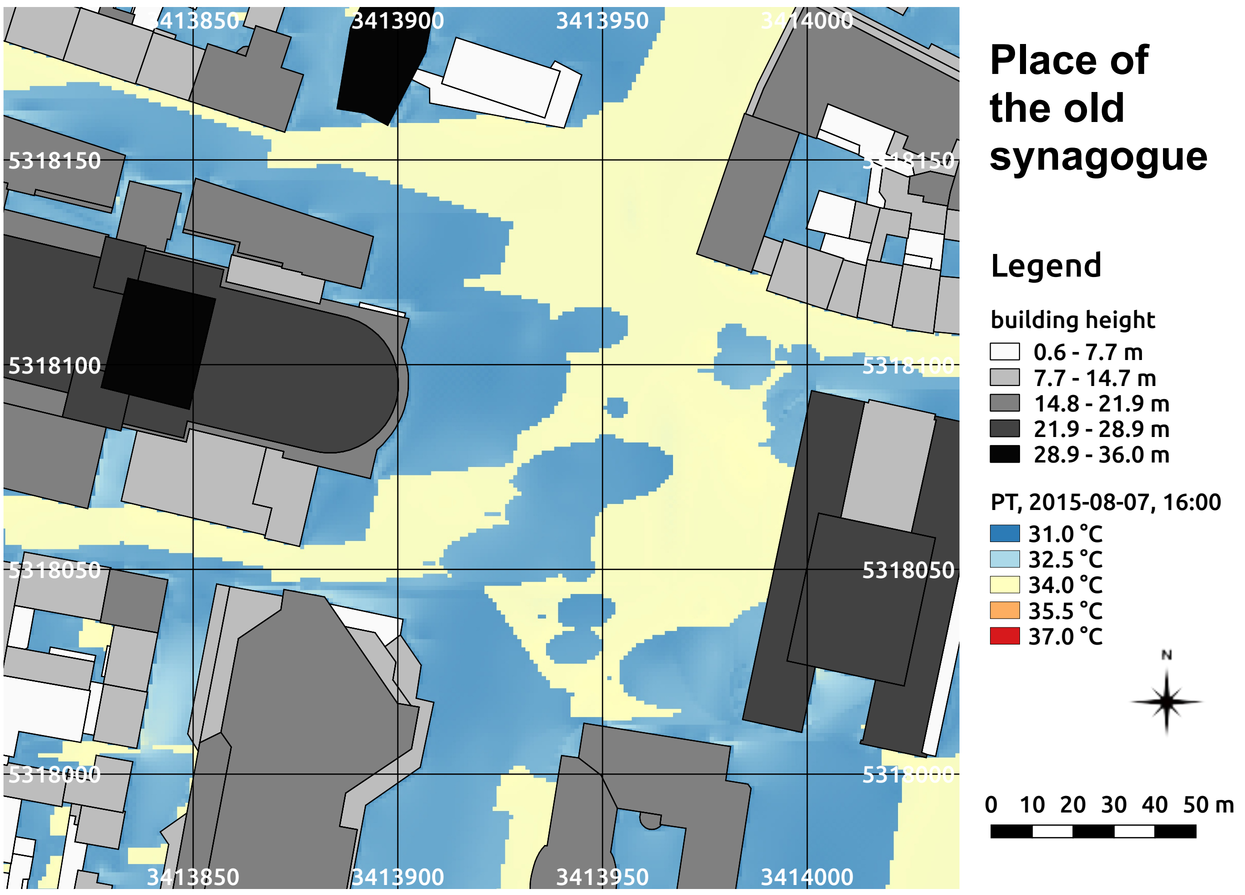


**fig. 7.9:** PT on the 07$^{th}$ of August 2015 at 16:00 at the place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree locations and positions are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

### 7.1.1.6 Universal Thermal Climate Index

The SkyHelios model is capable of calculation the relatively new thermal index "Universal Thermal Climate Index" (see section 4.3.6.2). This new functionality is tested for the place of the old synagogue (compare to fig. 7.10). For the meteorological input conditions, the 07$^{th}$ of August 2015 at 12:00 is selected. This date and time is chosen to be able to compare the results to those calculated by other indices (e.g. with PET, fig. 7.12). The decision for the time was due to wind speed being the highest at 12:00 among the four situations. As the range of valid input conditions for the UTCI calculations is rather narrow (see section 4.3.6.2), this time was expected to generate the smallest amount of fail values.

Looking at fig. 7.10 it is obvious, that the area is dominated by fail values generated due to wind

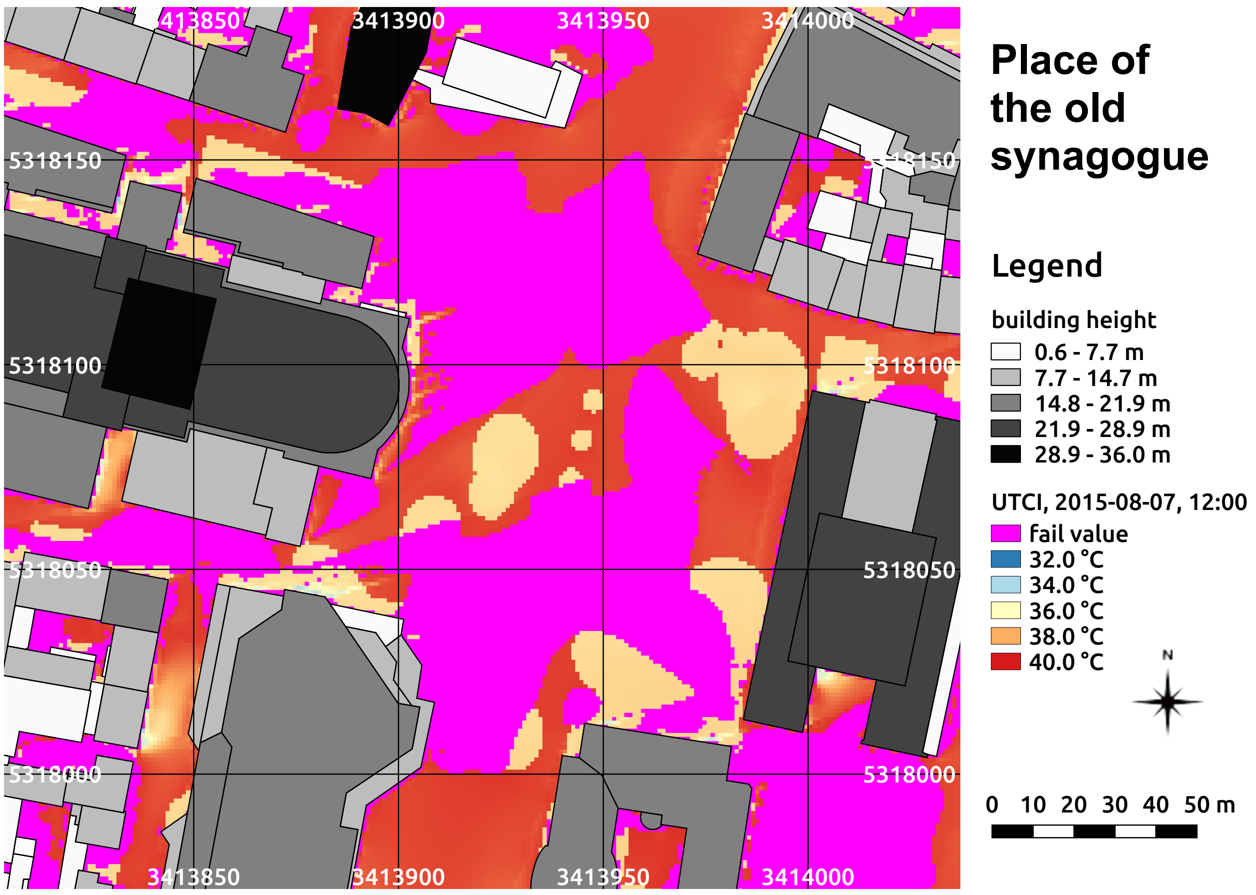


**fig. 7.10:** UTCI on the 07th of August 2015 at 12:00 at the place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree locations and positions are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

speed (extrapolated to 10 m above ground level using the vertical profile provided by Havenith et al. (2012)) below 0.5 m/s. In areas with sufficient wind speed UTCI ranges in between 35.3°C and 38.8°C in shaded locations and 38.4°C to 39.5°C at locations exposed to direct solar irradiation. Applying the assessment scale after Błażejczyk et al. (2013, see tab. 4.4), this means strong heat stress in most of the shaded locations. Only some are exceeding 38.0°C and, thus, are causing very strong heat stress. The locations without shading are matching the thermal stress category "very strong heat stress" without any exception. It, thus, better agrees to the results of PT (see above), than those of PET (see below).

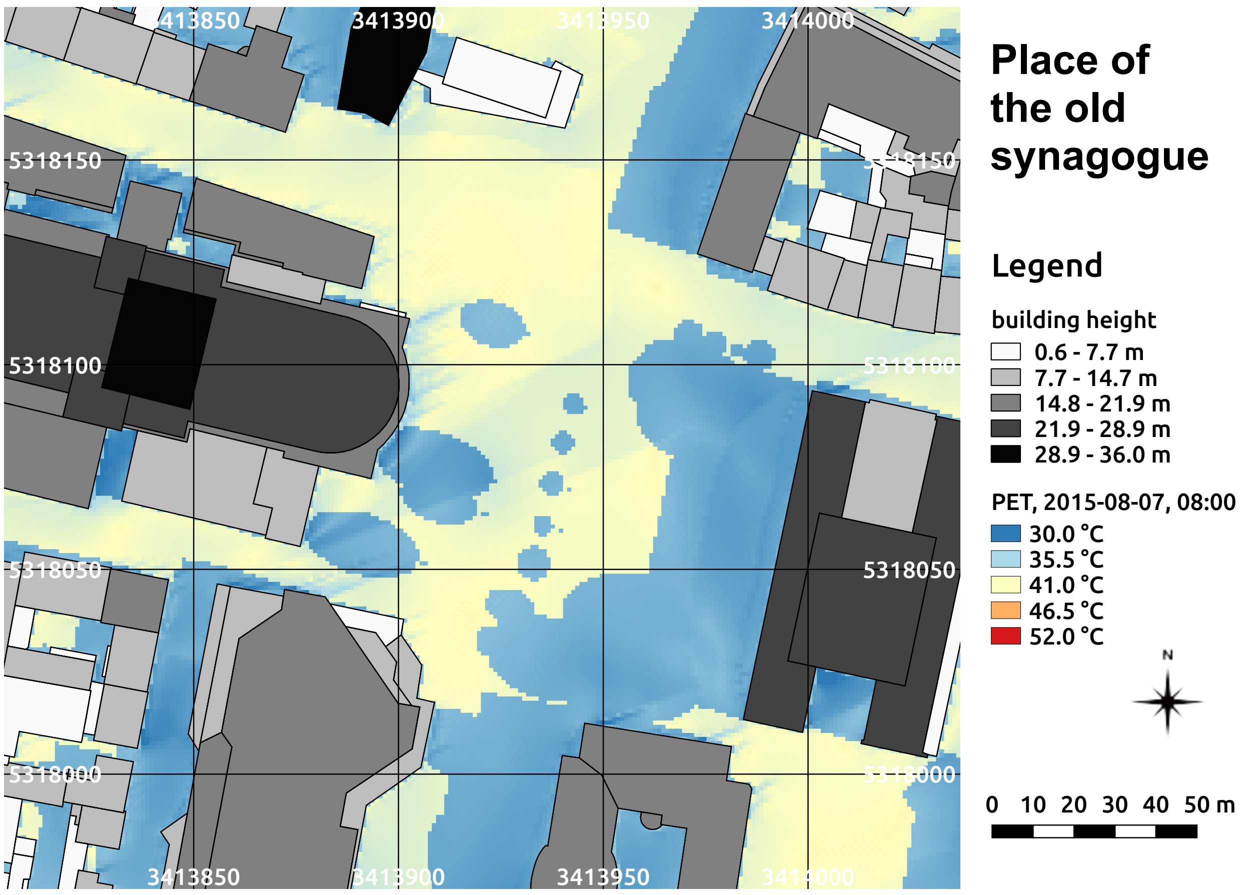


**fig. 7.11:** PET on the 07th of August 2015 at 08:00 at the place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree locations and positions are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

#### 7.1.1.7 Physiologically Equivalent Temperature

The SkyHelios model is primarily designed to estimate human thermal perception within urban areas spatially. This can be done using thermal indices, e.g. the Physiologically Equivalent Temperature (PET, refer to section 4.3.6.3).

As stated above, the 07th of August 2015 was a very hot summer day. This on the one hand leads to high values for $T_{mrt}$ (see above). On the other hand, very low wind speed is even more increasing heat discomfort, as shown by figs. 7.11 to 7.14.

At 08:00 in the morning, PET is already fairly high all over the place of the old synagogue (see fig. 7.11). In spite air temperature ($T_a$) is only 25.4°C and global radiation is low at that time of day (compare to tab. 6.4), the low wind speed already leads to fairly high PET. Within the shaded

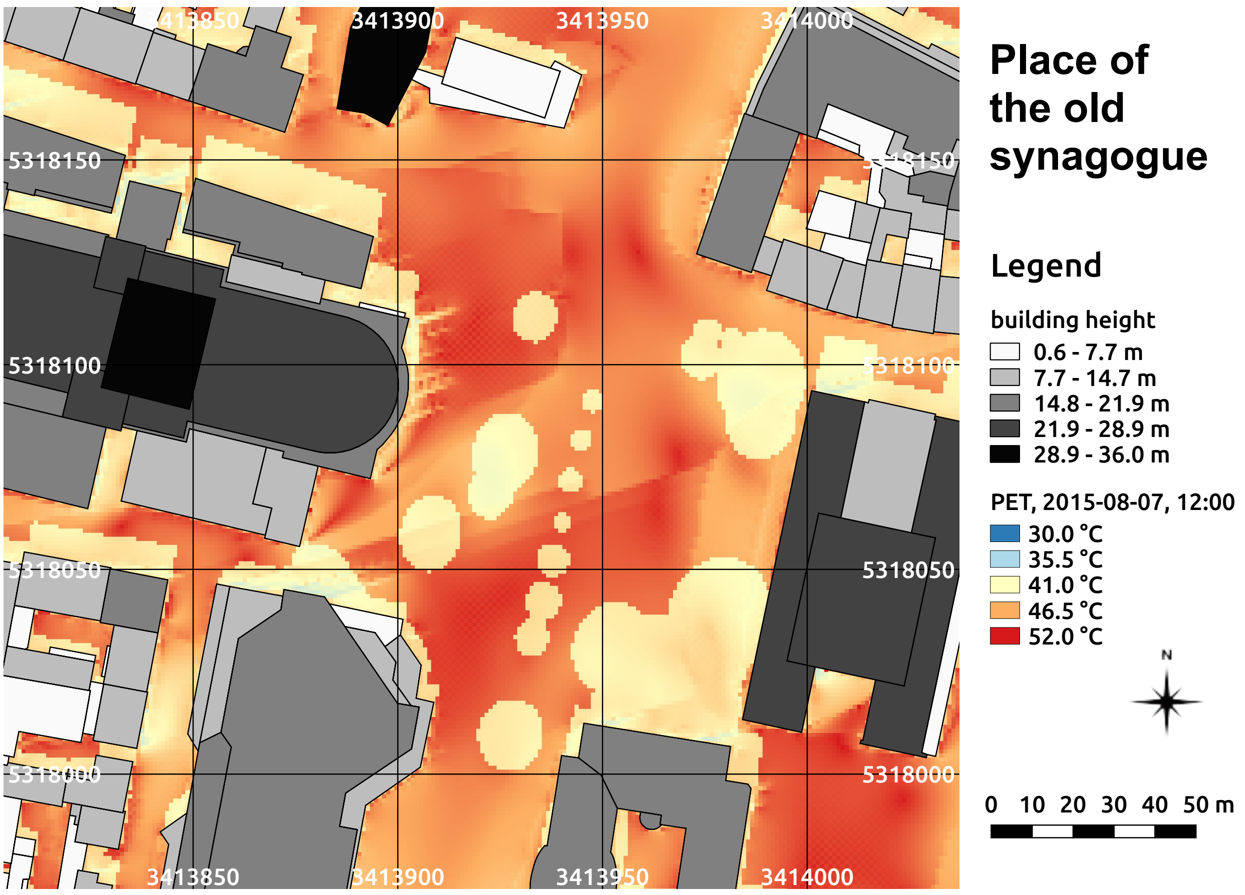


**fig. 7.12:** PET on the 07th of August 2015 at 12:00 at the place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree locations and positions are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

areas, PET ranges in between 30.1°C and 33.5°C. According to the thermal sensation classes after Matzarakis and Mayer (1996, compare to tab. 4.6), this already stands for moderate heat stress. Outside the shaded areas, PET ranges from 38.2°C to 40.8°C. All of this is within the class "hot", representing strong heat stress (Matzarakis and Mayer 1996). The difference between sunny and shady conditions therefore is approximately one thermal perception class. Even though it is only 08:00 in the morning, all values already are within the classes of the upper third of the thermal sensation classification (compare to tab. 4.6).

Until midday, the thermal conditions on the place of the old synagogue are becoming more extreme. This is mostly due to an increased $T_a$ of 33.5°C and strongly increased global radiation of 841 $\frac{W}{m^2}$ (compare to tab. 6.4). At 12:00, even the shaded areas are bearing PET of 39.7°C to 43.6°C (see fig. 7.12). That means, that a human being even within the "cooler" spots in the shaded areas (the

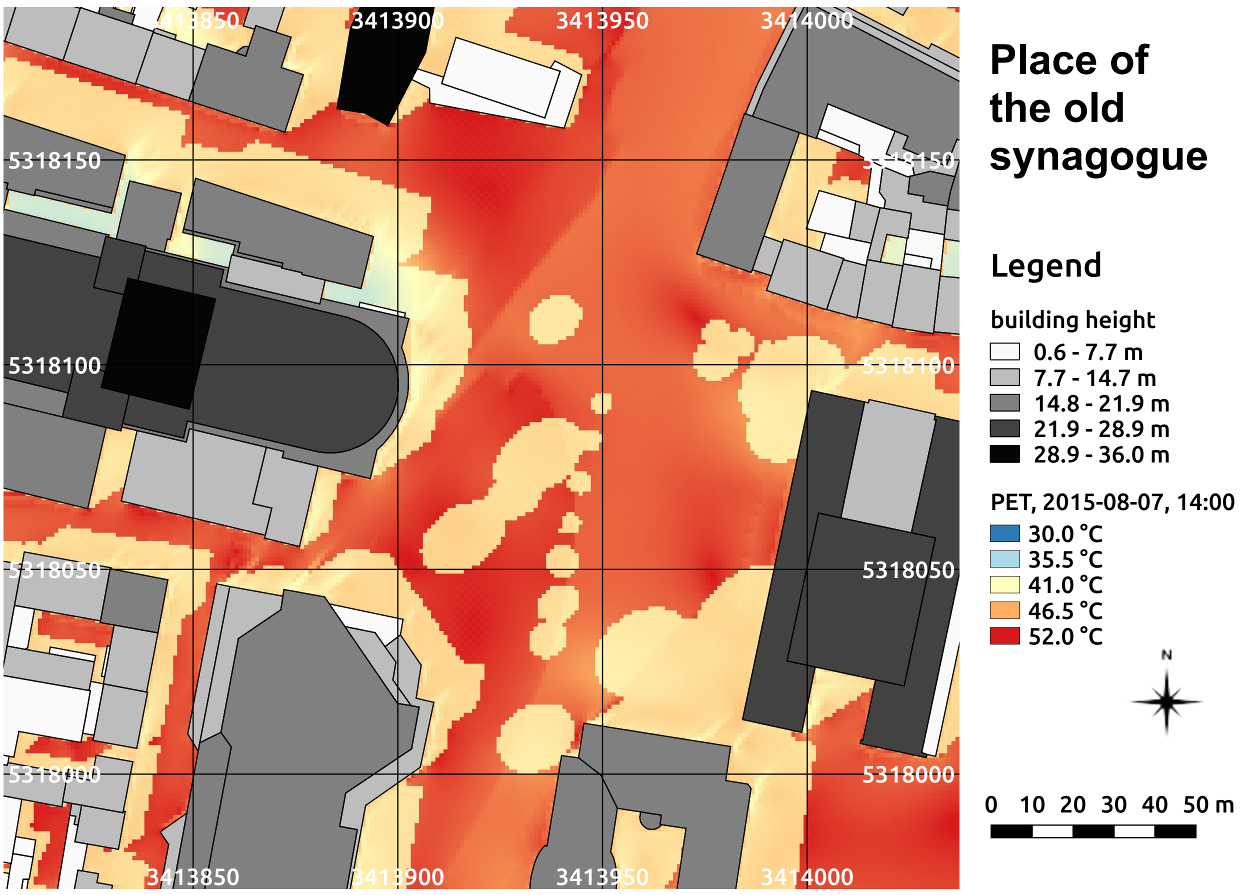


**fig. 7.13:** PET on the 07$^{th}$ of August 2015 at 14:00. at the place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree locations and positions are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

location with higher wind speed), will experience strong heat stress according to tab. 4.6. In the shaded areas with lower wind speed, 41.0°C are exceeded, leading to extreme heat stress.
Outside the shaded areas, especially in the central part of the place of the old synagogue, as well as on the place of the white rose (south-eastern corner of the area of interest), way hotter conditions are calculated (compare to fig. 7.12). PET approaches values of 45.7°C to 50.8°C in areas exposed to direct sunlight.
Another two hours later, the place heated up even more (compare to fig. 7.13). At 14:00 $T_a$ again increased compared to the 12:00 situation by 2.0°C, while incident wind speed in 10 m is slightly lower (compare to tab. 6.4). Also global irradiation is slightly decreased. This leads to only very slightly higher values of PET for 14:00 than at 12:00.
In the shaded areas PET ranges between 41.2°C and 43.7°C. In both cases 41.0°C are exceeded

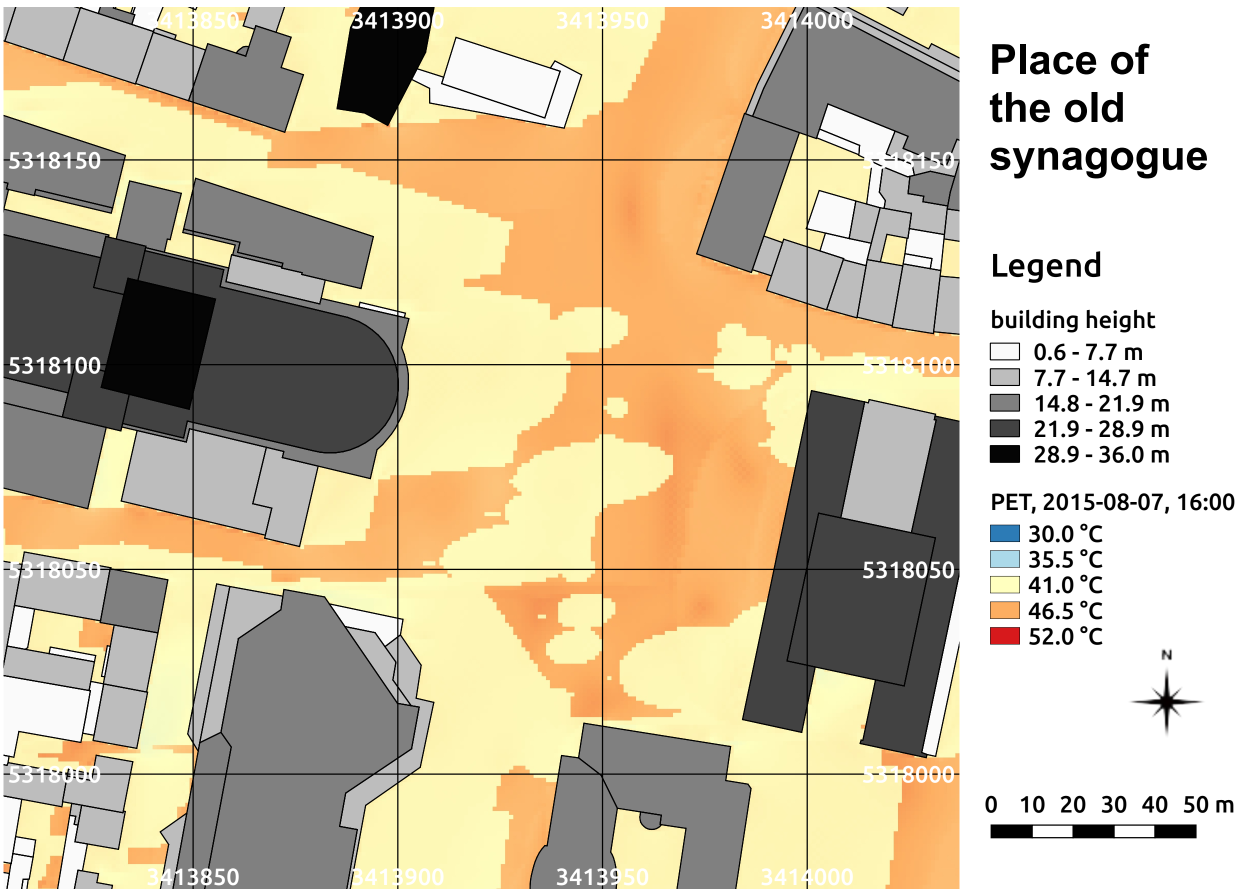


**fig. 7.14:** PET on the 07$^{th}$ of August 2015 at 16:00 at the place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree locations and positions are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

and extreme heat stress is present. This of course also holds for all locations outside the shade, where PET varies in between 48.6°C and 51.8°C depending on wind speed. Compared to 12:00 the situation on the place of the old synagogue improved a little anyway, as the shaded areas are already a little larger due to the lower sun elevation at 14:00.
At 16:00 in the afternoon the prevailing conditions are already way cooler (compare to fig. 7.14). In spite $T_a$ increased even more (compared to 14:00) and now reaches 36.9°C, the lower global radiation of only 590$\frac{W}{m^2}$ (see to tab. 6.4) leads to significantly lower PET. Within the shaded areas PET only reaches 40.3°C to 41.7°C. Depending on wind speed, a human being within the shaded areas would suffer from strong heat stress or extreme heat stress. Outside the shaded area even at 16:00 only the latter class is present. PET ranges from 45.3°C to 47.1°C there. Compared to the situation at 14:00, the areas exposed to direct sunlight are, however, already way smaller due to the

lower solar elevation angle.

### 7.1.2 Redesigned place of the old synagogue

The human thermal bioclimate in the redesigned place of the old synagogue was analysed based on the thermal indices PT and PET calculated by the SkyHelios model. The area of interest and the date and time are the same as for the old place of the old synagogue (see section 7.1.1) to allow for a direct comparison of the effect of the two designs on human thermal perception.
The university library in the South-Western corner of the place of the old synagogue was replaced by a new building during the redesign of the place. To consider the new library, the old building was removed from the shapefile and the southern part of the new library building was added to the obstacle file holding the trees (This is due to obstacle files are allowing for buildings with differing base and roof outline). As obstacle files can not be displayed in GIS, the library, as well as the building in the North of the place are not present in the distribution maps for the redesigned place of the old synagogue. However, their impact can be seen.

#### 7.1.2.1 Perceived Temperature

The first index to be compared is the Perceived Temperature (compare to section 4.3.6.1). It is determined for the redesigned place of the old synagogue in 1 m resolution for the four times present in the meteorological data input file (tab. 6.4) using the SkyHelios model.
At 08:00 in the morning, the redesigned place of the old synagogue is already fairly warm (compare to fig. 7.15). The absolute values for PT for shaded and unshaded locations are not differing a lot. However, the distribution and the size of the shaded areas changed during the redesign (compare fig. 7.6 to fig. 7.15).
As most of the new trees are located west of the KG II, they are in positions, that are shaded by the building anyway. Some of the trees that are removed during the redesign, especially the ones along the Rotteckring, are already providing shade at 08:00 before the redesign. This locations now are exposed to direct sunlight. The only new tree providing shade at this time of day is the one North-East of the theatre. At 08:00 the area exposed to direct sunlight therefore is increased by the redesign.

At midday, the place o the old synagogue heated up a lot. It is now dominated by PT exceeding 35.5°C (compare to fig. 7.16). Comparing the situation before (fig. 7.7) and after the redesign (fig. 7.16) one can see, that at 12:00 the area of shaded and unshaded locations is almost the same. However, due to the removed trees along the Rotteckring, the central area without shading is larger

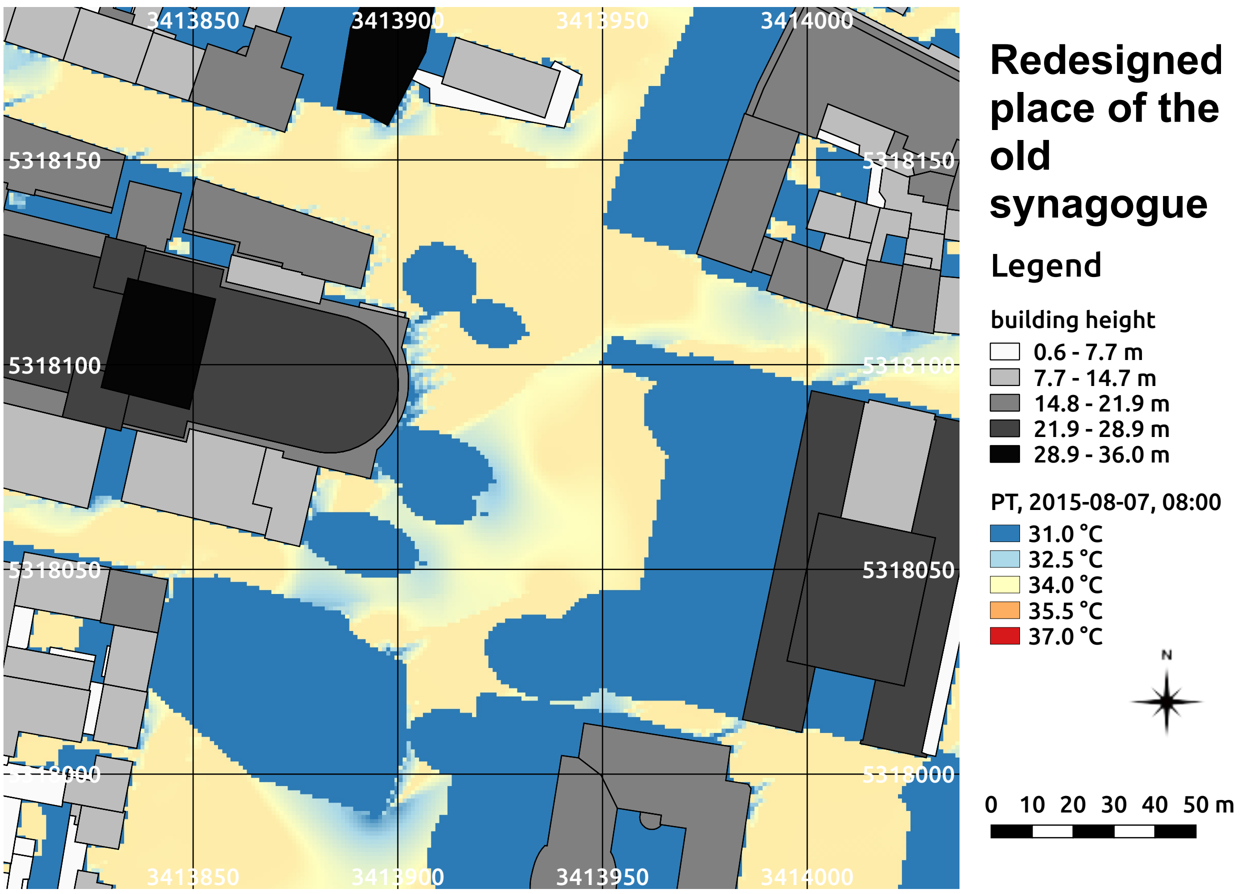


**fig. 7.15:** PT on the 07th of August 2015 at 08:00 at the redesigned place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree locations and positions are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

after the redesign.
Absolute values for PT inside the shaded, as well as the unshaded locations is decreased slightly after the redesign. This is due to an increase in wind speed. However, this is not caused by the redesign of the place of the old synagogue, but due to the new university library causing less wind speed reduction for this wind direction.

Two hours later, at 14:00, the redesigned place of the old synagogue is even a little hotter than at 12:00 (compare fig. 7.16 to fig. 7.17). Generally, PT is increased by approximately 0.3°C between 12:00 and 14:00, what is comparable to the increase calculated for the old place of the old synagogue. However, while PT is increased comparing shaded to shaded areas and unshaded to unshaded areas, the shaded areas are already significantly larger at 14:00.

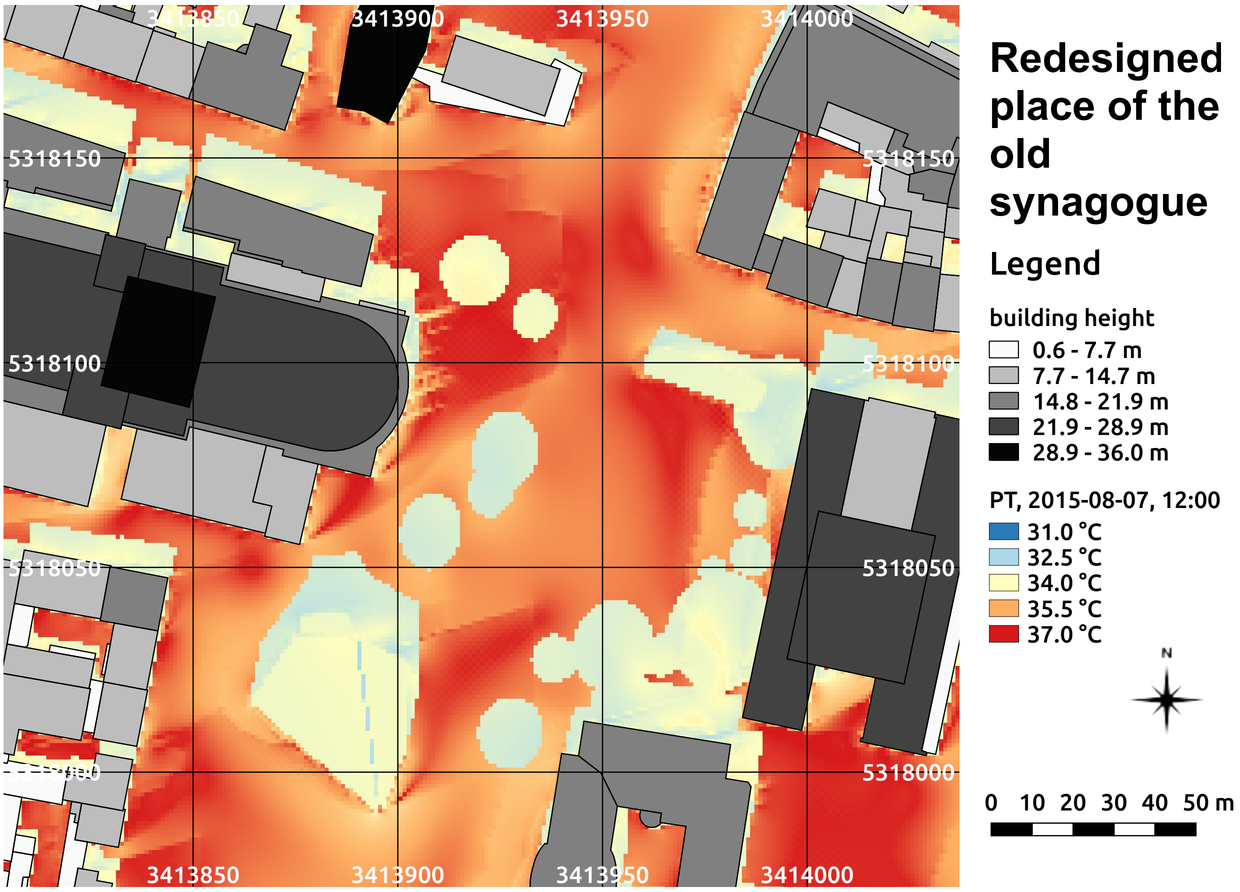


**fig. 7.16:** PT on the 07th of August 2015 at 12:00 at the redesigned place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree locations and positions are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

Comparing the old place (fig. 7.8) to the redesigned one, again, the effect of the differently shaped university library in the South-West of the place can be seen, decreasing PT due to an increase in wind speed. Also the steeper design of the library building enlarges the area shaded by it. However, as the shaded area caused by individual obstacles increases due to a lower solar elevation angle, the lack of shading in the central area of the place also becomes more significant.

Another two hours later, thermal stress has decreased a lot all over the place (see fig. 7.18). As wind speed is lower at 16:00 (compare to tab. 6.4), the effect of the lesser wind speed reduction caused by the new library is weaker. The difference in PT comparing the shaded to the shaded and the unshaded to the unshaded areas is therefore insubstantial.
The direct comparison of PT at the old (see fig. 7.9) to PT at the redesigned place of the old

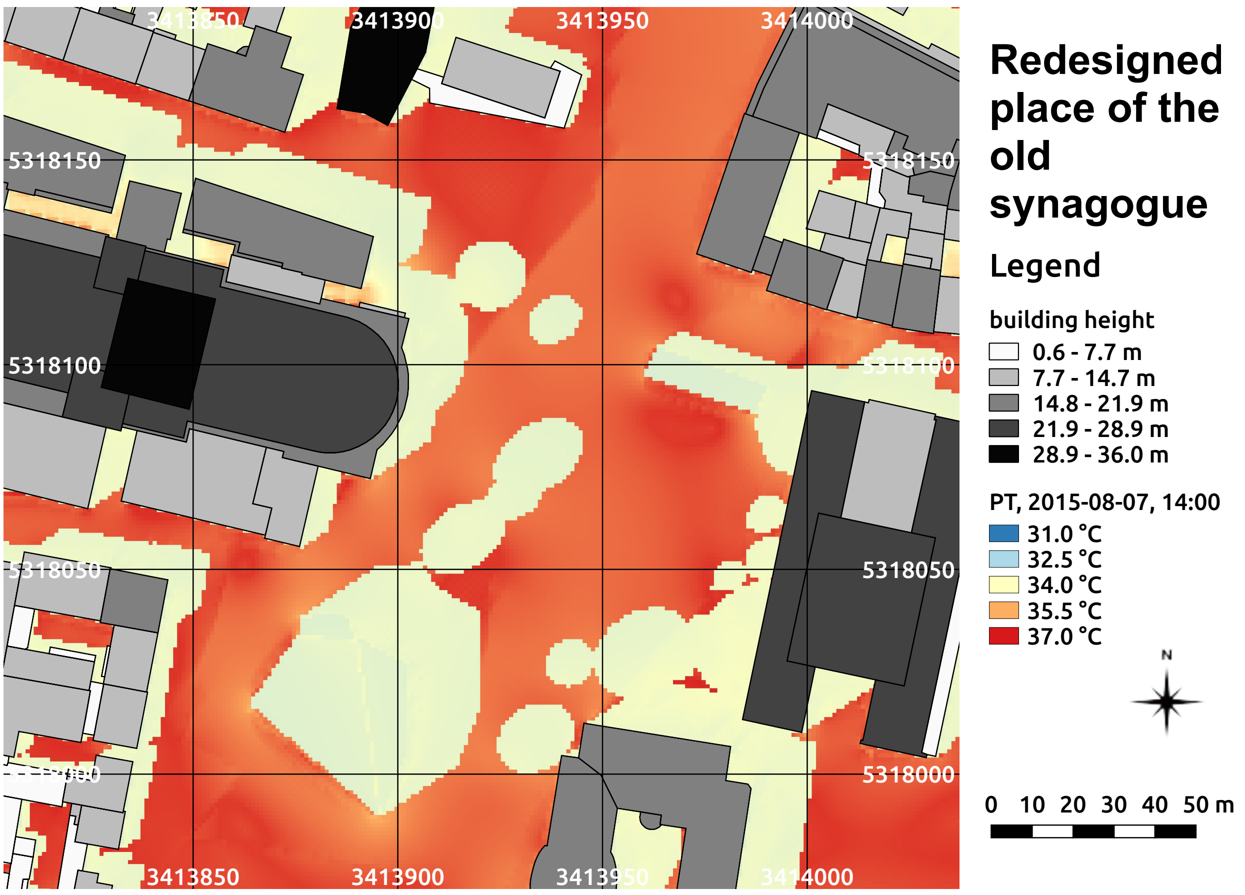


**fig. 7.17:** PT on the 07th of August 2015 at 14:00 at the redesigned place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree locations and positions are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

synagogue (fig. 7.18) based on the same meteorological input conditions at 16:00 shows only very few difference between the size of the shaded and unshaded areas. The sun is already that low, that the two trees East of the theatre are shading most of those in the center of the place (compare fig. 7.9 to fig. 7.18). The new trees planted West of the KG II, however, are mostly shading the building and therefore only show very weak effect on the thermal conditions on the place.

### 7.1.2.2 Physiologically Equivalent Temperature

The two designs of the place of the old synagogue were also compared using the Physiologically Equivalent Temperature (PET, see section 4.3.6.3 for details). PET predicts quite hot conditions for

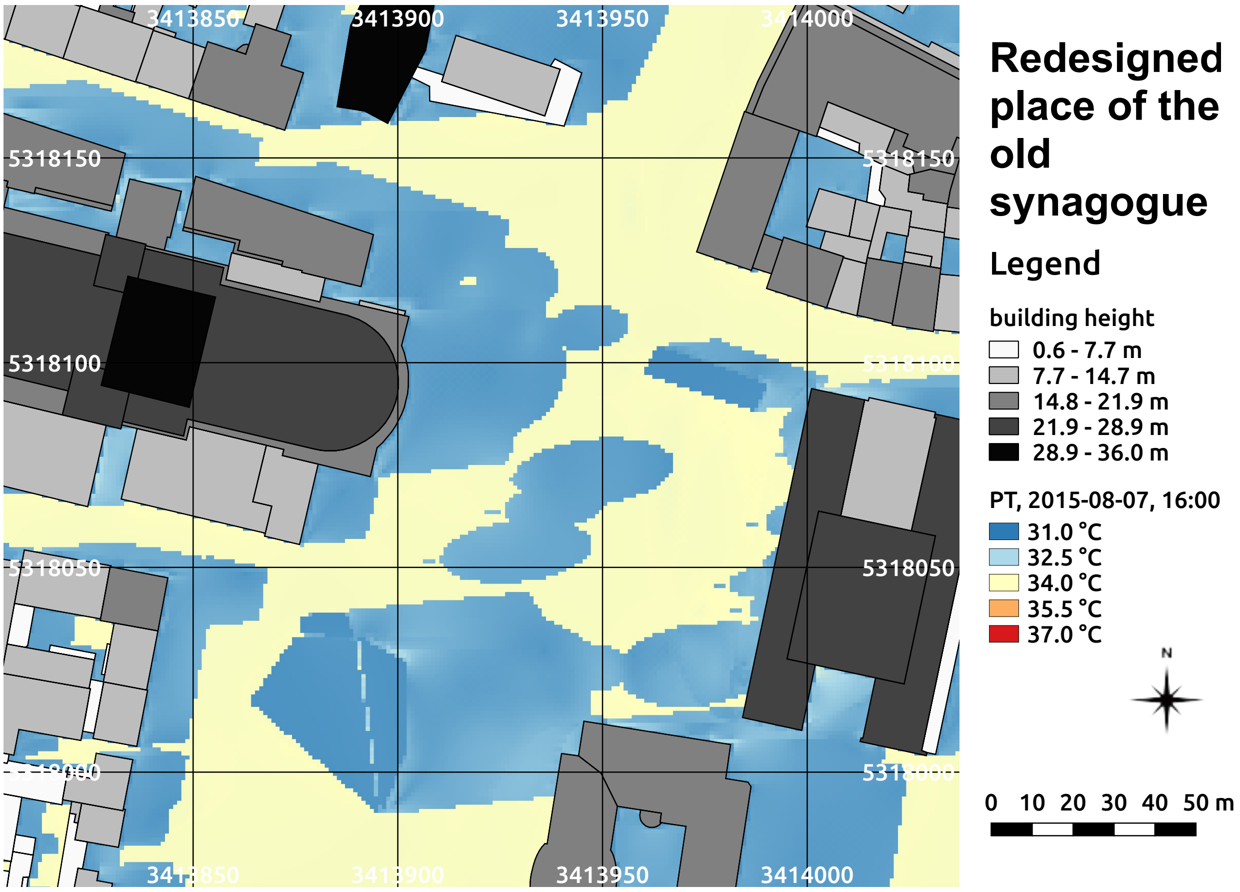


**fig. 7.18:** PT on the 07th of August 2015 at 16:00 at the redesigned place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree locations and positions are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

both, the old (fig. 7.11) as well as for the redesigned place (fig. 7.19) on the 07th of August 2015 at 08:00. The direct comparison shows, that the old university library causes slightly hotter conditions on its North-East side by providing larger wind shelter. The new university library shows a stronger air flow around it and, thus, reduces PET slightly by approximately 1.5°C. This effect, however, is small compared to the effect of the trees in the central part of the place of the old synagogue, that will be removed during the redesign.

In areas, where they are providing shade on the old place, PET is around 33.3°C. This is exceeded a lot by the redesigned place on which the same areas, now exposed to direct sunlight, are, with PET of approximately 39.5°C 6.3°C, hotter. The new small trees added West of the KG II in the East of the place of the old synagogue are shaded by the building and therefore show almost no effect on the thermal conditions.

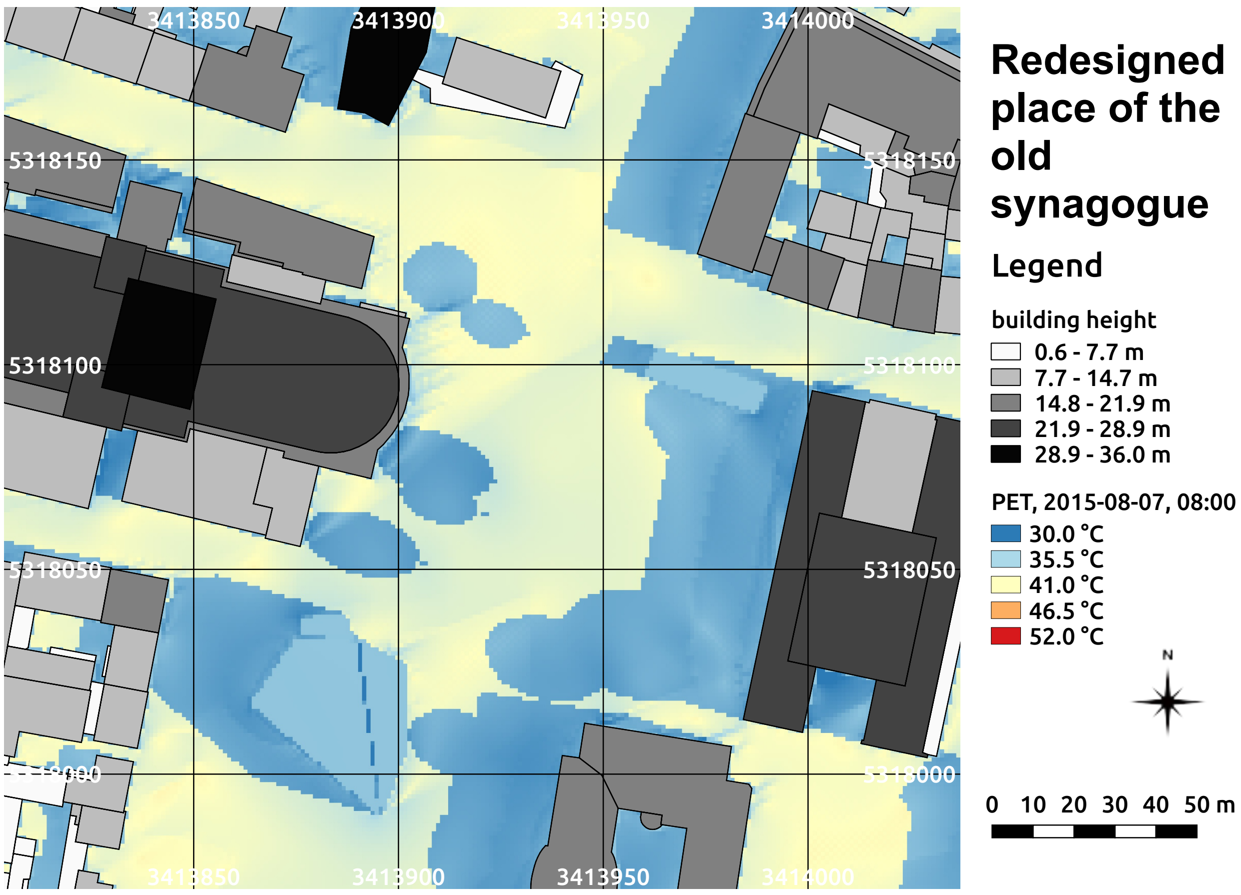


**fig. 7.19:** PET on the 07$^{th}$ of August 2015 at 08:00 at the redesigned place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree locations and positions are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

At 12:00 the higher sun elevations causes way hotter conditions than at 08:00 due to highly increased radiation. In spite wind speed is increased as well, PET already exceeds 41.0°C in vast areas (compare to fig. 7.20).

Comparing the midday situation for the old place of the old synagogue to the new one assessed by PET, again, the beneficial effect of the higher wind velocity North of the university library can be seen. This effect increased strongly since 08:00 in the morning. It now leads to PET reduced by 3.8°C due to the new library's design.

The trees in the center of the old place removed by the redesign are causing a larger area with thermally extremely stressful conditions in the center of the place. This can hardly be counterbalanced by the new small trees West of the KG II.

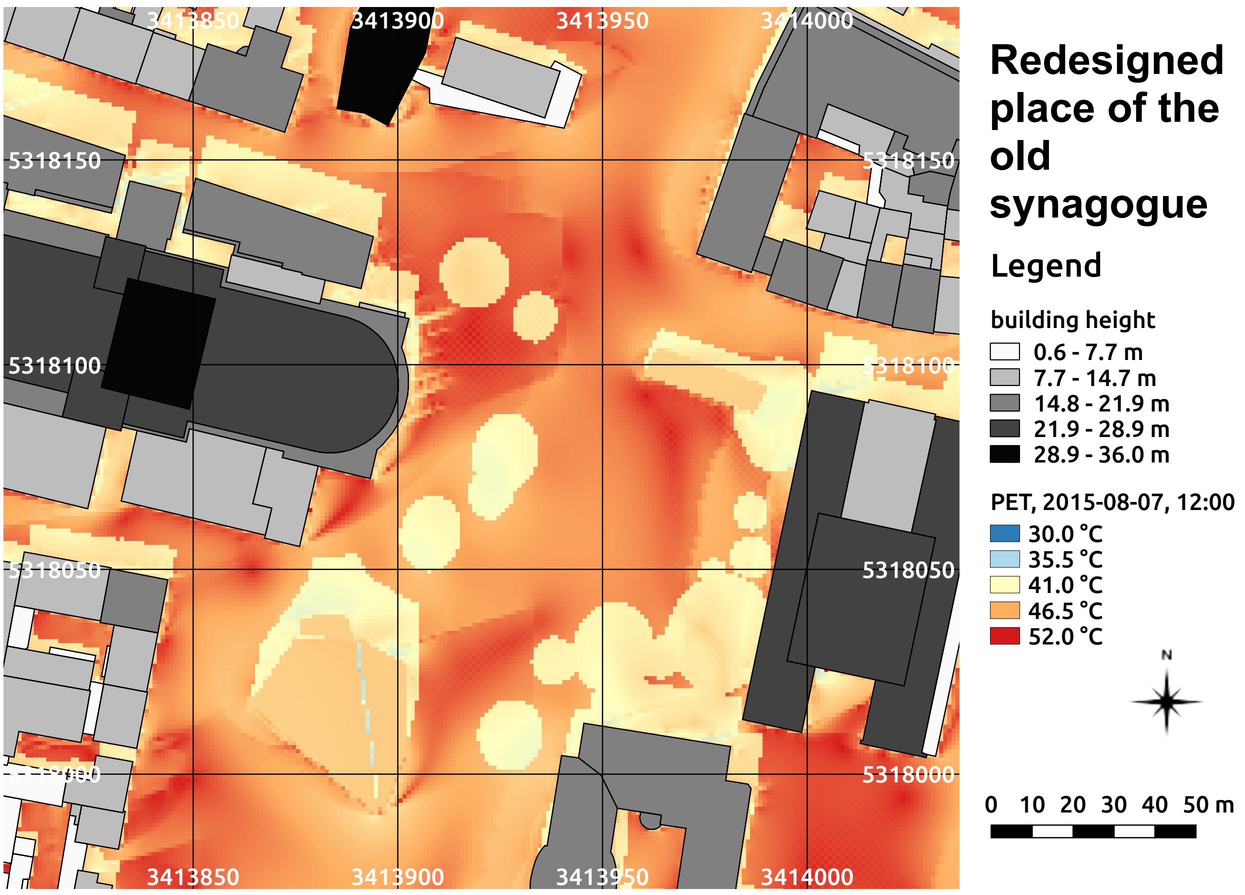


**fig. 7.20:** PET on the 07$^{th}$ of August 2015 at 12:00 at the redesigned place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree locations and positions are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

Two hours later, at 14:00, the redesigned place of the old synagogue is in most locations even hotter than at 12:00 (see fig. 7.21).

The area shaded by trees not influenced by the university library, that is different for the old and the redesigned place, show PET of 42.2°C to 44.2°C under the trees in the South-East of the place. This is little higher than PET in the same locations on the old place (compare to fig. 7.13), that is 42.0°C to 43.9°C. The difference can be explained with the higher density of trees present in the South-East of the old place. PET in the central part of the place, that is exposed to direct sunlight, but is not close enough to the university library to show the effect described above, ranges in between 47.8°C to 52.0°C on the redesigned place perfectly agreeing to PET on the old place. The higher wind speed on the redesigned place causes PET of around 48.0°C in between the library

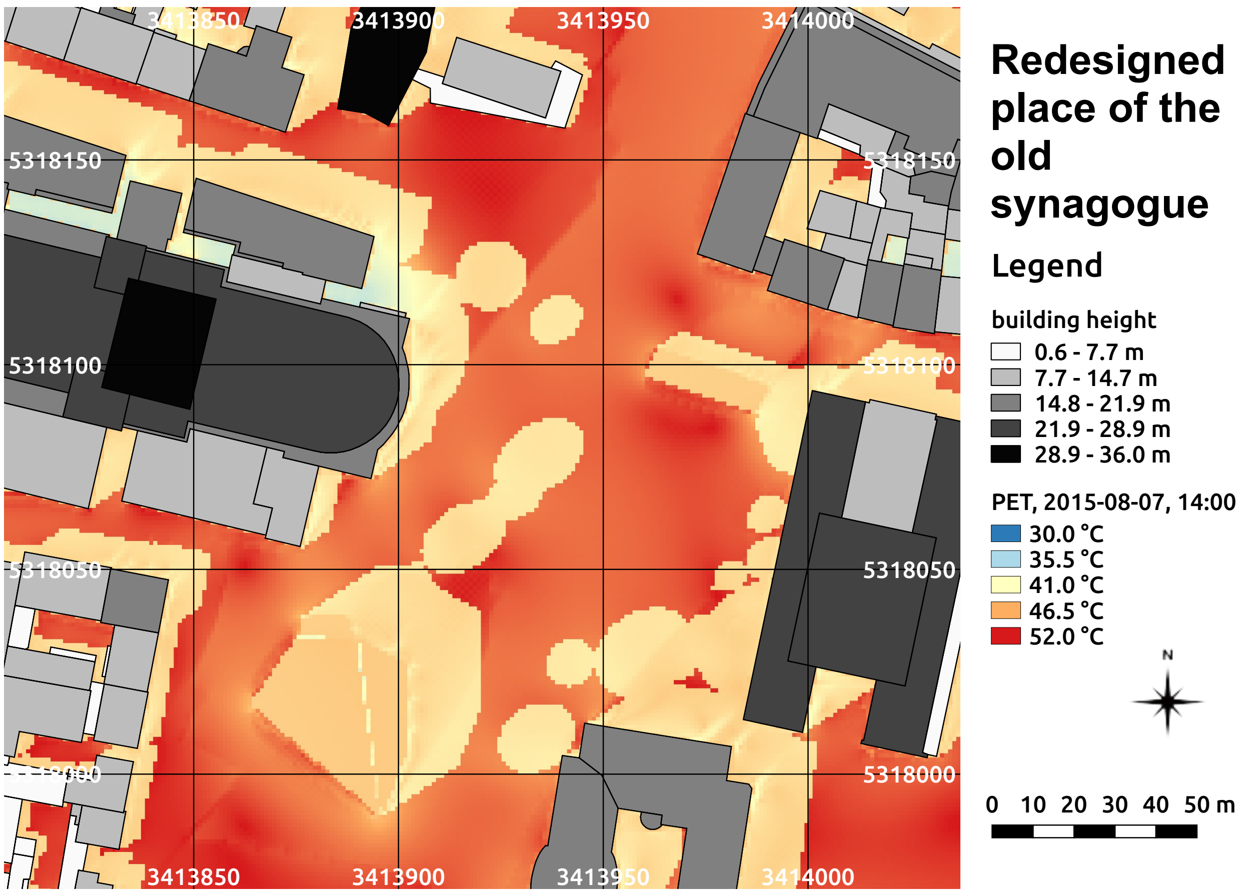


**fig. 7.21:** PET on the 07$^{th}$ of August 2015 at 14:00 at the redesigned place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree locations and positions are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

and the theatre. The same location is much hotter on the old place with PET of around 49.5°C. The shaded area available on the place of the old synagogue stays approximately the same size in total. Again, the redesigned place profits a lot from the rebuild university library providing a way larger shaded area than the old one. Without this effect, the shaded area would be smaller after the redesign.

Another two hours later, the average thermal conditions on the redesigned place of the old synagogue in means of PET are significantly less extreme than at 14:00 (compare fig. 7.22 to fig. 7.21). In some of the shaded areas PET is decreased below 41.0°C, causing only strong heat stress, instead of extreme heat stress according to Matzarakis and Mayer (1996, tab. 4.6). This, however, disagrees to the results provided by the assessment of PT (compare to fig. 7.18), indicating "warm"

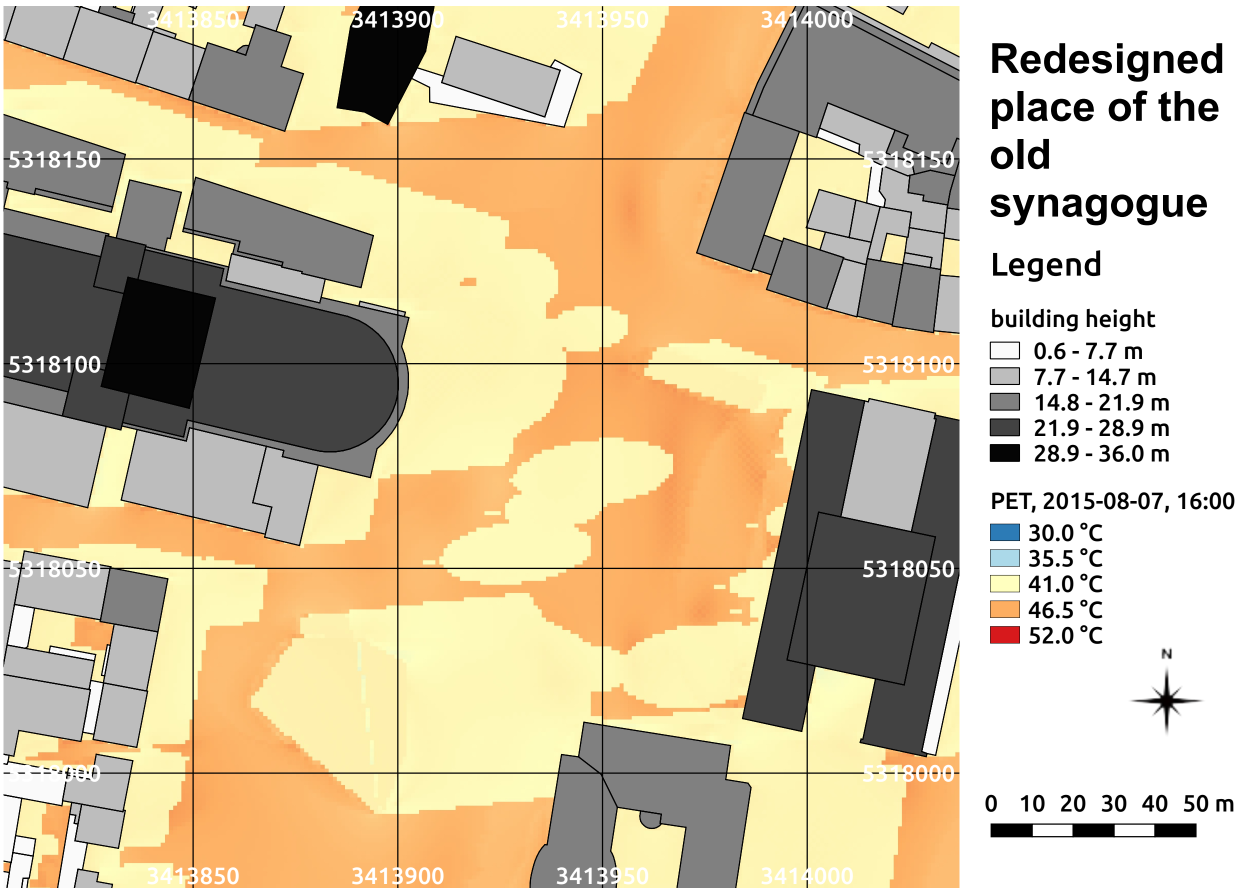


**fig. 7.22:** PET on the 07th of August 2015 at 16:00 at the redesigned place of the old synagogue in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) as well as tree locations and positions are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

to "hot" conditions, what is one thermal perception class lower.
Due to the lower incident wind speed, the influence of the new university library is weaker than at 14:00. This causes the absolute numbers of PET to agree well comparing the old place of the old synagogue (fig. 7.14) to the redesigned one (fig. 7.22). Within the shaded locations, PET is ranging from 40.5°C to 41.8°C. Areas exposed to direct sunlight show PET of 45.2°C to 47.0°C.
The area shaded by obstacles is slightly larger on the old place of the old synagogue. This is due to the new trees West of the KG II are too close to the building to improve thermal condition on the place a lot at 16:00.

## 7.2 Institutes Quarter

The second test domain applied to test the new functionality of the SkyHelios model is the Institutes Quarter in Freiburg (see section 6.8.2 for details). The second test domain was selected for its size. Calculating with a horizontal resolution of 1 m on 1 m, the Institutes Quarter test domain covers 544 on 715 grid points. Calculations for a huge domain like the Institutes Quarter are only possible after the modifications to the SkyHelios model improving the internal data processing. This new functionality is tested in this section.

### 7.2.1 Sky View Factor

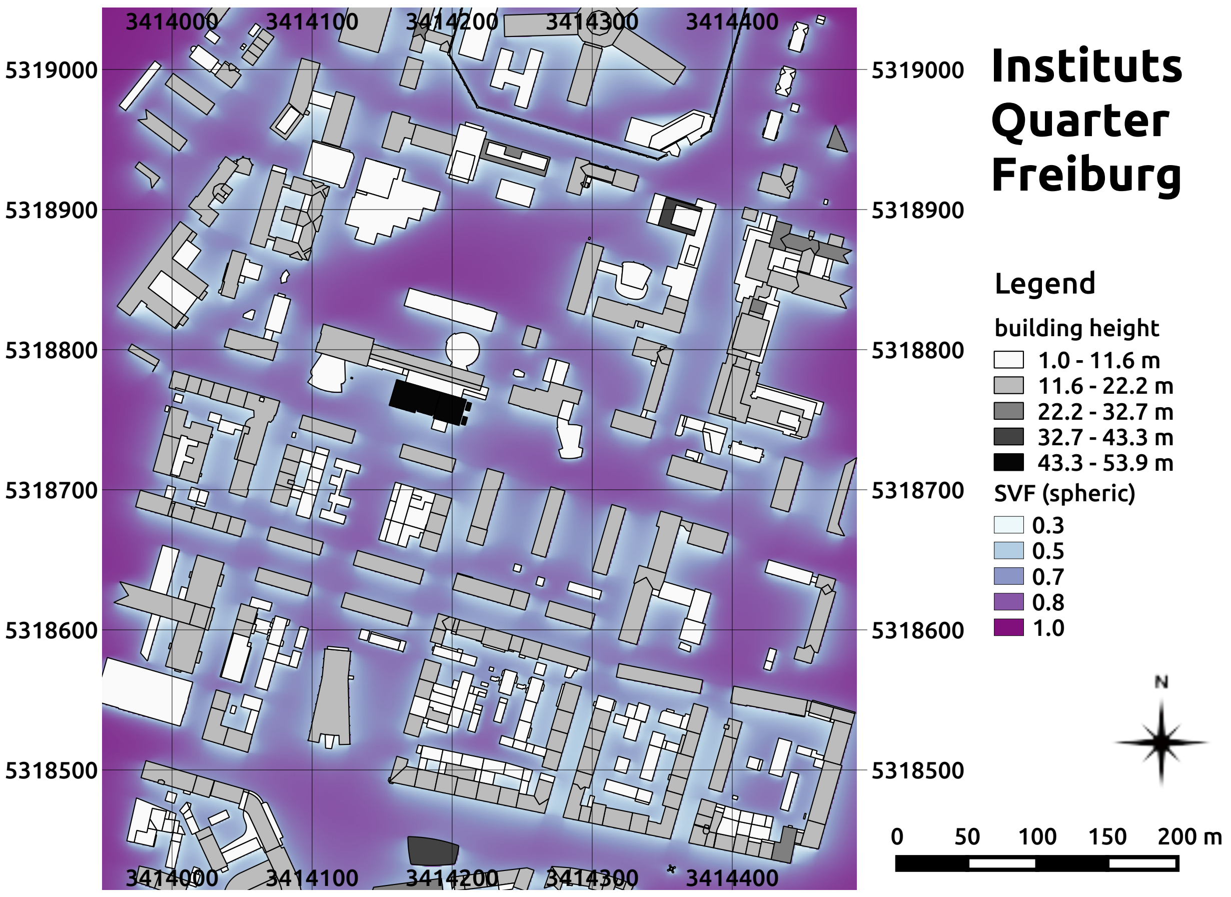


**fig. 7.23:** Spatial distribution of the spheric SVF in the Institutes Quarter in Freiburg, south-west Germany calculated by the SkyHelios model. The building heights above ground (m) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

The spheric Sky View Factor (SVF) is calculated for the whole Insitutes Quarter in Freiburg in 1 m on 1 m resolution. Results show, that the SkyHelios model is able to run large areas of interest flawlessly. This is representative for most variables calculated by the SkyHelios main model, e.g. SVF (planar and spheric), shading, sunshine durations and the static variables $T_a$ and Rh / VP.
The spatial distribution of spheric SVF within the Institutes Quarter (compare to fig. 7.23) shows lower values close to obstacles and high values for open spaces, e.g. the open space North of the Chemistry Highrise (the black building in the central area). The distribution suffers from the lack of spatial vegetation data input, that is unavailable for this area.

### 7.2.2 Wind Speed

One important parameter, that is calculated by an external module is wind speed. This of course also has to be tested for a huge input area like the Institutes Quarter. The wind model is the part of SkyHelios that is expected to be the first one to get in trouble facing a large input area as it requires three dimensional calculations and, thus, lots of memory space. The distribution of wind speed for the 07$^{th}$ of August 2015 at 08:00 in 1.5 m height at the Institutes Quarter (see fig. 7.24) shows, that the SkyHelios model is able to calculate wind speed for a large area of interest. The wind model was running this particular setting in 6.4 min on a below-average machine.
On the 07$^{th}$ of August 2015 at 08:00 in 1.5 m height, average wind speed is ranging from 0.3 m/s in areas experiencing wind shelter by buildings to 1.2 m/s within eddies between buildings. The situation at 12:00 is presented here exemplary for showing the highest wind speed. The spatial distribution for other three lines of the meteorological input file (tab. 6.4) can be found in the appendix (see figs. 10.1 to 10.3).

### 7.2.3 Perceived Temperature

The parameters that need testing the most is the thermal indices. This is due to being the main function of SkyHelios, but also because they include most of the other parameters considered by the SkyHelios model. The first thermal index to be tested is, again, the Perceived Temperature.
At 08:00 on the 07$^{th}$ of August, the Institutes Quarter is already fairly heated up (see fig. 7.25) in spite the sun elevation is still low and the shaded areas are quite large. Due to the relatively low wind speed at 08:00 (compare to fig. 10.1) PT already ranges in between 27.7°C and 29.7°C indicating "warm" conditions according to tab. 4.3.
The unshaded areas offer a wider range of PT, spanning from 30.7°C to 34.3°C and, thus, covering

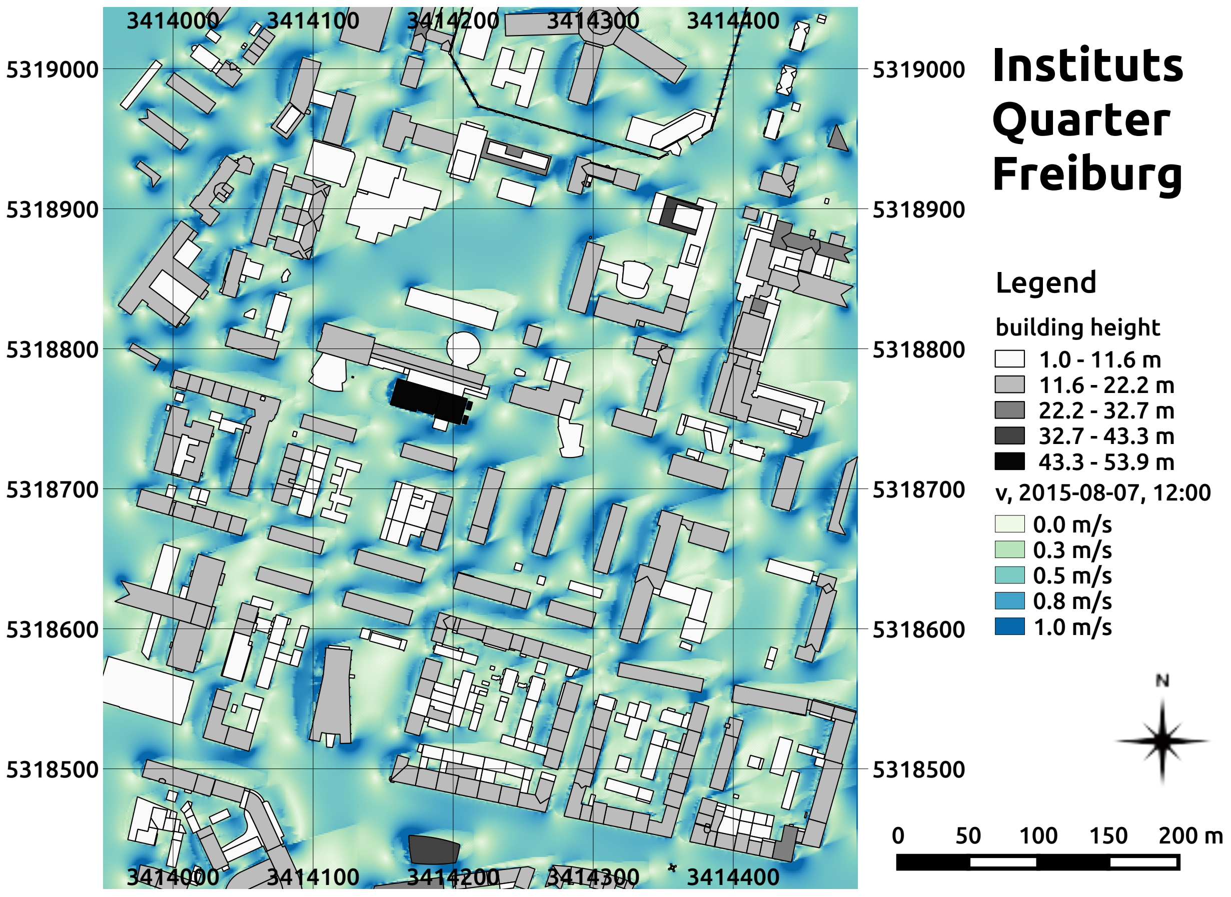


**fig. 7.24:** Spatial distribution of wind speed for the 07$^{th}$ of August 2015 at 12:00 in 1.5 m height at the Institutes Quarter in Freiburg, south-west Germany calculated by the SkyHelios model. Incident wind in 10 m height is 4.5 m/s from 253°. Building heights above ground (m) as well as tree properties and positions (compare to fig. 6.15) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

the classes "warm" and "hot".

At midday, thermal conditions in the whole Institutes Quarter have worsened. The shaded areas now show PT of 32.9°C to 34.2°C. This complies to the class "hot" in tab. 4.3 indicating great heat stress even within the shaded areas.
Outside the shaded areas PT ranges in between 34.3°C and 36.9°C. This is hotter than in the shaded areas, but still within the class "hot". The class "very hot", starting from PT of 38.0°C is not reached at any location within the Institutes Quarter at 12:00.

Two hours later, at 14:00, thermal stress increased slightly even compared to the conditions at 12:00

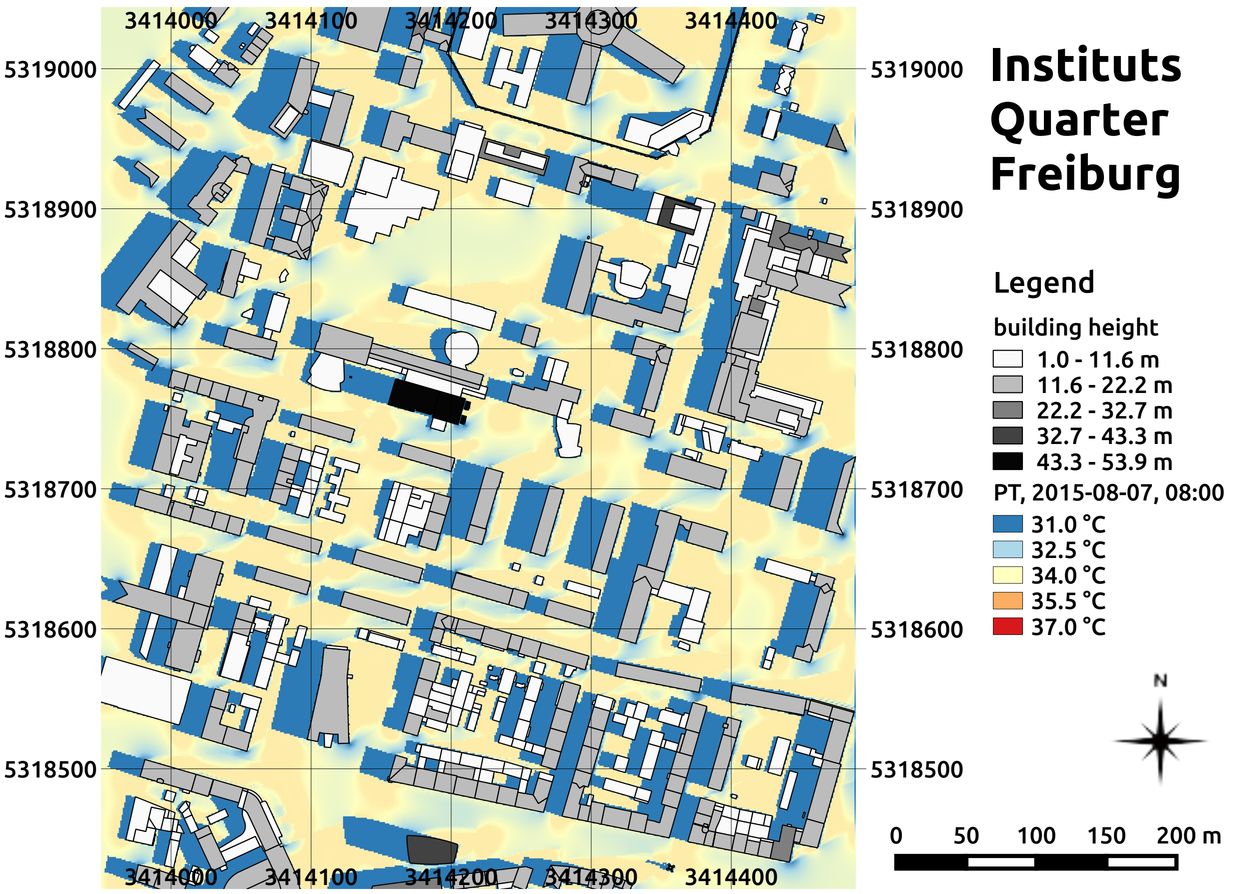


**fig. 7.25:** PT on the 07$^{th}$ of August 2015 at 08:00 at the Institutes Quarter in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

(compare fig. 7.26 to fig. 7.27). The unshaded areas now show very constant thermal conditions with PT in range of 33.6°C to 33.7°C. Areas with direct solar radiation are showing PT of 35.6°C to 36.8°C. While both is quite hot, it is both still part of the class "hot". The class "very hot", starting from PT of 38.0°C is not reached anywhere. Extreme heat stress is therefore not present on the 07$^{th}$ of August within the Institutes Quarter at 14:00 according to PT.

The spatial distribution of PT for 16:00, showing the situation another two hour later, indicates more comfortable conditions than at 14:00 (compare fig. 7.27 to fig. 7.28). Within the shaded areas, PT is only 31.5°C to 32.3°C. This means, that part of the shaded areas are cooler than 32.0°C, so they match the class "warm". The rest of the shaded areas still are causing great heat stress being part of the class "hot".
Within areas exposed to direct solar radiation PT is between 33.8°C and 34.0°C. The variation in

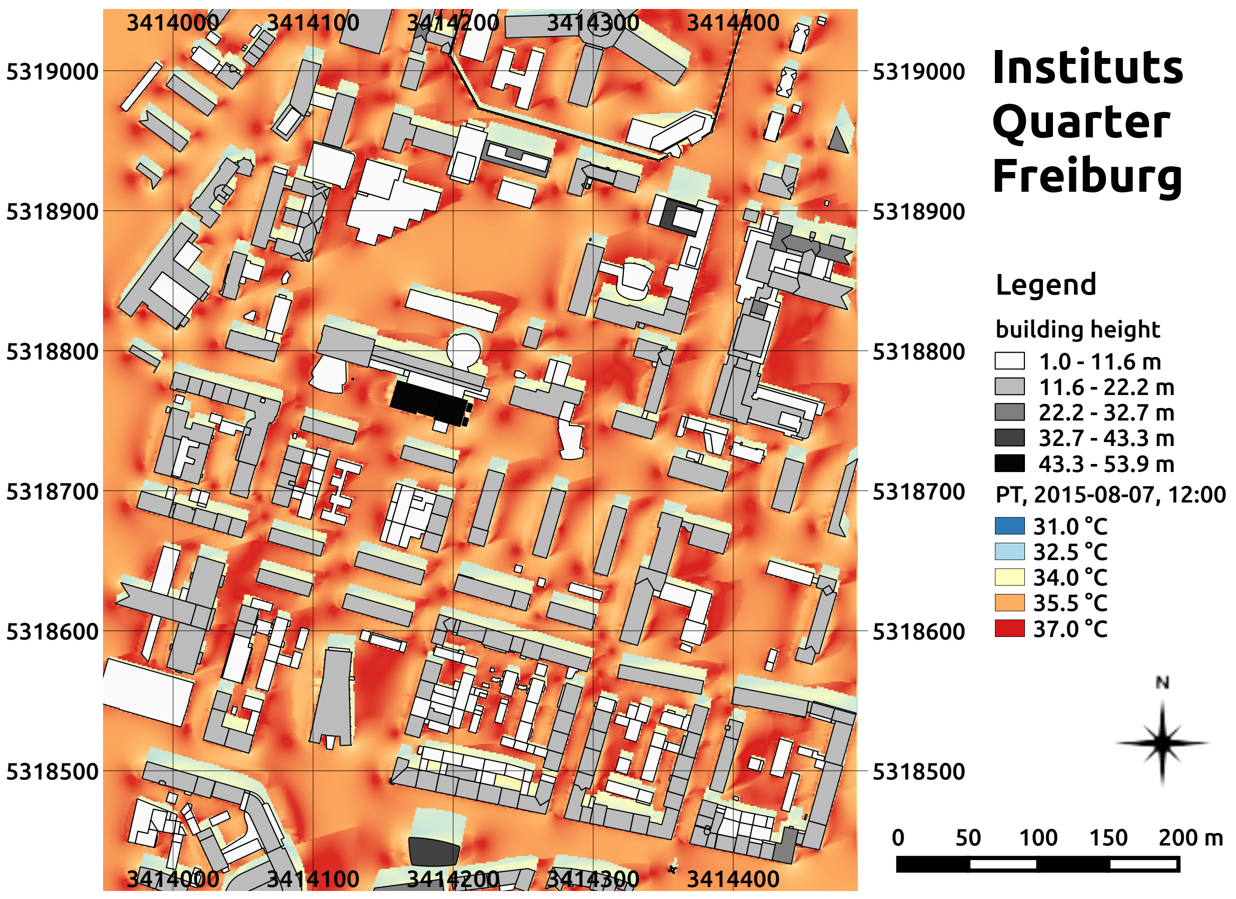


**fig. 7.26:** PT on the 07th of August 2015 at 12:00 at the Institutes Quarter in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

PT is quite narrow in the unshaded areas only spanning 0.2°C. All the values are fitting the class "hot".

### 7.2.4 Universal Thermal Climate Index

Another index that can be determined by the SkyHelios model is the relatively new thermal index "Universal Thermal Climate Index" (see section 4.3.6.2). The newly included functionality is of course also tested for the large test domain, the Institutes Quarter (compare to fig. 7.29). For the meteorological input conditions, the same meteorological data input file is selected to be able to compare the results to those calculated by other indices (e.g. with PT, fig. 7.26). The decision for the 07th of August 2015 at 12:00 was due to wind speed being the highest at 12:00 among the four meteorological datasets. As the range of valid input conditions for the UTCI calculations is rather

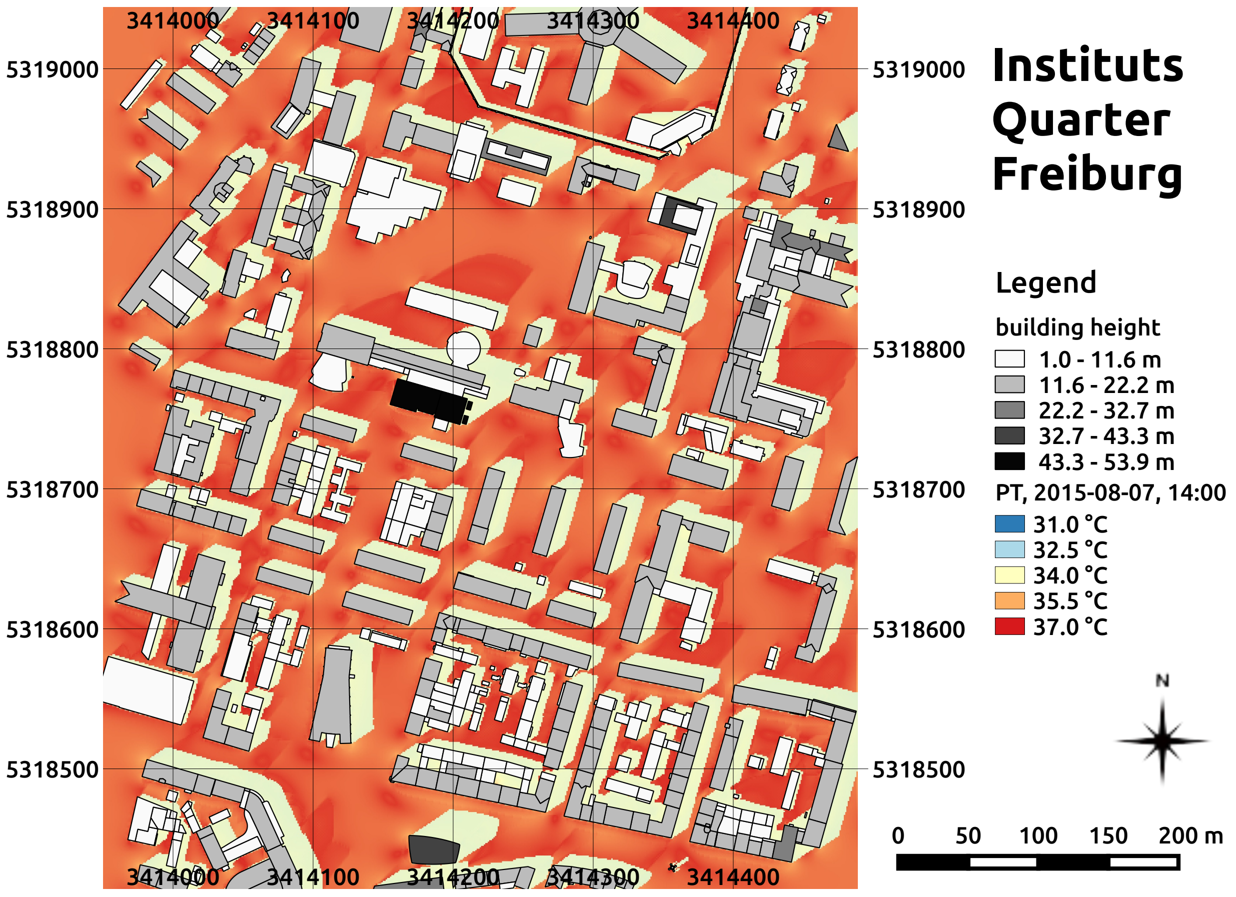


**fig. 7.27:** PT on the 07th of August 2015 at 14:00 at the Institutes Quarter in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

narrow (see section 4.3.6.2), this set of input data was expected to generate the smallest amount of fail values.

Looking at fig. 7.29 it becomes obvious, that the area is dominated by fail values (depicted in pink) generated due to wind speed (extrapolated to 10 m above ground level using the vertical profile provided by Havenith et al. (2012)) below 0.5 m/s. In areas with sufficient wind speed UTCI ranges in between 36.5°C and 39.0°C in shaded locations, and 38.2°C to 39.5°C at locations exposed to direct solar irradiation.

Applying the assessment scale after Błażejczyk et al. (2013, see tab. 4.4), this means strong heat stress in most of the shaded locations. Only some are exceeding 38.0°C and, thus, are causing very strong heat stress. The locations without shading are counting to the thermal stress category "very strong heat stress" without any exception. It, thus, better agrees to the results of PT (see above), than those of PET indicating extreme heat stress (see below).

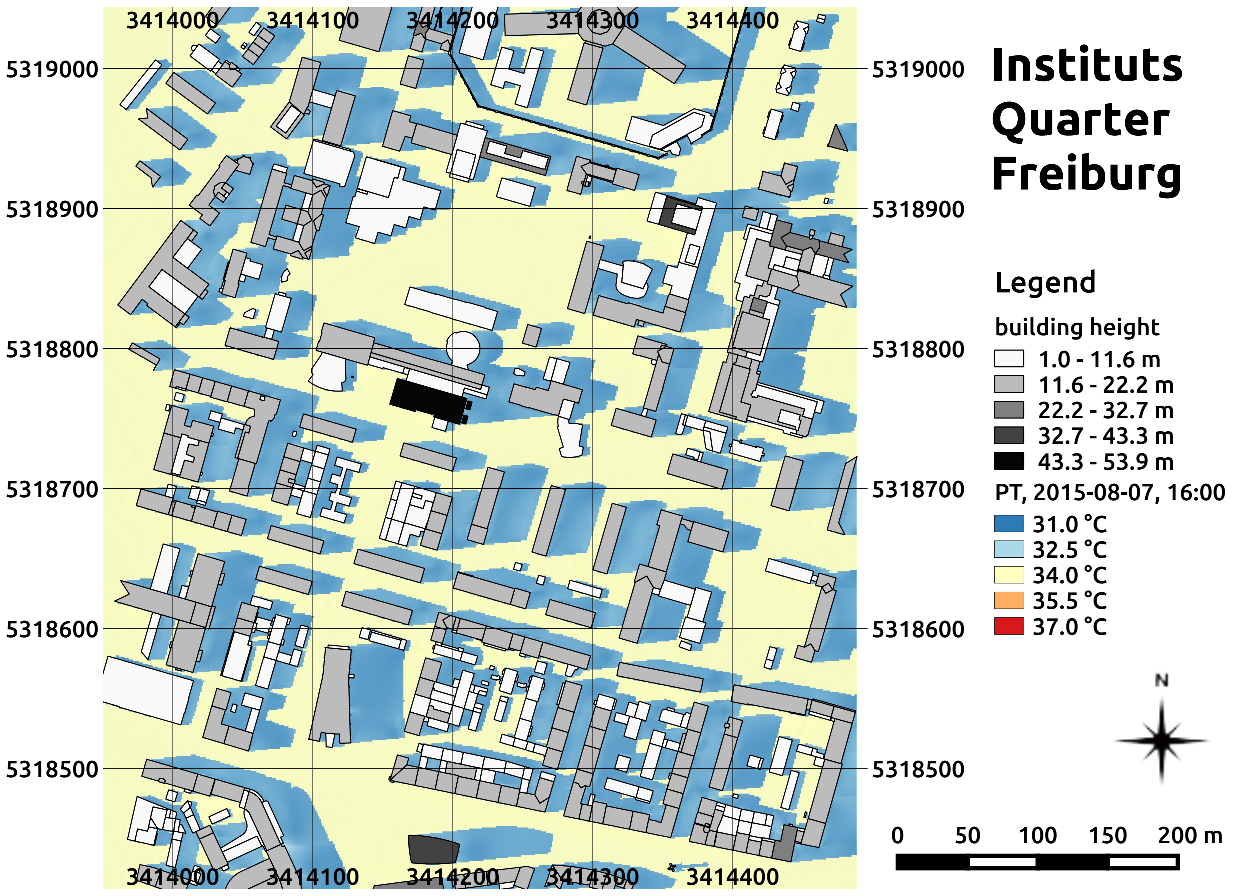


**fig. 7.28:** PT on the 07th of August 2015 at 16:00 at the Institutes Quarter in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

Due to the high amount of fail values, UTCI distributions are not presented here for the other three meteorological data sets. However, they are included in the appendix for comparison (see figs. 10.4 to 10.6).

### 7.2.5 Physiologically Equivalent Temperature

One of the most commonly used thermal indices in human thermal bioclimatology is the Physiologically Equivalent Temperature. PET therefore is of course also tested for the very large input area "Institutes Quarter" and with the same meteorological data input file to allow for inter comparison between PET and the indices PT and UTCI (see above).
As for PT, the thermal conditions are chronologically assessed based on PET, starting with the situation on the 07th of August 2015 at 08:00 at the Institutes Quarter in Freiburg. Looking at the

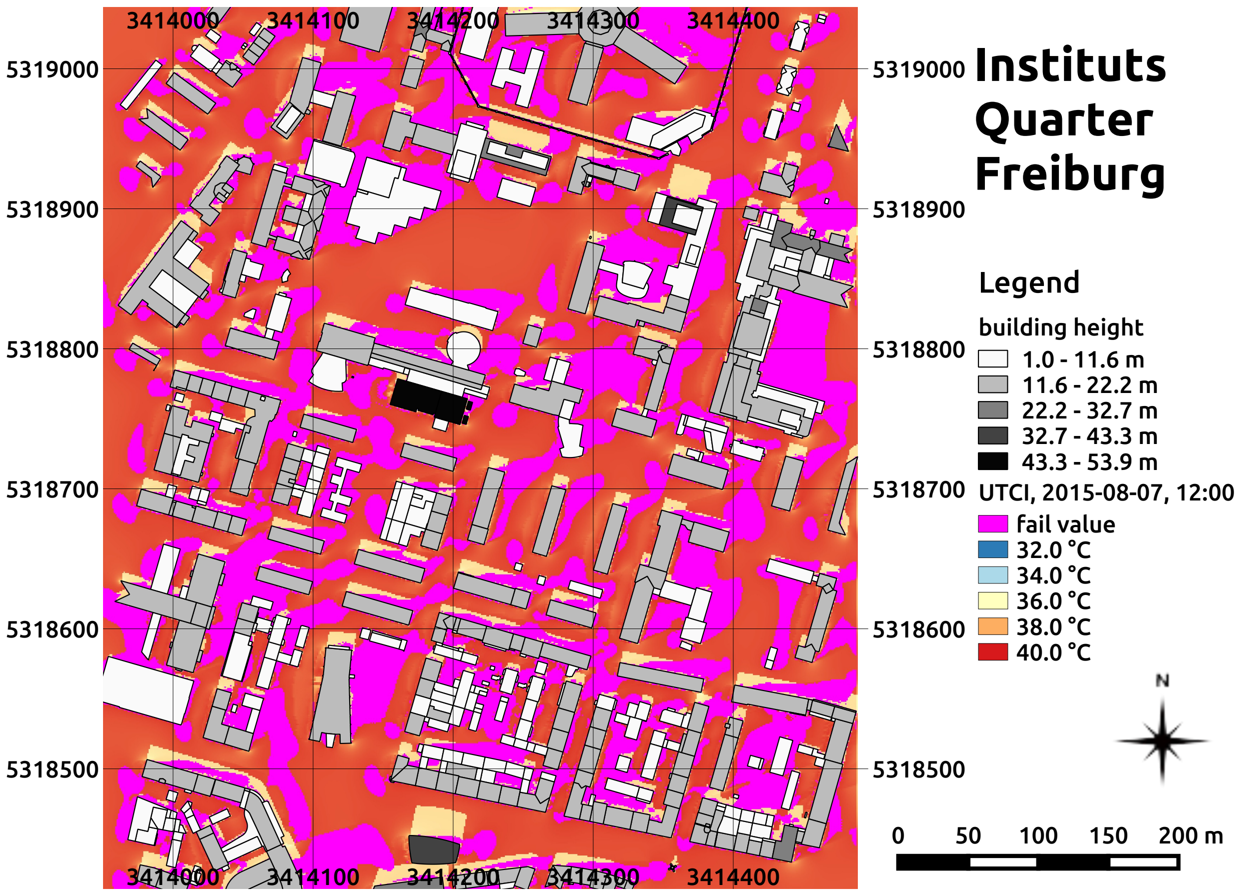


**fig. 7.29:** UTCI on the $07^{th}$ of August 2015 at 12:00 at the Institutes Quarter in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

distribution of PET at 08:00 (fig. 7.30) it is clear to see, that PET calculates the highest values among the three indices (compare to figs. 7.25 and 10.4).
The quite large shaded areas show PET in between 30.3°C and 33.5°C. The shaded areas therefore are causing moderate to strong heat stress according to tab. 4.6 being part of the classes "warm" or "hot".
The areas exposed to solar radiation are way warmer. PET ranges from 37.3°C to 41.8°C there. While most of the unshaded locations are part of the class "hot", causing strong heat stress, some locations with low wind speed are already exceeding PET of 41.0°C causing extreme heat stress according to tab. 4.6. This means, PET is indicating almost one class more extreme conditions than PT. A comparison to the values determined for UTCI fails, as there are almost only fail values due to low wind speed at 08:00 (see fig. 10.4).

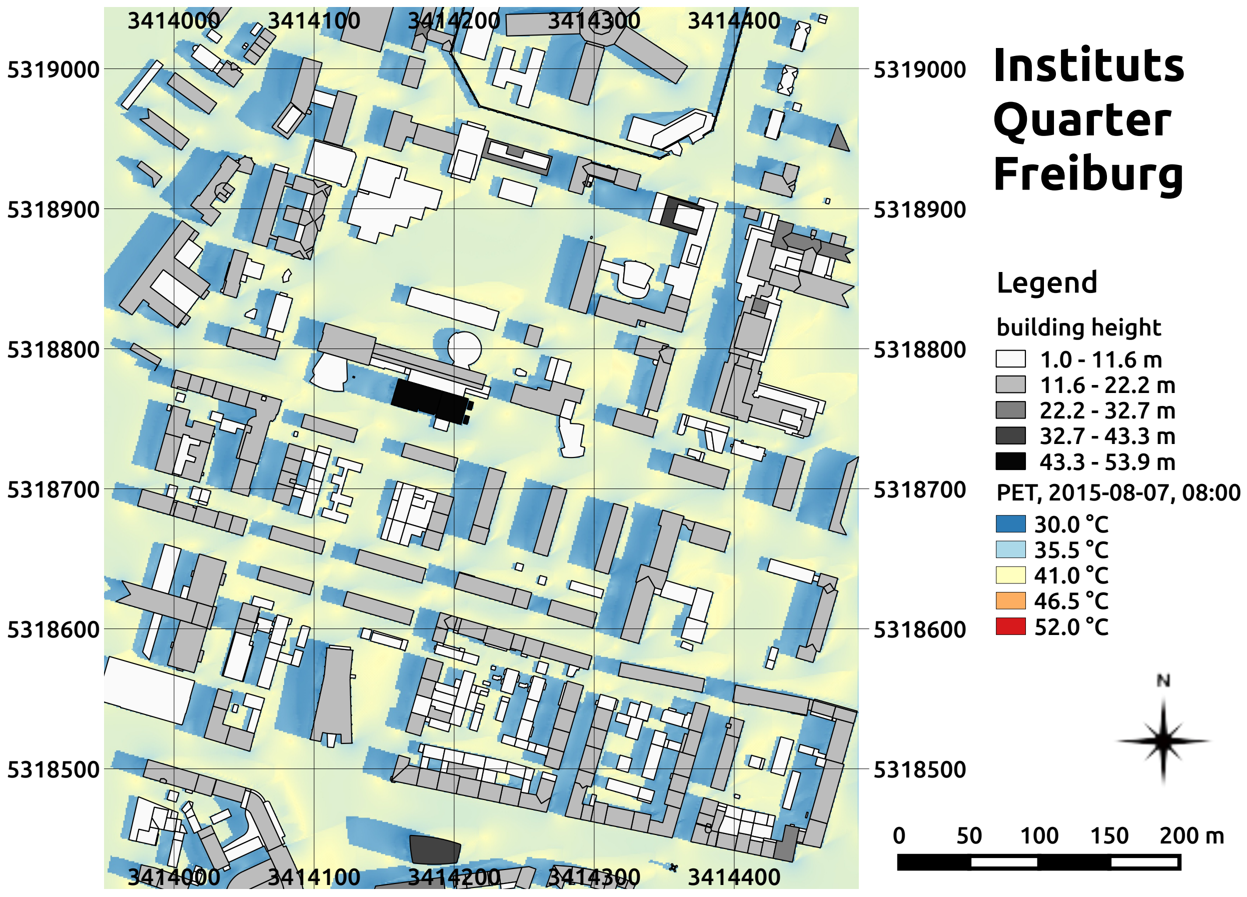


**fig. 7.30:** PET on the 07th of August 2015 at 08:00 at the Institutes Quarter in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

Until midday, the shaded areas became very small (compare to fig. 7.31). Due to the high solar elevation, the areas exposed to direct solar radiation are largest at 12:00. Within the small shaded areas remaining, PET reaches 39.9°C to 43.1°C. Even part of the shaded locations are providing conditions that are assessed to cause extreme heat stress by PET being part of the class "very hot". The rest of the shaded locations is part of the class "hot" causing strong heat stress.
Within the huge unshaded areas PET shows values of 42.3°C to up to 51.5°C. All the unshaded areas are therefore part of the class "very hot" and are causing extreme heat stress to humans according to tab. 4.6. Both is approximately one class hotter than predicted by PT. The few valid values for UTCI (fig. 7.29) also better agree to those determined by PT, than those shown by PET.

Two hours later, at 14:00, also PET calculates the highest values among the four meteorological input data sets (see fig. 7.32). While the shaded areas became larger since the 12:00 situation, they

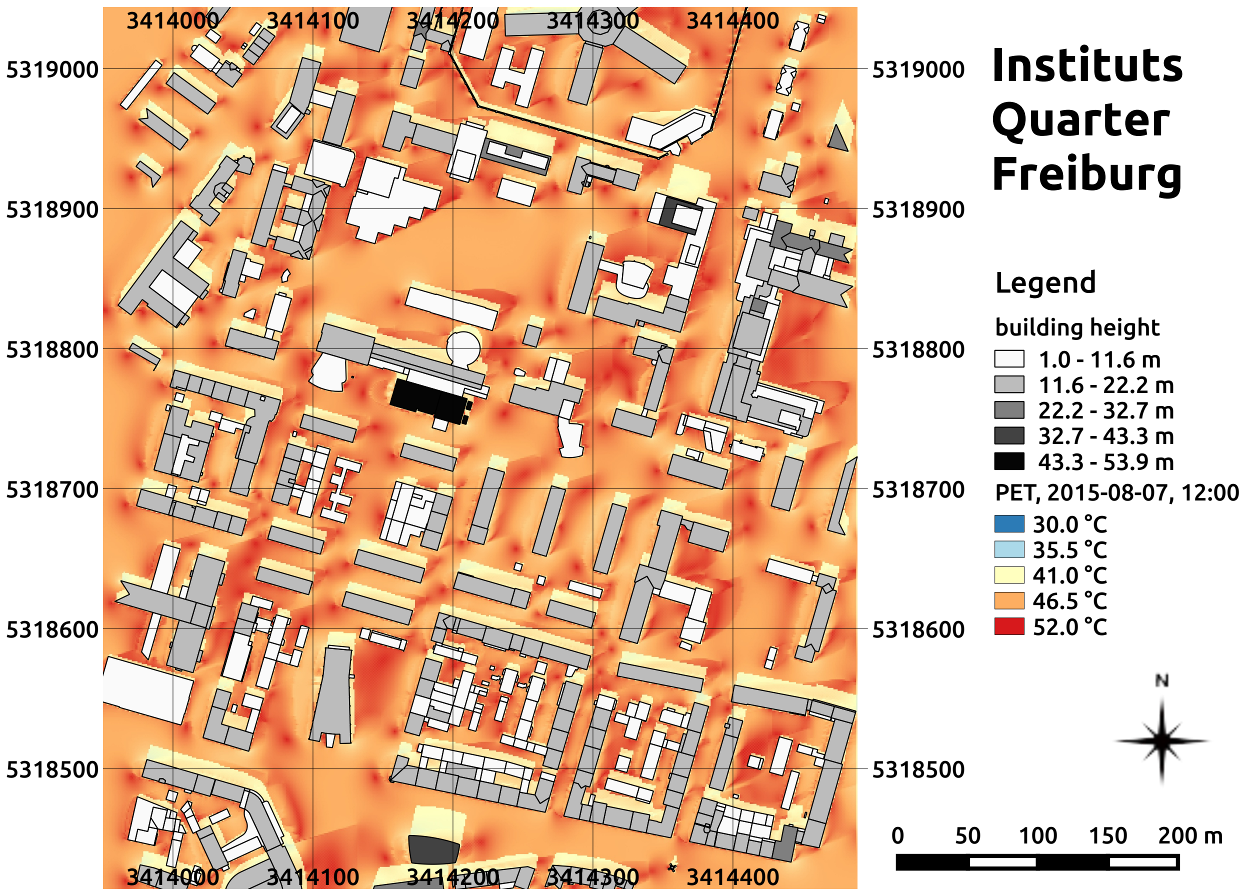


**fig. 7.31:** PET on the 07$^{th}$ of August 2015 at 12:00 at the Institutes Quarter in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

can not offer comfortable conditions at 14:00. Within the shaded areas PET ranges from 40.2°C to 43.9°C making the shaded areas mostly part of the class "very hot" causing extreme heat stress. Only some shaded locations show PET of lower than 41.0°C only causing strong heat stress. The unshaded areas are offering the most uncomfortable conditions among all the indices and all the meteorological input data sets. PET indicates with values of 46.3°C to 52.0°C extreme heat stress at all locations exposed to direct sunlight. Both, the shaded, as well as the unshaded areas are approximately one class hotter than predicted by PT (compare to fig. 7.27) and (where available) UTCI (compare to fig. 10.5).

Two hours later, at 16:00 in the afternoon, condition within the Institutes Quarter are much more comfortable again. Agreeing to all the indices, PET calculates significantly lower values at 16:00 (fig. 7.33) than at 14:00 (fig. 7.32).

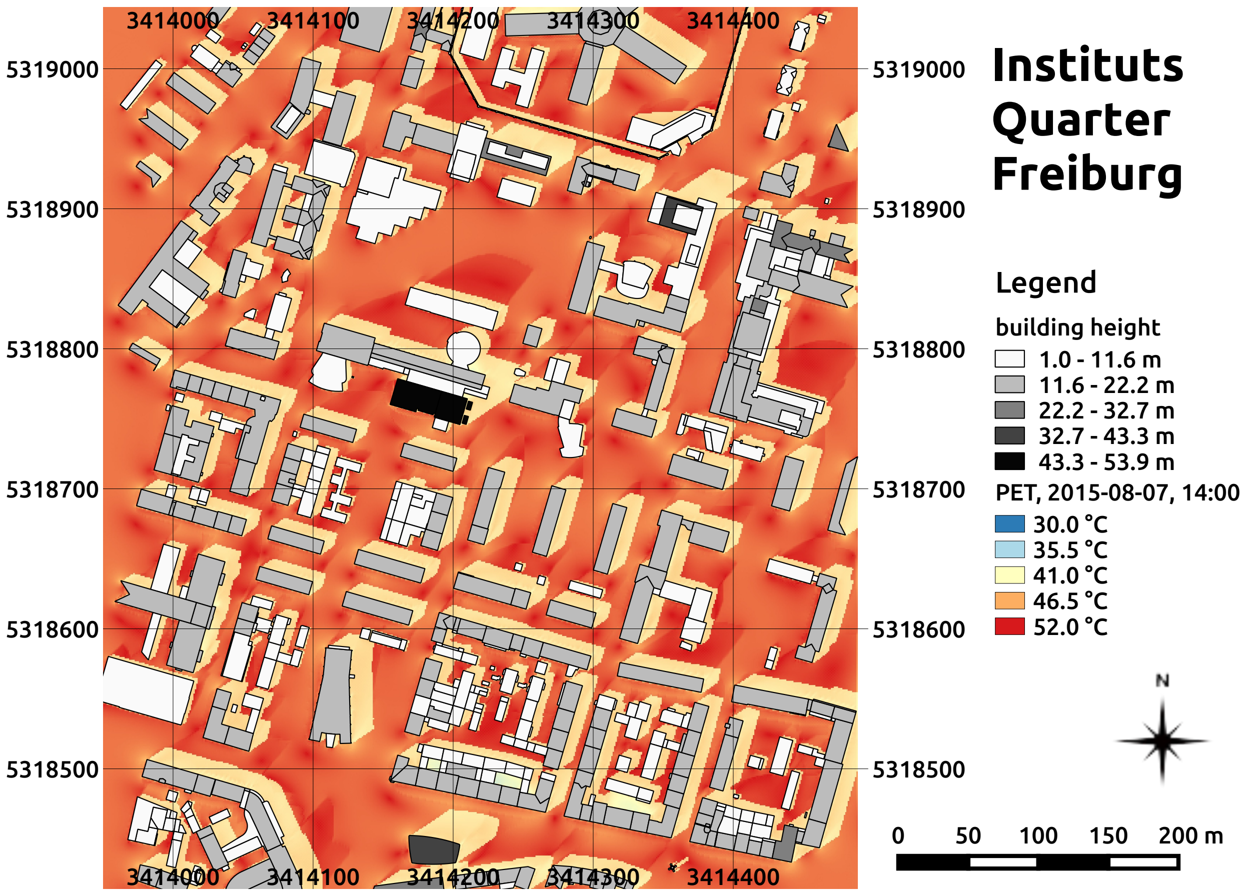


**fig. 7.32:** PET on the 07th of August 2015 at 14:00 at the Institutes Quarter in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

However, the large shaded areas still show PET of 40.9°C to 41.7°C, mostly causing extreme heat stress as part of the class "very hot". Only the few locations, where PET drops below 41.0°C are causing strong heat stress.
Within the unshaded locations, PET is ranging in between 45.3°C and 47.2°C. All the unshaded locations are therefore part of the class "very hot" indicating extreme heat stress. Both, the assessment for the shaded, as well as for the unshaded conditions therefore disagrees to the assessment by PT (compare to fig. 7.28) and UTCI (compare to fig. 10.6). Both of the other thermal indices show values matching approximately one class lower than the one for PET.

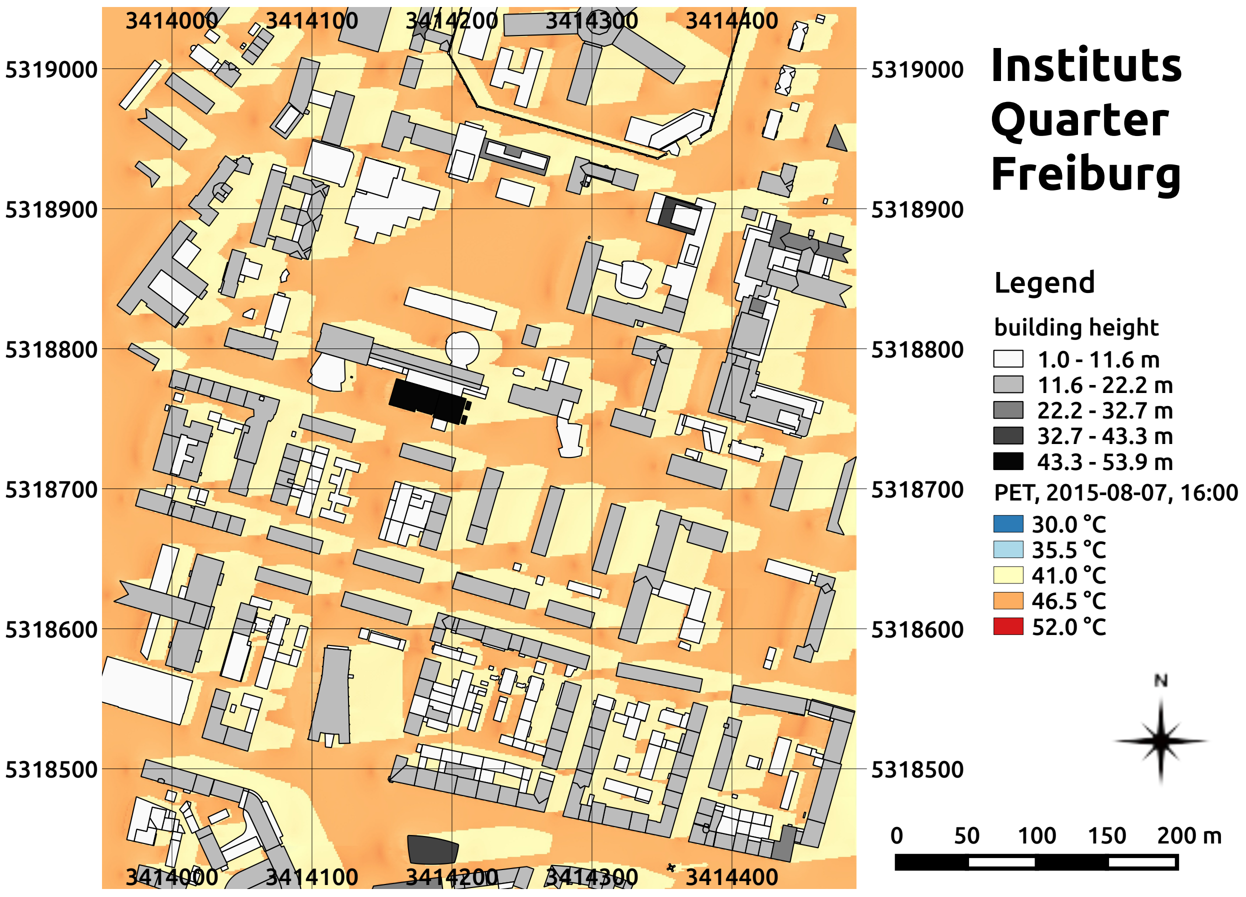


**fig. 7.33:** PET on the 07th of August 2015 at 16:00 at the Institutes Quarter in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

## 7.3 Comparison to measurements

Results calculated by the advanced SkyHelios model are compared to measurements by local stations on the place of the old synagogue as described in section 6.8.3. As $T_a$ and $VP$ are considered constant in space by the advances SkyHelios model, results for those two parameters are not compared here. They are expected to perfectly agree to those of the urban climate Station Freiburg providing the input data.

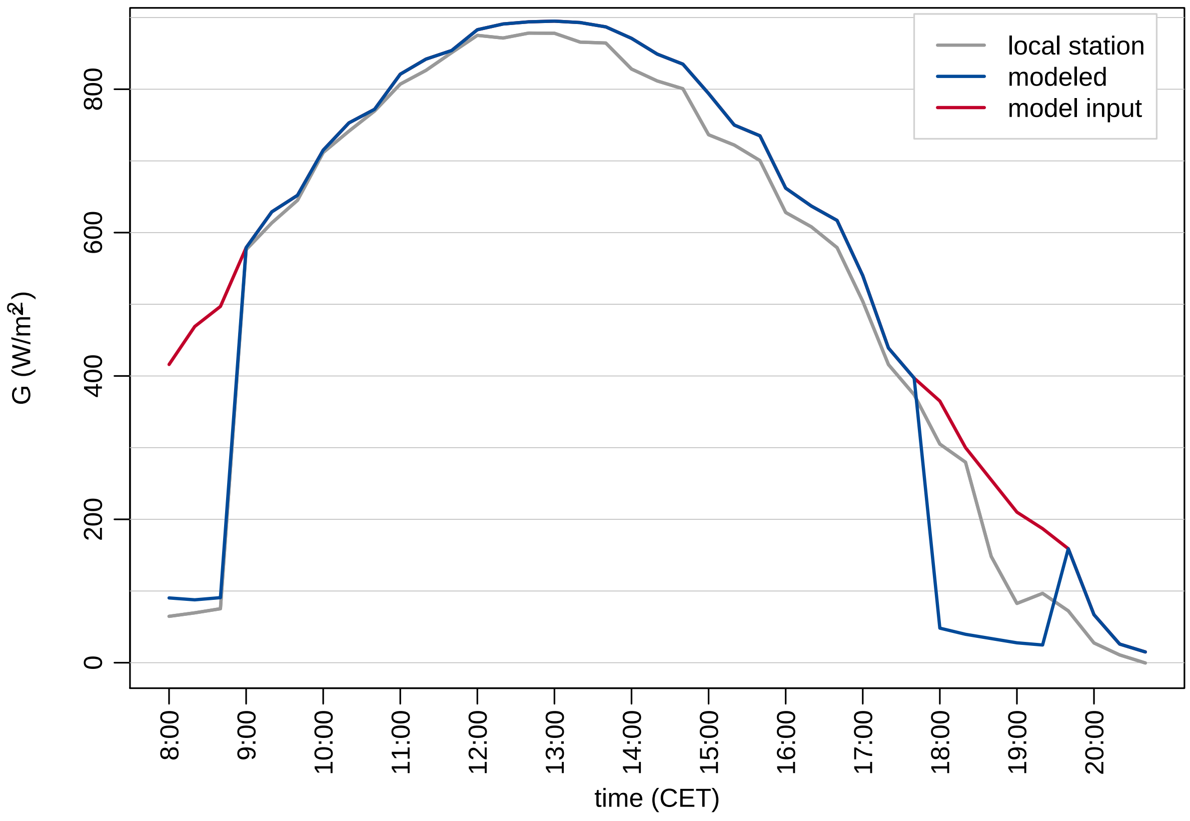


**fig. 7.34:** Comparison of global radiation (G) on the place of the old synagogue on the 01$^{st}$ of July 2008 as measured by a local biometeorological station (grey) and calculated by the SkyHelios model (blue). The model input, provided by the urban climate station Freiburg, is shown in red color.

### 7.3.1 Global radiation

Global radiation was only recorded by the local urban climate station ("fix" in fig. 6.15). Therefore only results for this station can be compared to the calculations.
Comparing the global radiation calculated by the advanced SkyHelios model (blue line in fig. 7.34) to the values measured by the local biometeorological station (grey line in fig. 7.34) a good agreement can be seen for most of the times compared. While there is a slight deviation of approximately 20 $\frac{W}{m^2}$ in the time between 08:00 and 09:00 that can be explained by the insufficient representation of a tree in the spatial input of the model, the values for the rest of the morning are in perfect agreement. The deviation in the afternoon of up to 30 $\frac{W}{m^2}$ are already present in the input date (red line in fig. 7.34) and therefore have to be considered deviation in the measurements between the rooftop station and the local biometeorological station on site. A stronger deviation can be seen in the evening from

approximately 18:00 to 19:30. The reason for this is, that the position is shaded in the model while the on-site station was exposed to direct solar radiation. The cause of this is either an imprecision in the digital representation of the theatre used by the model, or a position mismatch between the local biometeorological station and the point of investigation.

### 7.3.2 Wind speed

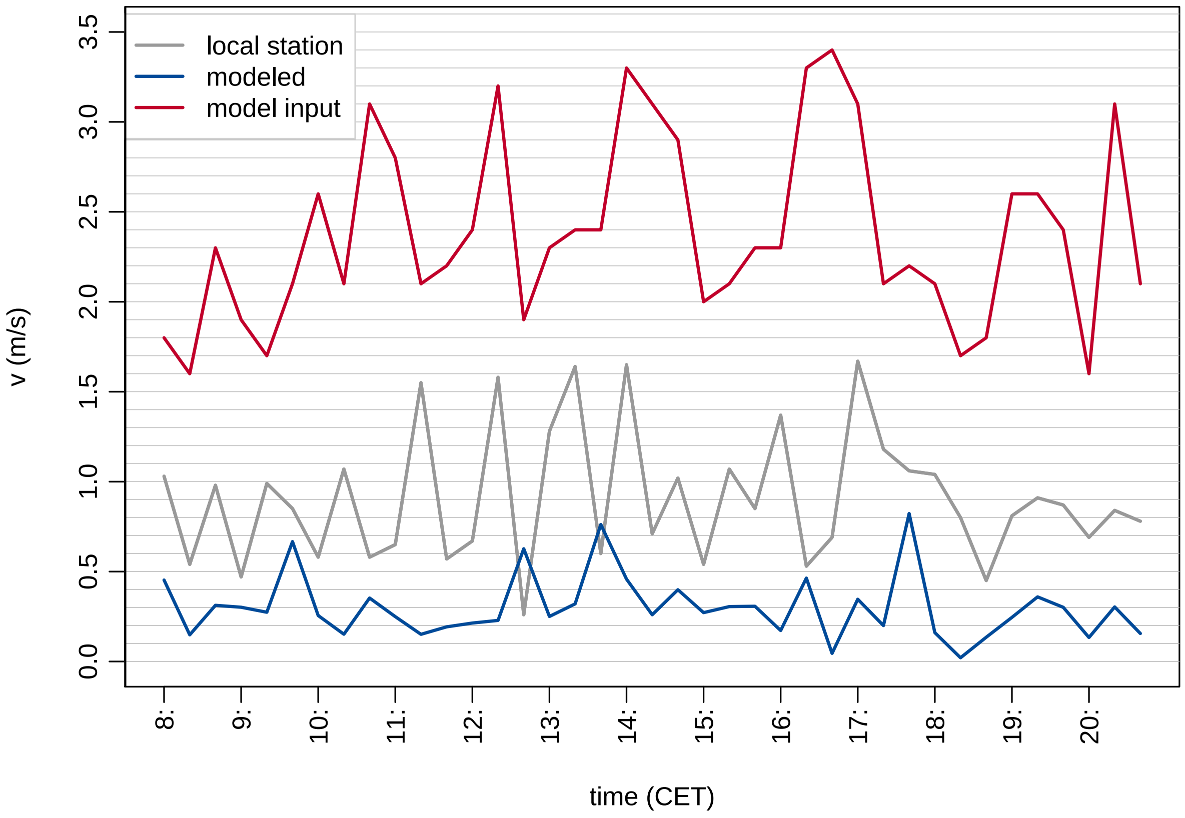


**fig. 7.35:** Comparison of wind speed ($v$) on the place of the old synagogue on the 01$^{st}$ of July 2008 as measured by a local biometeorological station (grey) and calculated by the SkyHelios model (blue). The model input provided by the urban climate station Freiburg is depicted in red.

Another important parameter compared is wind speed. Comparing the values recorded at the on-site station to the values modelled by the advanced SkyHelios model for the same position it can be seen, that the modelled values agree far better to the on-site measurements, than the model input wind speed. However, the calculated values appear to be lower than the measured ones (compare blue line (modelled) to grey lines (measured by the on-site station) in fig. 7.35). While the pattern is quite similar, the values are too small throughout the day. This is even worse for the comparisons to the mobile measuring points MP1 to MP3 (see figs. 10.7 to 10.9) that can be found in section 10.3.1. However, wind speed provided by the mobile station at all the three points appears to be quite high, disagreeing to the readings of both, the on-site station as well as the urban climate station Freiburg.

### 7.3.3 Wind direction

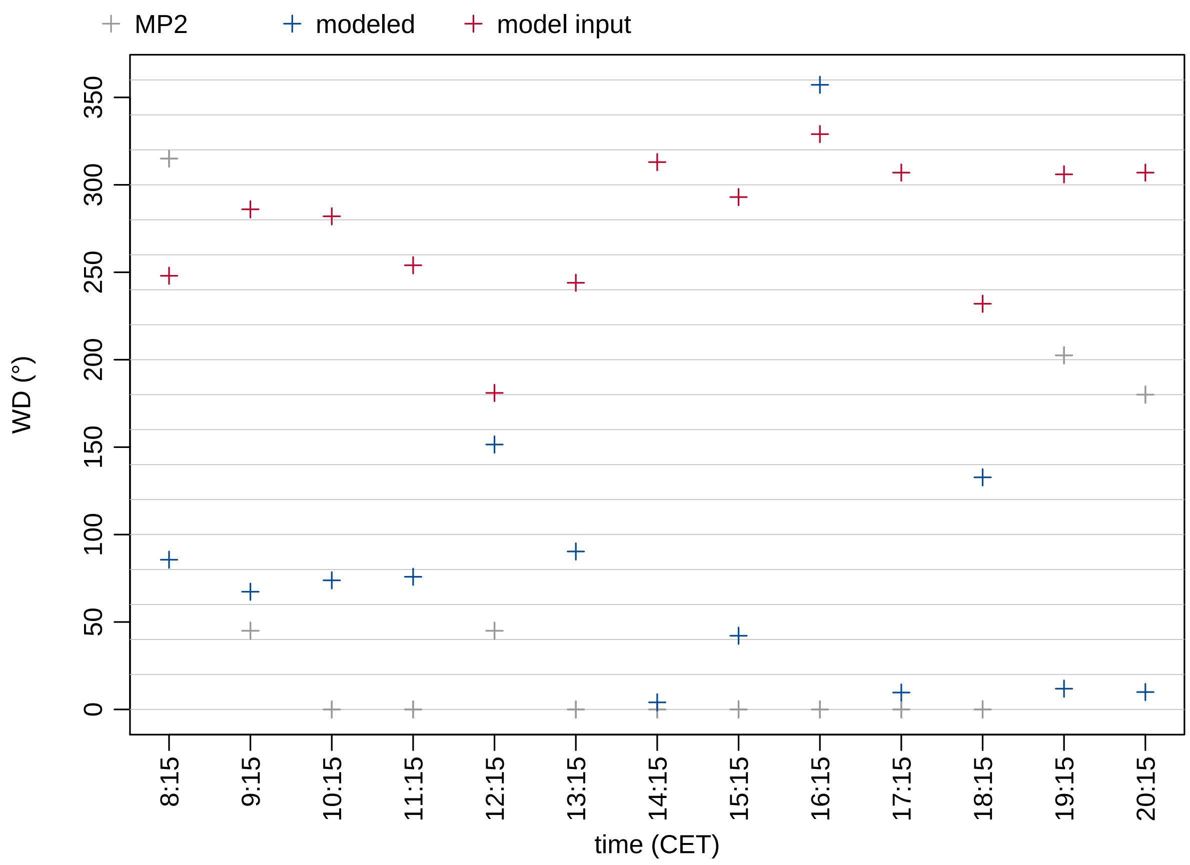


**fig. 7.36:** Comparison of wind direction ($WD$) on the place of the old synagogue on the 01$^{st}$ of July 2008 as measured at measuring point 2 (MP2) by a mobile biometeorological station (grey) and calculated by the SkyHelios model (blue). The input values for the advanced SkyHelios model are given by the red crosses for comparison.

Not only wind speed, but also wind direction is calculated by the advanced SkyHelios model. $WD$ therefore also is compared to the measurements. This is a little bit challenging, as wind speed was very low on the 01$^{st}$ of July 2008 and, thus, wind direction is very unstable. In contrast to $v$, the measurement point 2 (MP2) was therefore selected for the comparison, as it, in contrast to the other

data points, shows the most stable wind direction (see fig. 7.36).
Looking at fig. 7.36 one can see, that the modelled wind direction does not perfectly match the one recorded by the mobile urban-biometeorological station at MP2. However, the modelled values are almost always closer to the measured ones, than the incident wind direction stated by the red crosses in fig. 7.36.

### 7.3.4 Mean radiant temperature

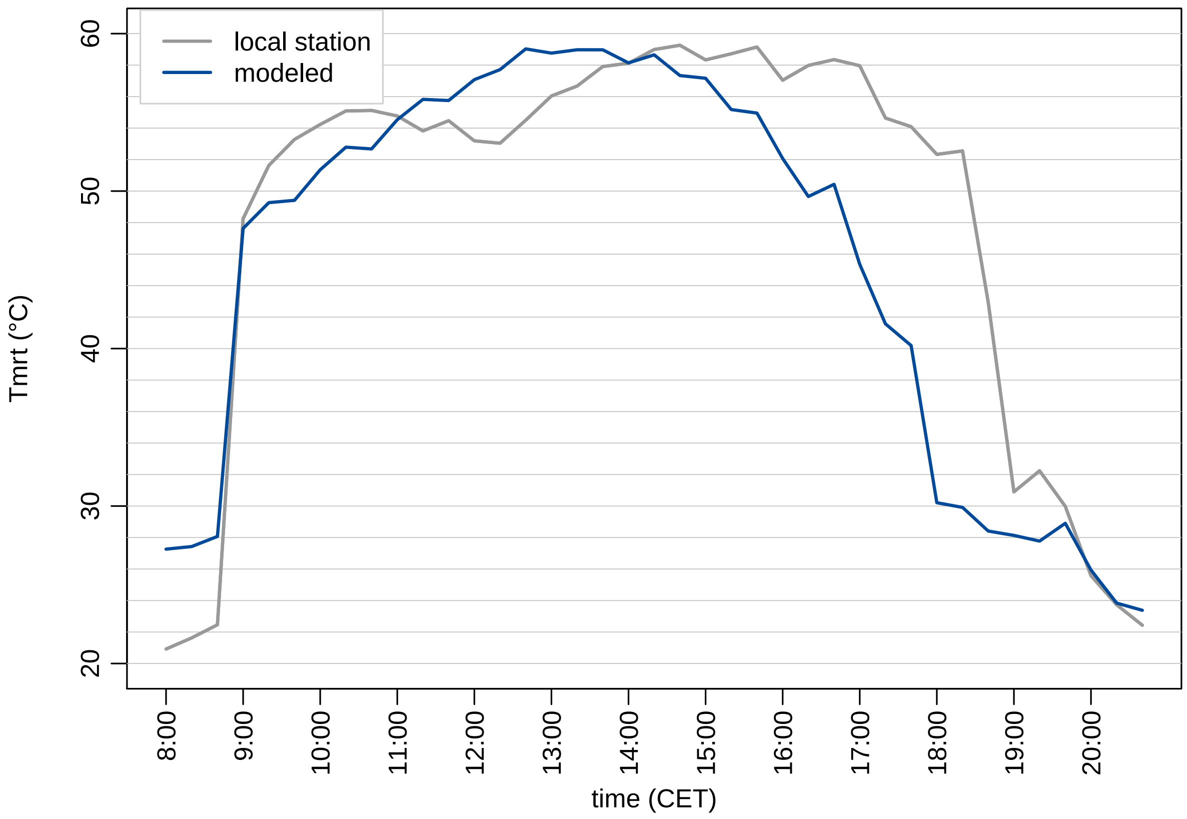


**fig. 7.37:** Comparison of mean radiant temperature ($T_{mrt}$) on the place of the old synagogue on the 01$^{st}$ of July 2008 as measured by a local biometeorological station (grey) and calculated by the SkyHelios model (blue).

The first parameter in the comparison with interdependencies with other parameters is the mean radiant temperature. To understand the results for $T_{mrt}$, those for global radiation (see section 7.3.1) and the ones for wind speed (compare to section 7.3.2) need to be considered. Comparing the values for $T_{mrt}$ calculated by the advanced SkyHelios model to those calculated based on six-directions measurements on-site one can see, that they are in fairly well agreement (see fig. 7.37). Deviations between 8:00 and 9:00 can be explained by those found for global radiation. The same holds for the time in between 17:00 and 19:00. An overall different pattern around noon can also be seen in fig. 7.37. As this can not be explained through differences in global radiation or wind speed (that anyway only shows slight impact through surface temperature), this is most likely due to other radiation fluxes (longwave or horizontal shortwave radiation fluxes).

### 7.3.5 Physiologically Equivalent Temperature

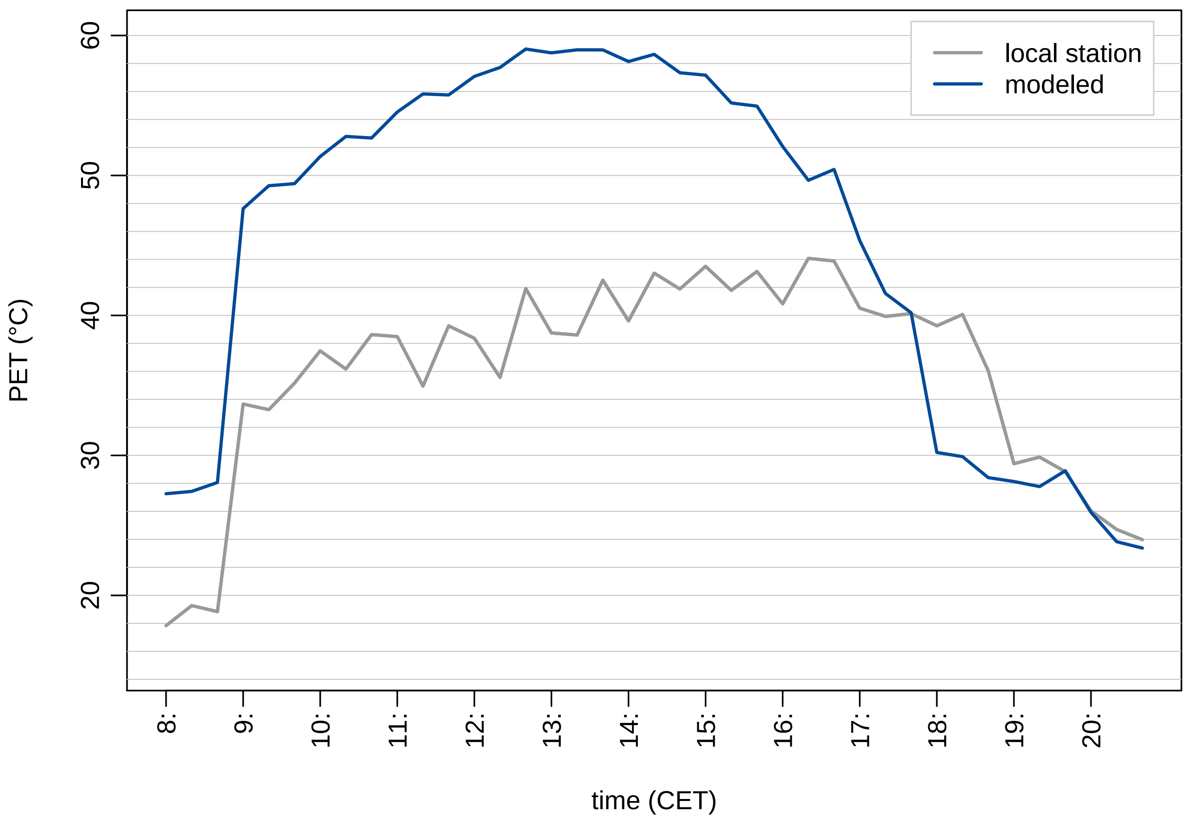


**fig. 7.38:** Comparison of Physiologically Equivalent Temperature (PET) on the place of the old synagogue on the 01$^{st}$ of July 2008 as calculated based on measurements by a local biometeorological station (grey) and calculated by the SkyHelios model (blue).

Depending on all the parameters described above, the thermal index Physiologically Equivalent Temperature also contains all their errors. This can be seen easily comparing PET calculated based on the measurements by the local biometeorological station and modelled by the advanced

SkyHelios model (fig. 7.38). While the general pattern is similar, PET calculated by the advanced SkyHelios model is way hotter than the one calculated based on the measurements from 08:00 until 17:30. At 17:30 the values almost perfectly agree. Modelled PET is colder afterwards until 19:30, from where the values agree pretty well.
The overestimation of PET compared to PET based on measurements for most of the day can mostly be explained through the severe underestimation in wind speed. As wind speed, especially in hot conditions, strongly affects PET (refer to section 4.3.6.3), a rather huge deviation of up to 20.4°C is created by that. Other deviation is caused by the differences in $T_{mrt}$ (compare to img.ComparisonModelGruberFixTmrt.pdf). As $T_{mrt}$ is way lower in the model, than for the station in the time between 16:00 to 19:00, this first compensates the overestimation if PET due to the low wind speed. From 17:30 on, an overcompensation can be seen, and PET in the model becomes smaller than for the recorded values until 19:30.

### 7.3.6 Perceived Temperature

A similar situation as for PET can be seen for the Perceived Temperature. In a direct comparison of PT calculated based on modelled (blue line in fig. 7.39) and measured input parameters (grey line in fig. 7.39) a similar deviation can be seen.
As for PET, the modelled values for PT are way higher than those based on measurements for the time in between 08:00 and 18:00. After 18:00, modelled PT is slightly lower than the one calculated for the measurements. Again, most of the deviation can be explained by the underestimation in wind speed. As PT is even more sensitive to wind speed than PET (see section 4.3.6.1), the deviation caused by the underestimation of wind speed is even larger, reaching values of about 25.0°C.
After 16:00 the lower values for $T_{mrt}$ (compare to fig. 7.37) are causing modelled PT to approach the one calculated based on the on-site measurements until they are even slightly lower from 18:00 on.

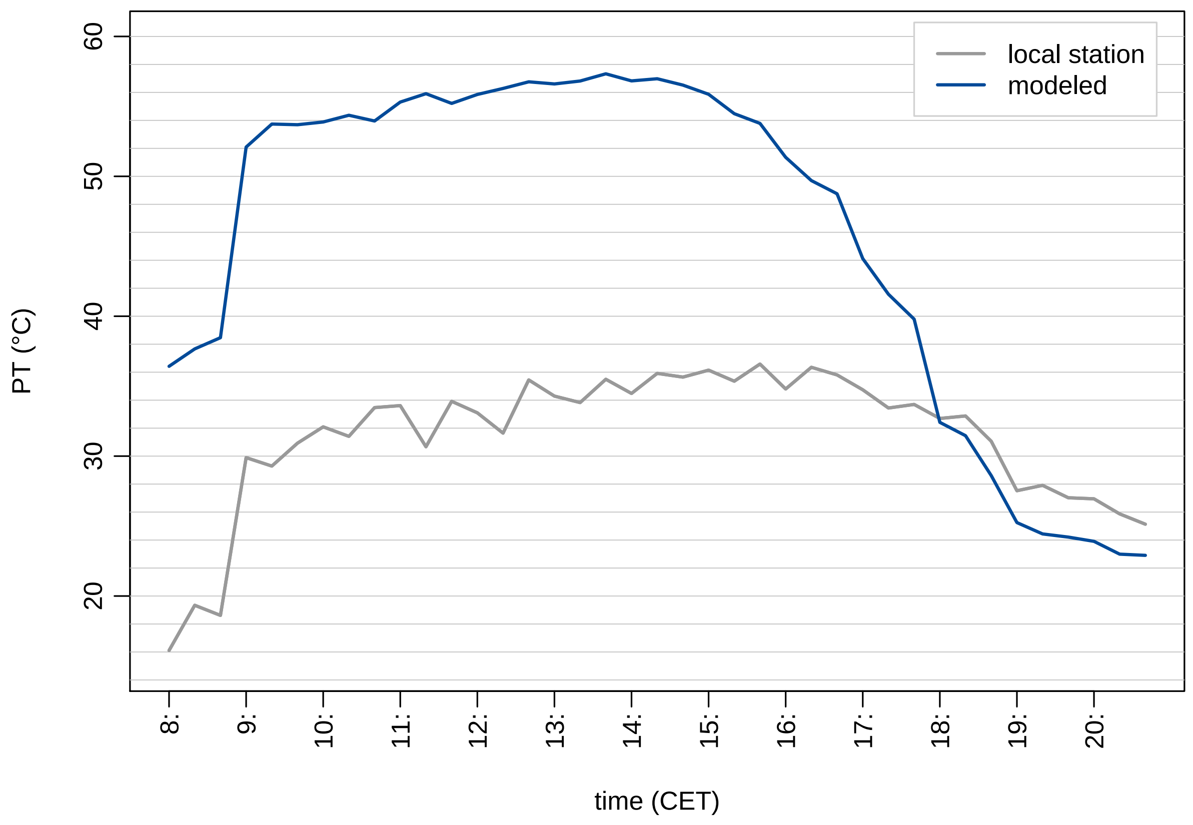


**fig. 7.39:** Comparison of Perceived Temperature (PET) on the place of the old synagogue on the 01$^{st}$ of July 2008 as calculated based on measurements by a local biometeorological station (grey) and calculated by the SkyHelios model (blue).

# 8 Discussion

The SkyHelios model is extended by several modules and functionalities. Lots of the new functionality deserve discussion concerning physics and implementation. The same, of course, also holds for the results gathered for the two test cases.

## 8.1 Implementation of a diagnostic wind model

While it is hard to determine wind speed and direction in complex obstacle settings like cities (Hosker 1985), this becomes even harder, if a model also has to do that in sufficient quality and in very short time. Röckle (1990) provides a powerful solution to this challenge by introducing a model strategy that can deliver spatially resolved data in a fairly short time for a street to neighbourhood scale. This strategy is applied in the new diagnostic wind module integrated into SkyHelios.
Concerning the usage of numerical modelling, there always are a lot of things to discuss. This holds especially for a new model with modified parametrizations. In this section, first the different steps taken for the creation of the initial wind filed will be discussed. Subsequently, the minimisation of divergence will be reconsidered. Finally, the results will be discussed.

### 8.1.1 Construction of the initial wind field

The first important thing done by a diagnostic wind model is the construction of an initial wind field. This is done by several sub routines. During this process, a lot of parametrizations are used that shall be discussed here. Also some pre-assumptions had to be made that should be mentioned in this subsection.
The model currently calculates a wind field for every wind speed and direction entered by the user. However, the parametrizations applied may fail or deliver inappropriate results for very low, or very high wind speed. As the parametrizations currently applied in the model are identical with, or are quite similar to those described in Röckle (1990), Bagal et al. (2004) and Singh et al. (2008), and are applied in other models for quite some time now, severe imprecision is unlikely.

#### 8.1.1.1 Construction of an obstacle list

The "SepBuild" method (see section 6.1.1) used to construct the internal buildings table (compare to tab. 6.1) from the obstacle grid passed by the calling program (SkyHelios) is based on a rather simple, but very safe method (described in section 6.1.1). Thus, every small part of an obstacle will be considered in the calculations. On the other hand the calculation of the obstacles table may take a lot of time in the case of many complex shaped obstacles present in the model area. Also the routine will produce a lot of very similar obstacles to consider during the construction of the initial wind field, slowing down the wind model as a whole. A modification of this routine that checks for very similar buildings, and removes them, could significantly decrease calculation time for all the further modules. On the other hand, this would possibly reduce precision in the results.
A better solution is speeding up the determination of the individual obstacles. This is achieved without loss of precision by running the main iteration of the sub-module in parallel.
To avoid redundant calculations, the obstacles table is saved into a temporary text file. If a wind model is requested later on concerning the same area of interest, the obstacles table is read from the temporary file. With the two acceleration techniques implemented, the method is considered sufficiently efficient. However, it still takes some time to generate the obstacle list for a new area of input.
Another improvement to decrease the number of obstacles is to use classes for the porosity index. This could be implemented easily and might reduce the number of obstacles within a model area considerably, if there are many obstacles with high fluctuation of porosity in neighbouring cells, e.g. many different trees. As SkyHelios only supports vegetation with static porosity, this acceleration technique is neglected as the impact on performance is considered very low.

#### 8.1.1.2 Initialisation of the velocity grids

The method to set the vertical wind profile during the initialisation of the three velocity grids assumes an urban canopy profile (section 4.3.1.1) for each horizontal cell center within the area of interest using a custom $z_0$ and $z_d$. This bears some uncertainties that have to be considered.
The urban canopy profile after Macdonald (2000) is a commonly used assumption for the vertical wind profile within an urban environment. The urban canopy profile is considered a major improvement compared to the more commonly applied logarithmic profile Röckle (e.g. applied by  1990) as the logarithmic velocity profile (see section 4.3.1.1) must be assumed invalid for any height below the displacement height $z_d$.
The new methodology to calculate a reference speed at the model top using a station profile and calculating individual vertical velocity profiles for each horizontal grid cell is assumed to be very stable and to deliver the best possible precision for the initial guess. However, it has to be considered

that this methodology can lead to rather sharp transitions due to high spatial variation in $z_0$ and $z_d$. The parametrizations used in the further calculations to determine the wind field are designed and tested to deal with a homogeneous initial velocity grid only. Sharp transitions in the initial speed grids might therefore lead to inaccuracy. However, such inaccuracy can not be seen in the results for the two test domains. In case the spatial variability in $z_0$ and $z_d$ lead to unacceptable inaccuracy, statistical smoothing of the $z_0$ and $z_d$ distribution might be able to improve the results significantly.
In the current implementation, the wind model does not account for atmospheric stability calculating the initial profile. As the module uses an atmospheric stability index $\alpha$ to calculate probability of flow over or around an obstacle, this possibly could be used to improve the vertical profile of the initial wind field as well.
Assuming a vertical speed of 0 m/s at the beginning may not meet the local conditions of a real model area. While it is hard to consider local thermal dynamics in a time-independent environment, the consideration of a terrain model would most likely strongly improve the initial guess.

#### 8.1.1.3 Calculation of the front eddy zone

The "FrontEddyBagal" method (see section 6.1.3 for details) applies a quite new parametrization as introduced by Bagal et al. (2004). It is, on the one hand, much more complex than the one proposed by Röckle (1990) and therefore requires more computation time. On the other hand, the new parametrization allows for increased precision (Bagal et al. 2004). As the computational effort generated by this method is low anyway, the increased precision is considered more important.
This agrees very well to a study by Singh et al. (2008) comparing the results calculated by this parametrization to experimental data. Singh et al. (2008) conclude, that the stagnation zone calculated by the original parametrization after Röckle (1990) is too large in space and "poorly predicts the velocities in this region" (Singh et al. 2008).

#### 8.1.1.4 Position and initial conditions of the lee-side recirculation

The method to calculate position and modifications of an obstacles recirculation zone (refer to section 6.1.4) is based on a parametrization proposed by Röckle (1990). This routine is rather fast. However, it is found to over-estimate the recirculation in the lee of an obstacle (Pardyjak et al. 2004). While the size and the impact of the recirculation could be fitted easily, the shelter model by Taylor and Salmon (1993) that is used in the QUIC model, is found to provide values that better meet those from wind tunnel experiments (Pardyjak et al. 2004). The disadvantage of the method presented in Taylor and Salmon (1993) is the required drag coefficient ($C_D$). As $C_D$ is strongly depending on

wind direction (Claus et al. 2012), it would have to be determined for each obstacle and all wind directions separately.
The overestimation of wind speed within the recirculation as calculated using the parametrization proposed by Röckle (1990) is found to be mostly due to unfavourable building parameters (width to height or length ratio), that lead to a huge $L_R$. As shown by the results, the modified equation for the determination of $L_R$ successfully limits the maximum recirculation length and leads to more reasonable results.
However, for some very unfavourable shapes of obstacles together with an incident wind direction close to perpendicular, the recirculation can become oddly shaped like it was only quarter-ellipse shaped. This is due to obstacles are split into many smaller obstacles matching a Cartesian grid. Some of the sub-obstacles might become quite wide, but only one cell long. If the wind direction is very unfavourable the one cell long sub-obstacle will form the corner of the whole obstacle. This leads to a very wide and a very small ellipse. The very sparse ellipse can thereby be thinner than one cell, depending on incident wind direction.

#### 8.1.1.5 Position of and initial conditions inside a street canyon

The method searching for obstacle configurations, that could cause a street canyon vortex, the "StreetCanyonSingh" method (refer to section 6.1.6), is implemented into the wind model based on an improved parametrization proposed by Singh et al. (2008).
A study comparing the parametrization by Singh et al. (2008) using QUIC-URB (see section 4.3.4.5 for details) and the original parametrization by Röckle (1990) to experimental data found reasonable agreement of the measured data to the modelling results (Singh et al. 2008). However, the original street canyon parametrization after Röckle (1990) was also found to overestimate wind speed in general (Singh et al. 2008).
The determination of the location of a street canyon is based on a modified version of the methodology proposed by Singh et al. (2008). The methodology is relying on $L_R$ projected in wind direction instead of iterating over the whole domain in x and y direction as proposed by Röckle (1990). The methodology after Röckle (1990) therefore is faster, if the number of obstacles is very high while the one proposed by Singh et al. (2008) will be computationally cheaper for large model domains. Is is preferred in this case as it is more safe for model domains with highly irregular obstacle configurations.
The modification of the methodology determining the size and location of street canyon vortices was necessary, as the original one proposed by Singh et al. (2008) detected too many very short street canyons. Results show, that this error does not occur using the modified methodology.

### 8.1.2 Reduction of divergence in the initial wind field

The method used to minimize the divergence is based on Successive Over-Relaxation (see section 4.3.4.2), what can be considered a quite common approach. According to the results, the method seems to reduce divergence fairly well. A disadvantage of the currently implemented routine is that divergence can never be removed completely.
The remaining divergence can be controlled easily by setting the abort level. Precision can be increased using a smaller abort level. However, this is only done, if the default value is overwritten. This step is not taken by default for two reasons: First, a decreasing abort level will strongly increase run time. Second, the current level results in a final wind field that is already suitable for the use in an urban biometerological model. Thus, another reduction of divergence for the cost of computation time is not assumed to be necessary. The default value of $10^{-5}$ therefore is an rather arbitrary value, that was found to provide sufficient precision within a reasonable computation time.
The current implementation of the SOR routine is quite fast. In case of the test domain "Institutes Quarter", it took the routine approximately five minutes to reach the abort criteria. This can hardly be speeded up any further, as SOR can not be parallelized, what must be considered a major disadvantage of this "fast" method.
The implementation as unmanaged C++ code is considered to provide maximum performance. Other improvement to the runtime of the SOR method, however, could be achieved by further code optimisation.
If the module will still be found to be to expensive after the steps mentioned above are taken, the SOR method could be replaced by a full-multigrid method (see section 4.3.4.2 for details). This approach is tested to provide large reductions in computational time compared to the SOR method (Wang et al. 2005). Wang et al. (2005) found the calculation time to be reduced by 20 to 30 times using the multigid method compared to a red-black Seidel method. Another advantage of the multigrid method is the opportunity to split up the calculations to use several CPUs at the same time. As computers tend to have more and more CPU cores, this could provide significantly reduced calculation time.

### 8.1.3 Integration of the wind model into the SkyHelios model

The wind model described above is intended to be integrated into the urban biometeorological model SkyHelios (section 4.4.2). To achieve this some preconditions have to be fulfilled.
First, the SkyHelios model needs to be able to communicate with the wind model in some way. As SkyHelios is written in Visual C#, it is able to call Visual C# and Visual C++ methods directly through the .Net framework. Managed code is, however, slightly slower than unmanaged code (like e.g. C++). A second option to integrate code into a Visual C# program is calling an unmanaged

.dll via the PInvoke interface. This leads to a slower call of the method, but allows for running faster, unmanaged code. As the wind model only needs to be called once and then is running on its own until returning the results, the second option is considered the faster one. The slight loss of computation time during the call is expected to be more than re-gained during the wind models calculations.
An important ability for a wind model developed to be integrated into the SkyHelios model is the ability to calculate independent wind fields for many independent sets of meteorological input conditions in a rather short time. This is important as SkyHelios is designed to calculate long data series with unknown or varying timesteps. While a prognostic model is more simple and can deliver very good precision, it would require a prognostic model to set up a new wind field for every row of data and iterate until stable conditions are achieved. As this would take too much time, a diagnostic wind model is considered the better choice for the use together with the SkyHelios model. The decision about the model type is therefore made in favour of a time independent diagnostic model like the one described above.
Calculating wind fields can be done saving memory space by updating the main flux grids while calculating parts of the influence of individual obstacles (e.g. the lee-side recirculation). However, this requires the main flux grids to be updated again and again. A faster method is to store temporary flux grids in all relevant directions for all the sub modules. While this is more efficient regarding computation time, it requires a lot of memory space limiting the maximum model domain size, that can be calculated.
Long data series will also require the wind model to be quite fast as it will be called again and again for each set of data. A slow wind model would therefore severely increase calculation time for the main model and would therefore prevent the user from calculating datasets of a length, that is sufficient for statistical significance. Therefore both, SkyHelios as well as the wind model, are designed with run-time in mind. As the limit for the maximum size of the area of interest in SkyHelios currently is rather the graphics memory, than the main memory, the wind model is developed saving computation time rather than main memory space.

## 8.2 Constant air temperature and humidity

The SkyHelios model is currently not able to calculate spatial modifications in air temperature ($T_a$) and air humidity ($RH$ or $VP$). They are therefore considered to be constant in space within the whole model area.
$T_a$ and $VP$ being constant in space can be seen as a major shortcoming in the current version of the SkyHelios model. However, spatial variation in $T_a$ is quite small. E.g. Mayer et al. (2008) found spatial variations of slightly about 1°C for the test cases on a summer day in Freiburg, South-

West Germany. While the sensitivity of most thermal indices to $T_a$ is quite high (e.g. Fröhlich and Matzarakis 2016), it will not considerably exceed the variation of $T_a$ itself.
Basically the same also holds for air humidity. Even within urban areas, $RH$ or $VP$ usually do not vary strongly within rather short spatial distances. The thermal indices response to changes in $VP$ are differing significantly. While PET does not quite seem to be influenced a lot by $VP$, UTCI and PT show some response (e.g. Fröhlich and Matzarakis 2016). However, even for the latter, the possible error is smaller than 1°C and therefore fairly small.
A module calculating spatial variation in $T_a$ and $VP$ therefore is beyond the focus of this dissertation project. However, the SkyHelios model core is redesigned to allow for adding a module for $T_a$ and $VP$ calculations to be developed in a further study.

## 8.3 Thermal Indices

Thermal indices are a commonly used way to assess thermal conditions for human beings under certain conditions. While the three thermal indices implemented into the SkyHelios model (see section 4.3.6) are considered valuable tools for many different types of studies in the field of urban thermal human biometeorology, some things need to be considered applying the model.
First of all, the indices PT, UTCI and PET are assuming a steady state. This is achieved by assuming a certain adaption time for each index. This means for the SkyHelios model that results for a certain position within the model area are valid only for the person standing there for quite some time. Imprecision must, thus, be expected if a person is moving within the model area.

### 8.3.1 Perceived Temperature

The Perceived Temperature (PT) is a commonly applied index in human thermal biometeorology (see section 4.3.6.1). For this reason, it also was included into the SkyHelios model. Still it has some shortcomings, that need to be considered.
The results for PT generally appear to be very unevenly distributed (see section 4.3.6.1 and fig. 4.14). This might be caused by the automatically adapting clothing model, that is changing the heat transfer resistance stepwise. While the described issue leads to an odd distribution of long-term results, the spatial results are distributed in a plausible way (see figs. 7.6 to 7.9, figs. 7.15 to 7.18, and figs. 7.25 to 7.28). The absolute difference between two grid values might bear slight imprecision due to changing clothing insulation steps. The overall distribution, however, is not expected to be affected a lot.

Other shortcomings within the PT calculations result from the insufficient consideration of "Person-Data" (compare to section 6.5). While the parameters age, height and weight can be passed to the method calculating PT, only the persons weight appears to be considered in the calculations. While this is a major issue for studies dealing with differences in thermal conditions for different persons, the majority of the studies dealing with spatial differences or overall frequencies remain almost unaffected.

### 8.3.2 Universal Thermal Climate Index

Another thermal index implemented in the SkyHelios model is the Universal Thermal Climate Index (UTCI). In spite it is a very new index developed by many experts in all relevant fields (Jendritzky et al. 2012) is has some shortcomings, that need further consideration.
The regression function for the estimation of UTCI does not support any other input parameter than $T_a$, $VP$, $v$ (in 10 m above ground level) and $T_{mrt}$. This means, that no physiological configuration is possible at all. However, as stated above, custom configurations for the physiology of the person is only used by few studies.
Another shortcoming of the regression equation for the determination of UTCI is the rather narrow range of valid meteorological input conditions (compare to section 4.3.6.2). Especially the short range of valid wind speed input from 0.5 m/s to 17.0 m/s in 10 m above ground leads to an unacceptable number of fail values in the results (compare to figs. 7.10, 7.29 and 10.4 to 10.6. Even worse, the fail values are not placed randomly in the results, but are occurring in areas with low wind speed only. The areas with strongest heat stress, the hot-spots, thus, are the ones that can not be assessed. This is not only unfavourable for the hot-spots are commonly considered to be the most interesting areas. The missing values only at the hottest locations also anticipate valid statistical analysis, as the error would lead to a trend.

### 8.3.3 Physiologically Equivalent Temperature

The Physiologically Equivalent Temperature (PET) is the third index supported by the SkyHelios model. It is the oldest one among the three indices, but is the most frequently applied one at the same time. This leads to PET being quite well validated with assessment scales available for many climates. However, also PET has certain shortcomings, that need to be mentioned.
PET allows not only for meteorological input, but also for the configuration of the sample person by lots of different parameters (see tab. 6.3). While this allows for very detailed settings, it has to be considered, that PET is only validated for few different settings.
The clothing model in PET is, unlike for PT and UTCI, not self adapting, but controlled by user input. It therefore takes a value of the clo index (see section 4.3.6.3). However, the clo value stated does

only influence the results very slightly. The strong underestimation of the clothing impact leads to very high results in warm, and very low results in cold conditions, what needs to be considered interpreting the results.
The most severe shortcoming in the calculation of PET is considered to be the underestimation of the influence by air humidity (e.g. Fröhlich and Matzarakis 2016). However, this shortcoming is less severe concerning the SkyHelios model, as SkyHelios currently uses the same value for VP for within the whole area of interest (see above). The error therefore is still present in the absolute numbers of the results for PET, but the spatial distribution remains significant.

## 8.4 Discussion of the test cases

The advanced SkyHelios model appears to be stable and to provide valid spatially resolved meteorological output as well as spatially resolved thermal indices. However, the results for the two test domains comprise some issues that need to be considered.
In the course of this dissertation project, the enhanced SkyHelios model was tested using two different test domains (see section 6.8 for details). However, both test domains have several things in common. They are both located in Freiburg, South-West Germany, what is to be considered very close on a global scale. Furthermore, the spatial input files for both test domains are based on the projected reference coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467). This leads to the uncertainty, that the model could fail or calculate results with huge imprecision for model areas in other parts of the world or other with spatial input files based on other geographic projections. However, the model was tested with different spatial input based on different projections, as well as different geographic locations (e.g. for Tainan, South-West Taiwan) during development without noticeable issues.
All the test runs for both of the domains presented in chapter 7 were performed with the same meteorological data input file holding the same four sets of meteorological input conditions (compare to tab. 6.4). This is done to facilitate intercomparison of the results. However, it also bears the uncertainty that there might be errors in the model that only occur, or only can be seen in the results for meteorological input conditions other than present in tab. 6.4. Therefore, of course, tests with other meteorological input conditions, as well as for other areas of interest have been performed during development of the model. However, they could not be presented in chapter 7 for the chapter would have become too large. As they did not reveal any new issues or abnormal behaviour by the model, they are considered neglectable for this work.

### 8.4.1 Comparison to measured data

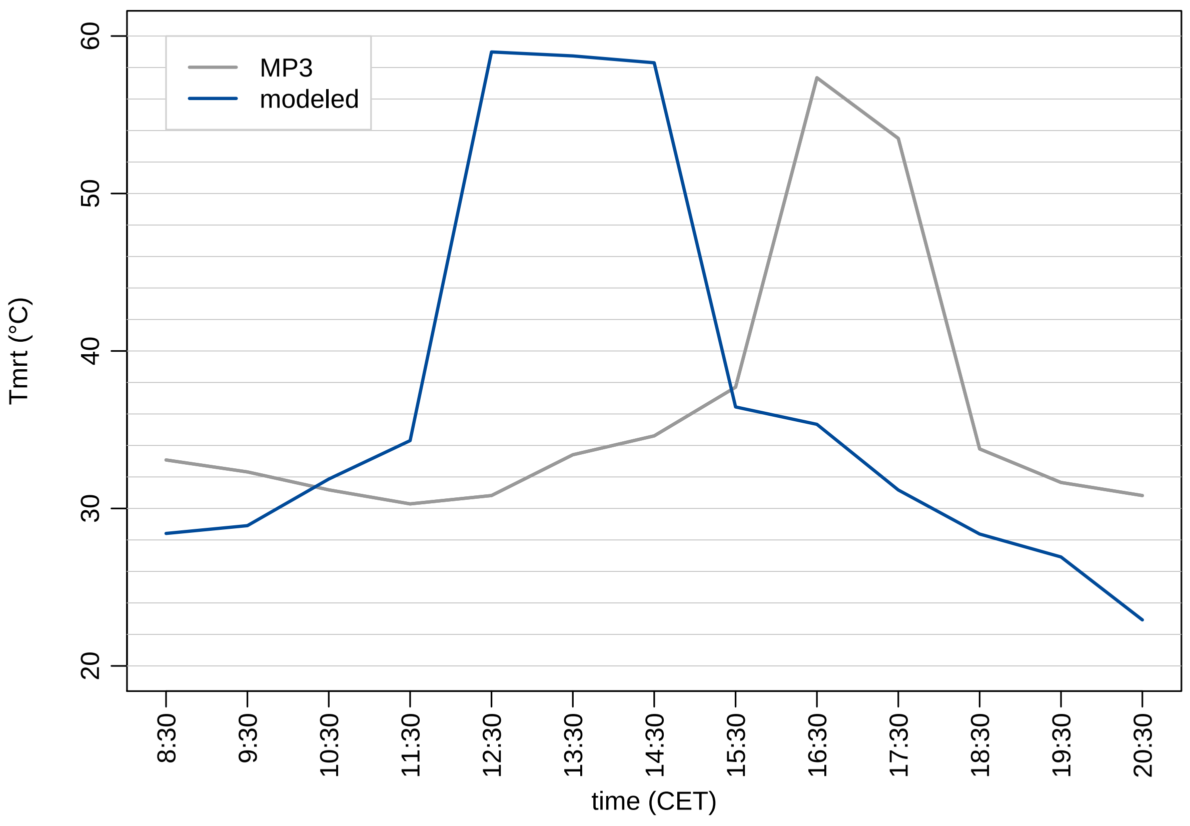


**fig. 8.1:** Comparison of mean radiant temperature ($T_{mrt}$) on the place of the old synagogue on the 01$^{st}$ of July 2008 as measured at measuring point 3 (MP3) by a mobile biometeorological station (grey) and calculated by the SkyHelios model (blue).

Comparing the results calculated by the advanced SkyHelios model to measured data is considered an important and valuable part of the model testing as it can give an insight on the models performance and the quality of the generated data. However, for the comparison described in sections 6.8.3 and 7.3, there are some aspects, that need to be considered.
First of all, the comparison is based on relatively few data. While the local climate station provided data for the time in between 08:00 and 21:00 three times an hour, resulting in 39 sets of data, the mobile station reports only once per hour for the three measurement points. For the mobile points, thus, there is only 14 sets of data available for the comparison. The small number of data sets, of course, limits the statistical significance of the comparison's results. However, a general agreement or huge inaccuracy will very likely be visible also in the small data set.
Other uncertainty of the comparison results from the limitation of the measurement campaign to one

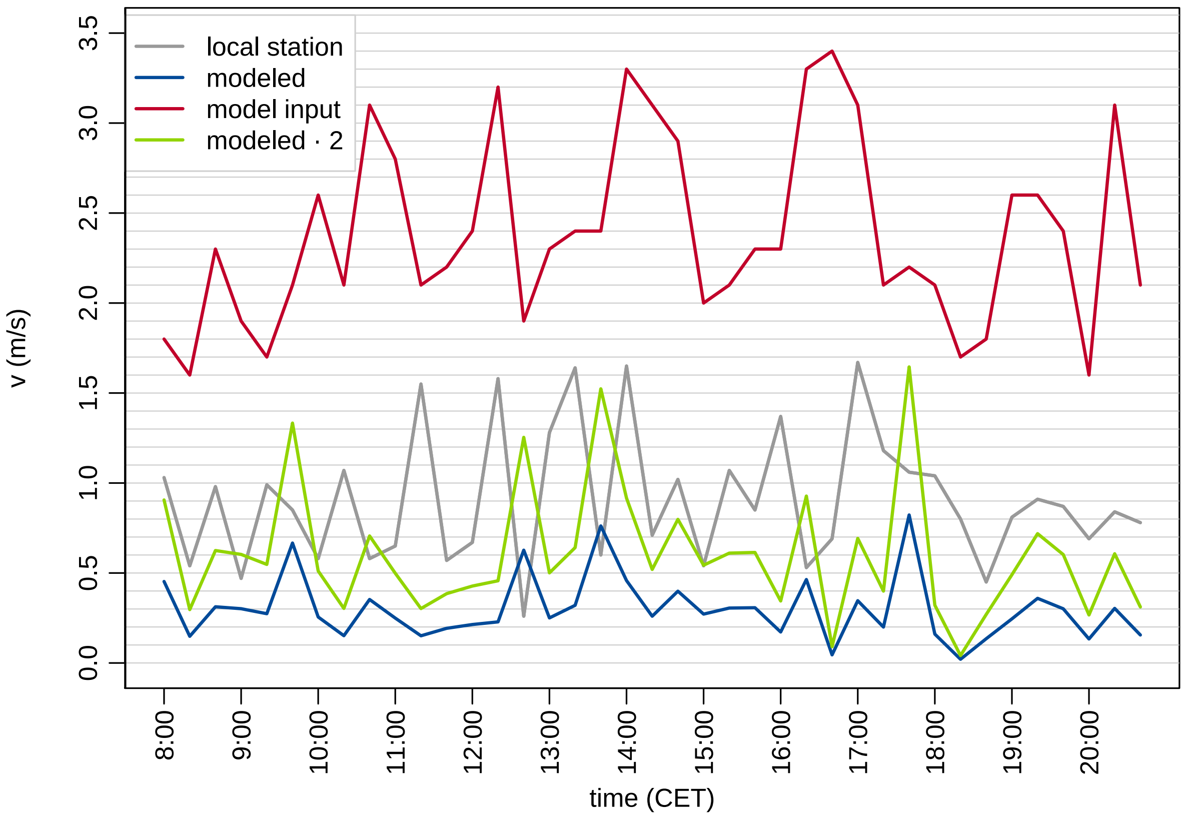


**fig. 8.2:** Comparison of wind speed ($v$) on the place of the old synagogue on the 01$^{st}$ of July 2008 as measured by a local biometeorological station (grey) and calculated by the SkyHelios model (blue). The model input provided by the urban climate station Freiburg is depicted in red. The modeled values corrected by factor 2 are shown in green color.

day in July 2008. While the meteorological conditions on this day are very interesting for thermal stress analysis, they are not changing a lot over the whole day. The models performance and the quality of the results are therefore only compared for a fairly short range of input conditions. However, the meteorological parameters provided by the measurements are typical hot summer day conditions and therefore are quite similar to those of many studies in the field of urban biometeorology.
The advanced SkyHelios model is capable of running in very high spatial resolution. For the calculations to be compared to the measurements this was 1 m on 1 m. The high spatial resolution is considered an advantage in general. It may, however, lead to uncertainty comparing to the measurements as the positions of the local biometeorological station and the mobile station are only known approximately. As many parameters, especially wind speed (see fig. 7.24) and direction, as well as the radiation fluxes (compare to fig. 7.5) show strong spatial variation within urban areas,

a mispositioned station can lead to rather strong inaccuracy. The same holds for the quality of the spatial input data. All inaccuracy and generalisations within the spatial input may lead to rather huge inaccuracy (e.g. if this causes a location to be exposed to direct sunlight in the model while being shaded in reality, refer to e.g fig. 8.1, 14:30 to 19:30). The presence of inaccuracy in the positions of the on-site stations or the spatial input can be seen through the differences in the Sky View Factor (section 4.3.5.1) as shown by tab. 8.1. This needs to be considered interpreting the results.

**tab. 8.1:** Spheric Sky View Factor as determined from fish-eye Photographs during the measurement campaign in 2008 and as calculated by the advanced SkyHelios model.

| Name | Measurements | Model |
|---|---|---|
| fix | 0.61 | 0.66 |
| MP1 | 0.37 | 0.48 |
| MP2 | 0.67 | 0.74 |
| MP3 | 0.39 | 0.52 |

While this can explain a good part of the imprecision, this does not hold for the underestimation of wind speed, that can be seen for all the points compared (figs. 7.35 and 10.7 to 10.9). Thereby it can also be seen, that wind speed recorded by the mobile station at the measuring points MP1 to MP3 is implausibly high. It is partly even exceeding wind speed recorded at the urban climate station Freiburg at roof-top level. Modelled wind speed, however does agree quite well to the measured values if corrected by factor 2.0 (see green line in fig. 8.2). Doing so does reveal the model predicts the pattern fairly well. The resulting mismatch can be explained in the uncertainty of the stations position, as well as the insufficient representation of the environment by the spatial model input. Correcting wind speed also leads to a quite good agreement between PET and PT calculated by the model and calculated for the on-site measurements (see figs. 10.18 and 10.22).

# 9 Concluding Remarks

The assessment of human thermal perception within complex urban areas can be quite challenging. This holds especially for cases, where spatial information is desired. Such information can be calculated by the advanced SkyHelios model. To achieve this, several parts had to be added to SkyHelios.

The SkyHelios model was extended by a diagnostic wind model, that can provide spatially resolved wind data for within complex settings of obstacles as shown by the results presented in sections 7.1 and 7.2. Still, some improvement can be done. In detail, the quality of the results for specific wind model elements is stated here.

The vertical profile is calculated by an improved approach after Macdonald (2000). The attempt of the improvement was to first calculate a vertical wind profile for a reference station. Based on the station profile, the wind speed at the upper model border (model top) is determined. The model top wind speed is finally used to calculate an individual vertical wind profile for each horizontal cell based on the local $z_0$ and $z_d$. This methodology is found to be very safe, as it will work with a wide range of $z_0$ and $z_d$. This allows for the consideration of spatially resolved roughness information, that can be calculated by the advanced SkyHelios model based on obstacles present in the area of interest and provided to the wind model. This makes the advanced SkyHelios model the only model in the field of human thermal biometeorology to be able to calculate wind data based on spatially resolved roughness parameters.

The size of the windward stagnation zone seems to be estimated very well, as it was found to compare well to the size of the one calculated by the models ENVI-met and QUIC-URB by a previous study. Also the wind speed modifications calculated by a parametrization after Bagal et al. (2004) is found to provide reasonable results during test runs (e.g. in sections 7.1 and 7.2).

The recirculation on the lee-side of an obstacle based on the modified parametrization agrees with the results of wind tunnel experiments (e.g. Hunt et al. (1978)) for regular obstacles and does provide a reasonable recirculation zone for irregular obstacles. However, for some unfavourable combinations of obstacle parts and incident wind direction, the shape of the recirculation might become odd. This could be fixed by checking the ratio of both half axis and moving the center point if the ratio exceeds a certain limit.

The original parametrization by Röckle (1990) used to calculate the recirculation is found to overpredict the modifications in a validation study by Singh et al. (2008). However, this was mostly due to

the overestimated recirculation size. The modified equation to calculate the maximum recirculation length is found to fix the issue and to provide a plausible recirculation (compare to e.g. sections 7.1 and 7.2).
The velocity deficit zone adjacent to the recirculation provides plausible values. They are matching the velocity at the end of the recirculation on the side facing the obstacle and the undisturbed wind speed on the other side. Results show, that due to the maximum recirculation length is determined with better quality by using the modified equation, the velocity deficit zone on the lee-side of the recirculation is both in the same place and shows a plausible length.
The calculations for the vortex in a street canyon formed by two obstacles based on the original parametrization proposed by Röckle (1990) produced a strong vortex, that extends high over the top of the surrounding obstacles as found during an evaluation of the ABC model by Singh et al. (2008). This is successfully avoided by using the parametrization after Singh et al. (2008). Also the new method to determine the location of street canyons appears to be more safe and leads to more realistic street canyon vortices and positions in the results.
Generally the new wind model can reliably provide wind speed and direction information. However, as shown by the comparison to measured data, wind speed currently is not provided in sufficient quality for further calculations by the advanced SkyHelios model, e.g. the calculation of thermal indices. According to the results (see section 7.3.2), wind speed calculated by the model for the height of 1.5 m appears to be approximately factor 2 lower than measured by the on-site station (compare to fig. 8.2). Though there is imprecision and some uncertainties found in some of the parametrizations, the model frame is found to be stable during all test runs. Since the model is developed according to modular construction, all parametrizations and service modules can be replaced or modified easily. Therefore, parametrizations and methods that are found to be inadequate during further validations can be replaced with low effort. The design based on a dynamically linked library would generally also allow the application by other programs and models. However, they would need to implement the callback method provided by the SkyHelios model.
The modified radiation module within the advanced SkyHelios model also supports roof-top global radiation input instead of determining it by astronomic calculations considering cloud cover. Results (e.g. fig. 7.4) show, that local global radiation is determined nicely for the whole area of input considering shading and the local surface temperature. Comparison to on-site measurements show, that global radiation is determined in quite good accuracy depending on the quality and accuracy of the spatial input files.
Based on global radiation, air temperature, vapour pressure and the local wind velocity provided by the wind model, the advanced SkyHelios model is able to estimate the mean radiant temperature based on local values for all relevant variables except for $T_a$ , that remains constant within the whole model area. As $T_a$ is usually only slightly varying in space within distances relevant for the SkyHelios model, this is considered to cause only slight uncertainties in $T_{mrt}$.
Comparison with on-site measurements attest the advanced SkyHelios quite good accuracy in $T_{mrt}$

calculations. Deviations of $T_{mrt}$ calculated based on the measurements as shown in the results (see section 7.3.4) are mostly due to uncertainties in the position of the on-site stations and to inaccuracy in the digital representation of the model area.

Results for all the three different model areas show, that the advanced SkyHelios model is capable of determining the human thermal indices PT, UTCI, and PET in high resolution for any urban area of interest. Thereby PT and PET can provide plausible results. UTCI, in contrary, can not be calculated for too many locations and is therefore found inappropriate for the application in urban biometeorology.

The strong overestimation of the thermal indices PT and PET as shown by the results (compare to sections 7.3.5 and 7.3.6) mostly arises from uncertainties in the position of the on-site stations and the severe underestimation of wind speed by the model. Fixing the latter is expected to strongly improve the quality of results for thermal indices calculated by the advanced SkyHelios model.

Going into detail, the first test case concerning the modification of human thermal comfort due to the redesign of the place of the old synagogue shows two aspects in the results. First, an enlarged area with extreme heat stress in the central area of the redesigned place on hot summer days. Second a slight reduction of heat stress through the design of the new university library can be seen in the area north of the library. This is due to the wider space between the library and the theatre allowing for higher wind speed for situations with incident wind direction from the South-West. From a technical perspective, the results for the Place of the old synagogue show, that the SkyHelios model can handle several different spatial input files. It therefore can be used for complex areas of interest without high effort in preparing the spatial input files prior to any calculations, what is considered a major advantage compared to most other models in urban biometeorology.

In the results for the Institutes Quarter the strong impact of radiation in means of $T_{mrt}$ and wind speed on PT and PET can be seen. UTCI can not be calculated for vast areas even for the most favourable situation with highest incident wind speed at 12:00 making spatial analysis impossible. The index is therefore found unfit for spatial analysis of urban thermal perception and stress within urban areas at hot summer days. The SkyHelios model, however, is shown to allow for calculation of thermal indices and meteorological parameters based on very large areas of interest in high resolution by the results for the Institutes Quarter. Most other models regularly applied in the field of urban biometeorology can either only handle quite small areas of interest, or can only calculate in rather coarse resolution. E.g. the prognostic micro-scale model ENVI-met (Bruse and Fleer 1998, Bruse 1999) does only support a maximum of 255 on 255 on 40 grid cells while for the SkyHelios model the only limit is the main system memory available.

Results for both test cases show, that the SkyHelios model can successfully consider a diurnal cycle with variation in all meteorological parameters through being fully diagnostic. Taking into account, that most other (prognostic) models in the field, e.g. ENVI-met can only consider a diurnal cycle in some of the input parameters (see section 4.4.3), this is considered a big advantage.

# Bibliography

Abreu-Harbich, L. V.; Labaki, L. C.; and Matzarakis, A., 2014: Thermal bioclimate in idealized urban street canyons in Campinas, Brazil. Theoretical and Applied Climatology 115(1), 333–340.

Arakawa, A. and Lamb, V. R., 1977: Computational Design of the Basic Dynamical Processes of the UCLA General Circulation Model. In: Chang, J., editor, Methods in Computational Physics: Advances in Research and Applications 17, 173–265. Elsevier.

ASHRAE, 2001: Fundamentals. American Society of Heating and Air-Conditioning Engineers Atlanta.

Bagal, N.; Pardyjak, E.; and Brown, M., 2004: Improved Upwind Cavity Parameterizations for a Fast Response Urban Wind Model. In: Proceedings of the AMS Symposium on Planning, Nowcasting, and Forecasting in the Urban Zone. Seattle, WA, USA.

Bottema, M., 1997: Urban roughness modelling in relation to pollutant dispersion. Atmospheric Environment 31(18), 3059–3075. doi: 10.1016/S1352-2310(97)00117-9.

Bottema, M. and Mestayer, P. G., 1998: Urban roughness mapping – validation techniques and some first results. Journal of Wind Engineering and Industrial Aerodynamics 74-76, 163–173. doi: 10.1016/S0167-6105(98)00014-2.

Boussinesq, J., 1897: Théorie de l'écoulement tourbillonnant et tumultueux des liquides dans les lits rectilignes a grande section. Gauthier-Villars et fils. Paris.

Brown, G. and Isfält, E., 1974: Solar irradiation and sun shading devices. Nat. Swedish Council for Building Research, Stockholm, Report R19.

Brunt, D., 1932: Notes on radiation in the atmosphere. I. Quarterly Journal of the Royal Meteorological Society 58(247), 389–420. doi: 10.1002/qj.49705824704.

Bruse, M., 1999: Die Auswirkungen kleinskaliger Umweltgestaltung auf das Mikroklima. Entwicklung des prognostischen numerischen Modells ENVI-met zur Simulation der Wind-, Temperatur- und Feuchteverteilung in städtischen Strukturen. PhD thesis Ruhr-Universität Bochum.

Bruse, M. and Fleer, H., 1998: Simulating surface–plant–air interactions inside urban environments with a three dimensional numerical model. Environmental Modelling & Software 13(3–4), 373–384. doi: 10.1016/S1364-8152(98)00042-5.

Brutsaert, W., 1975: Comments on surface roughness parameters and the height of dense vegetation. J. Meteorol. Soc. Japan 53, 96–97.

Błażejczyk, K.; Epstein, Y.; Jendritzky, G.; Staiger, H.; and Tinz, B., 2012: Comparison of UTCI to selected thermal indices. International Journal of Biometeorology 56(3), 515–535. doi: 10.1007/s00484-011-0453-2.

Błażejczyk, K.; Jendritzky, G.; Bröde, P.; Fiala, D.; Havenith, G.; Epstein, Y.; Psikuta, A.; and Kampmann, B., 2013: An introduction to the Universal Thermal Climate Index (UTCI). Geographia Polonica 86(1).

Charalampopoulos, I.; Tsiros, I.; Chronopoulou-Sereli, A.; and Matzarakis, A., 2013: Analysis of thermal bioclimate in various urban configurations in Athens, Greece. Urban Ecosystems 16(2), 217–233. doi: 10.1007/s11252-012-0252-5.

Charalampopoulos, I.; Tsiros, I.; Chronopoulou-Sereli, A.; and Matzarakis, A., 2015: A note on the evolution of the daily pattern of thermal comfort-related micrometeorological parameters in small urban sites in Athens. International Journal of Biometeorology 59(9), 1223–1236. doi: 10.1007/s00484-014-0934-1.

Chen, Y.-C. and Matzarakis, A., 2014: Modification of physiologically equivalent temperature. Journal of Heat Island Institute International (9), 26–32.

Chen, Y.-C.; Lin, T.-P.; and Matzarakis, A., 2014: Comparison of mean radiant temperature from field experiment and modelling: a case study in Freiburg, Germany. Theoretical and Applied Climatology 118(3), 535–551. doi: 10.1007/s00704-013-1081-z.

Cionco, R., 1965: Mathematical Model for Air Flow in a Vegetative Canopy. Journal of Applied Meteorology 4, 517–522.

Cionco, R., 1972: A Wind Profile Index For Canopy Flow. Boundary-Layer Meteorology 3(2), 255–263. doi: 10.1007/BF02033923.

Clarke, J. and Bach, W., 1971: Comparison of the comfort conditions in different urban and suburban micro environments. International Journal of Biometeorology 15(1), 41–54. doi: 10.1007/BF01804717.

Claus, J.; Krogstad, P.-A.; and Castro, I. P., 2012: Some Measurements of Surface Drag in Urban-Type Boundary Layers at Various Wind Angles. Boundary-Layer Meteorology 145(3), 407–422. doi: 10.1007/s10546-012-9736-3.

Compagnon, R. and Raydan, D., 2000: Irradiance and Illumination Distributions in Urban Areas. In: Architecture, city, environment : proceedings of PLEA 2000. Cambridge, UK. James & James (Science Publishers) Ltd.

Conti, S.; Meli, P.; Minelli, G.; Solimini, R.; Toccaceli, V.; Vichi, M.; Beltrano, C.; and Perini, L., 2005: Epidemiologic study of mortality during the Summer 2003 heat wave in Italy. Environmental Research 98(3), 390–399. doi: 10.1016/j.envres.2004.10.009.

Counihan, J., 1975: Adiabatic atmospheric boundary layers: A review and analysis of data from the period 1880–1972. Atmospheric Environment (1967) 9(10), 871–905. doi: 10.1016/0004-6981(75)90088-8.

Crawford, T. M. and Duchon, C. E., 1999: An Improved Parameterization for Estimating Effective Atmospheric Emissivity for Use in Calculating Daytime Downwelling Longwave Radiation. J Appl Meteorol 38, 474–480.

Davenport, A. G.; Grimmond, C. S. B.; Oke, T. R.; and Wieringa, J., 2000: Estimating the roughness of cities and sheltered country. In: 15th conference on probability and statistics in the atmospheric sciences/12th conference on applied climatology, Ashville, NC, American Meteorological Society, 96–99.

Duarte, H. F.; Dias, N. L.; and Maggiotto, S. R., 2006: Assessing daytime downward longwave radiation estimates for clear and cloudy skies in Southern Brazil. Agricultural and Forest Meteorology 139(3-4), 171–181. doi: 10.1016/j.agrformet.2006.06.008.

Eichhorn, J., 1989: Entwicklung und Anwendung eines dreidimensionalen mikroskaligen Stadtklima-Modells. PhD thesis University of Mainz.

Eichhorn, J. and Kniffka, A., 2010: The numerical flow model MISKAM: State of development and evaluation of the basic version. Meteorologische Zeitschrift 19(1), 81–90. doi: 10.1127/0941-2948/2010/0425.

Emmanuel, R. and Johansson, E., 2006: Influence of urban morphology and sea breeze on hot humid microclimate: the case of Colombo, Sri Lanka. Climate Research 30(3), 189–200. doi: 10.3354/cr030189.

Endler, C., 2010: Analyse von hochaufgelösten Klimasimulationen für die Schwarzwaldregion. Eine tourismus-klimatische Perspektive. Inaugural-Dissertation, Albert-Ludwigs-Universität Freiburg.

Fanger, P., 1972: Thermal comfort. McGraw-Hill. New York.

Fiala, D.; Havenith, G.; Broede, P.; Kampmann, B.; and Jendritzky, G., 2012: UTCI-Fiala multi-node model of human heat transfer and temperature regulation. International Journal of Biometeorology 56(3), 429–441. doi: 10.1007/s00484-011-0424-7.

Fortune, S., 1987: A sweepline algorithm for Voronoi diagrams. Algorithmica 2(1-4), 153–174. doi: 10.1007/BF01840357.

Fröhlich, D. and Matzarakis, A., 2011: Hitzestress und Stadtplanung - Am Beispiel des „Platz der alten Synagoge“ in Freiburg im Breisgau. Gefahrstoffe – Reinhaltung der Luft 71(7/8), 333–338.

Fröhlich, D. and Matzarakis, A., 2013: Modeling of changes in thermal bioclimate: examples based on urban spaces in Freiburg, Germany. Theoretical and Applied Climatology (111), 547–558. doi: 10.1007/s00704-012-0678-y.

Fröhlich, D. and Matzarakis, A., 2016: A quantitative sensitivity analysis on the behaviour of common thermal indices under hot and windy conditions in Doha, Qatar. Theoretical and Applied Climatology 124(1-2), 179–187. doi: 10.1007/s00704-015-1410-5.

Gagge, A.; Fobelets, A.; and Berglund, L., 1986: A standard predictive index of human response to the thermal environment. ASHRAE Transactions (92), 709–731.

Gal, T.; Lindberg, F.; and Unger, J., 2009: Computing continuous sky view factors using 3d urban raster and vector databases: comparison and application to urban climate. Theoretical and Applied Climatology 95(1-2), 111–123. doi: 10.1007/s00704-007-0362-9.

GDAL Development Team, 2016: GDAL - Geospatial Data Abstraction Library, Version 1.10.1.

Grimmond, C.; Roth, M.; Oke, T.; Au, Y.; Best, M.; Betts, R.; Carmichael, G.; Cleugh, H.; Dabberdt, W.; Emmanuel, R.; Freitas, E.; Fortuniak, K.; Hanna, S.; Klein, P.; Kalkstein, L.; Liu, C.; Nickson, A.; Pearlmutter, D.; Sailor, D.; and Voogt, J., 2010: Climate and More Sustainable Cities: Climate Information for Improved Planning and Management of Cities (Producers/Capabilities Perspective). World Climate Conference - 3 1(0), 247–274. doi: 10.1016/j.proenv.2010.09.016.

Grimmond, C. S. B. and Oke, T. R., 1999: Aerodynamic Properties of Urban Areas Derived from Analysis of Surface Form. Journal of Applied Meteorology 38(9), 1262–1292. doi: 10.1175/1520-0450(1999)038<1262:APOUAD>2.0.CO;2.

Grimmond, C. S. B.; King, T. S.; Roth, M.; and Oke, T. R., 1998: Aerodynamic Roughness of Urban Areas Derived from Wind Observations. Boundary-Layer Meteorology 89(1), 1–24. doi: 10.1023/A:1001525622213.

Gross, G., 1997: ASMUS - A numerical model for simulations of wind and pollutant dispersion around individual buildings. 2.: Dispersion modelling and applications. Meteorologische Zeitschrift 6(3), 130–136.

Gross, G.; Vogel, H.; and Wippermann, F., 1987: Dispersion over and around a steep obstacle for varying thermal stratification—numerical simulations. Atmospheric Environment (1967) 21(3), 483–490. doi: 10.1016/0004-6981(87)90031-X.

Gross, G.; Röckle, R.; and Janssen, U., 1994: ASMUS - Ein numerisches Modell zur Berechnung der Strömung und der Schadstoffverteilung im Bereich einzelner Gebäude. I: Das Strömungsmodell. Meteorologische Zeitschrift (3), 267–274.

Havenith, G.; Fiala, D.; Błazejczyk, K.; Richards, M.; Bröde, P.; Holmér, I.; Rintamaki, H.; Benshabat, Y.; and Jendritzky, G., 2012: The UTCI-clothing model. International Journal of Biometeorology 56(3), 461–470. doi: 10.1007/s00484-011-0451-4.

Helbig, A.; Baumüller, J.; and Kerschgens, M. J., 1999: Stadtklima und Luftreinhaltung. 2., vollständig überarbeitete und ergänzte Auflage mit 200 Abbildungen und 79 Tabellen. Springer-Verlag. Berlin; Heidelberg [u.a.] 2 edition.

Herrmann, J. and Matzarakis, A., 2012: Mean radiant temperature in idealized urban canyons – Examples from Freiburg, Germany. International Journal of Biometeorology (56), 199–203.

Hosker, R., 1985: Flow around isolated structures and building clusters. A review. ASHRAE Transactions (91, part 2b).

Hunt, J.; Abell, C.; Peterka, J.; and Woo, H., 1978: Kinematical studies of the flows around free or surface-mounted obstacles. Applying topology to flow visualisation. Journal of Fluid Mechanics (86), 179–200.

Huttner, S., 2012: Further development and application of the 3D microclimate simulation ENVI-met. PhD thesis Johannes Gutenberg-Universität Mainz.

Hwang, R.-L.; Lin, T.-P.; and Matzarakis, A., 2011: Seasonal effects of urban street shading on long-term outdoor thermal comfort. Building and Environment 46(4), 863–870. doi: 10.1016/j.buildenv.2010.10.017.

Hämmerle, M.; Gál, T.; Unger, J.; and Matzarakis, A., 2011a: Comparison of models calculating the Sky View Factor used for urban climate investigations. Theoretical and Applied Climatology (105), 521–527.

Hämmerle, M.; Gál, T.; Unger, J.; and Matzarakis, A., 2011b: Introducing a script for calculating the sky view factor used for urban climate investigations. Acta Climatologica et Chorologica (44-45), 83–92.

Hämmerle, M.; Gál, T.; Unger, J.; and Matzarakis, A., 2014: Different aspects in the quantification of the Sky View Factor in complex environments. Acta Climatologica et Chorologica (47-48), 53–62.

Höppe, P., 1984: Die Energiebilanz des Menschen. Berichte des Meteorologischen Instituts Nr. 49 Ludwigs-Maximilians-Universität München.

Höppe, P., 1992: Ein neues Verfahren zur Bestimmung der mittleren Strahlungstemperatur im Freien. Wetter und Leben (44), 47–151.

Höppe, P. R., 1993a: Heat balance modelling. Experientia 49(9), 741–746.

Höppe, P. R., 1993b: Indoor climate. Experientia 49(9), 775–779.

Höppe, P. R., 1999: The physiological equivalent temperature – a universal index for the biometeorological assessment of the thermal environment. Int J Biometeorol 43, 71–75.

ISO, 1998: International standard 7726: Thermal environments: instruments and methods for measuring physical quantities. International Standard Organization, Geneva.

Iziomon, M. G.; Mayer, H.; and Matzarakis, A., 2003: Downward atmospheric longwave irradiance under clear and cloudy skies: Measurement and parameterization. Journal of Atmospheric and Solar-Terrestrial Physics 65(10), 1107–1116. doi: 10.1016/j/jastp.2003.07.007.

Janicke Consulting, 2011: AUSTAL2000, Program Documentation of Version 2.5. Überlingen.

Jendritzky, G.; Menz, H.; Schirmer, H.; and Schmidt-Kessen, W., 1990: Methodik zur raumbezogenen Bewertung der thermischen Komponente im Bioklima des Menschen (Fortgeschriebenes Klima- Michel-Modell). Beitr Akad Raumforsch Landesplan (114).

Jendritzky, G.; Dear, de R.; and Havenith, G., 2012: UTCI-Why another thermal index? Int J Biometeorol (56), 421–428.

Johnson, G. and Watson, I., 1984: Person view-factors in the urban-environment. Archives for Meteorology Geophysics and Bioclimatology Series B-Theoretical and Applied Climatology 34(3), 273–285. doi: 10.1007/BF02265493.

Kampstra, P., 2008: Beanplot: A Boxplot Alternative for Visual Comparison of Distributions. Journal of Statistical Software, Code Snippets 1(28), 1–9.

Kanda, M.; Inagaki, A.; Miyamoto, T.; Gryschka, M.; and Raasch, S., 2013: A New Aerodynamic Parametrization for Real Urban Surfaces. Boundary-Layer Meteorology 148(2), 357–377. doi: 10.1007/s10546-013-9818-x.

Kantor, N. and Unger, J., 2011: The most problematic variable in the course of human-biometeorological comfort assessment - the mean radiant temperature. Central European Journal of Geosciences 3(1), 90–100. doi: 10.2478/s13533-011-0010-x.

Kaplan, H. and Dinar, N., 1996: A Lagrangian Dispersion Model for Calculating Concentration Distribution within a built-up domain. Atmospheric Environment 30, 4197–4207.

Kasten, F., 1980: A simple parametrization of the pyrheliometric formula for determining the linke turbidity factor. Meteorologische Rundschau 33(4), 124–127.

Kasten, F. and Young, A., 1989: Revised optical air-mass tables and approximation formula. Applied Optics 28(22), 4735–4738.

Kastner-Klein, P.; Fedorovich, E.; and Rotach, M., 2001: A wind tunnel study of organised and turbulent air motions in urban street canyons. Journal of Wind Engineering and Industrial Aerodynamics 89(9), 849–861. doi: 10.1016/S0167-6105(01)00074-5.

Ketterer, C.; Ghasemi, I.; Reuter, U.; Rinke, R.; Kapp, R.; Bertram, A.; and Matzarakis, A., 2013: Veränderung des thermischen Bioklimas durch stadtplanerische Umgestaltung. Gefahrstoffe-Reinhaltung der Luft (7-8/2013), 323–329.

Ketterer, C.; Gangwisch, M.; Fröhlich, D.; and Matzarakis, A., 2016: Comparison of selected approaches for urban roughness determination based on voronoi cells. International Journal of Biometeorology. doi: 10.1007/s00484-016-1203-2.

Kondo, J. and Yamazawa, H., 1986: Aerodynamic Roughness Over An Inhomogenous Ground Surface. Boundary-Layer Meteorology 35(4), 331–348. doi: 10.1007/BF00118563.

Koppe, C.; Kovats, S.; Jendritzky, G.; and Menne, B., 2004: Heat-waves: risks and responses 2. World Health Organization. Copenhagen.

Krueger, E. L.; Minella, F. O.; and Matzarakis, A., 2014: Comparison of different methods of estimating the mean radiant temperature in outdoor thermal comfort studies. International Journal of Biometeorology 58(8), 1727–1737. doi: 10.1007/s00484-013-0777-1.

Kuttler, W., 2000: Stadtklima. In: Guderian, R., editor, Atmosphäre 1B, 420–470. Springer Verlag.

König-Langlo, G. and Schatzmann, M., 1991: Wind tunnel modeling of heavy gas dispersion. Atmospheric Environment. Part A. General Topics 25(7), 1189–1198. doi: 10.1016/0960-1686(91)90230-5.

Landsberg, H. E., 1981: The urban climate. The academic press. London.

Lettau, H., 1969: Note on Aerodynamic Roughness-Parameter Estimation on the Basis of Roughness-Element Description. Journal of Applied Meteorology 8(5), 828–832. doi: 10.1175/1520-0450(1969)008<0828:NOARPE>2.0.CO;2.

Lin, T.-P. and Matzarakis, A., 2011: Estimation of outdoor mean radiant temperature by filed experiment and modelling for human-biometeorology use. In: Proceedings of the 11th annual meeting of the European Meteorological Society, 2011–2089. Berlin, Germany.

Lin, T.-P.; Matzarakis, A.; and Hwang, R.-L., 2010a: Shading effect on long-term outdoor thermal comfort. International Symposium on the Interaction between Human and Building Environment Special Issue Section 45(1), 213–221. doi: 10.1016/j.buildenv.2009.06.002.

Lin, T.-P.; Matzarakis, A.; Hwang, R.-L.; and Huang, Y.-C., 2010b: Effect of pavements albedo on long-term outdoor thermal comfort. In: Proceedings of the 7th Conference on Biometeorology 20 of Ber. Meteorol. Inst. Univ. Freiburg, 498–504. Freiburg.

Lin, T.-P.; Tsai, K.-T.; Liao, C.-C.; and Huang, Y.-C., 2013: Effects of thermal comfort and adaptation on park attendance regarding different shading levels and activity types. Building and Environment 59(0), 599–611. doi: 10.1016/j.buildenv.2012.10.005.

Lindberg, F.; Holmer, B.; and Thorsson, S., 2008: SOLWEIG 1.0 - Modelling spatial variations of 3d radiant fluxes and mean radiant temperature in complex urban settings. International Journal of Biometeorology 52(7), 697–713. doi: 10.1007/s00484-008-0162-7.

Lopes, A.; Lopes, S.; Matzarakis, A.; and Alcoforado, M. J., 2011: The influence of the summer sea breeze on thermal comfort in Funchal (Madeira). A contribution to tourism and urban planning. Meteorologische Zeitschrift 20(5), 553–564. doi: 10.1127/0941-2948/2011/0248.

Macdonald, R. W., 2000: Modelling the mean velocity profile in the urban canopy layer. Boundary-Layer Meteorology 97(1), 25–45. doi: 10.1023/A:1002785830512.

Macdonald, R. W.; Griffiths, R. F.; and Hall, D. J., 1998: An improved method for the estimation of surface roughness of obstacle arrays. Atmospheric Environment 32(11), 1857–1864. doi: 10.1016/S1352-2310(97)00403-2.

Matzarakis, A., 2001: Die thermische Komponente des Stadtklimas. Nr. 6 in: Berichte des Meteorologischen Institutes der Universität Freiburg.

Matzarakis, A. and Endler, C., 2009: Physiologically Equivalent Temperature and Climate Change in Freiburg. Eighth Symposium on the Urban Environment. American Meteorological Society, Phoenix/Arizona, 10. to 15. January 2009 4(2), 1–8.

Matzarakis, A. and Endler, C., 2010: Climate change and thermal bioclimate in cities: impacts and options for adaptation in Freiburg, Germany. International Journal of Biometeorology 54(4), 479–483. doi: 10.1007/s00484-009-0296-2.

Matzarakis, A. and Matuschek, O., 2011: Sky view factor as a parameter in applied climatology rapid estimation by the SkyHelios model. Meteorologische Zeitschrift 20(1), 39–45.

Matzarakis and Mayer, 1992: Mapping of urban air paths for planning in Munich. Karlsruhe.

Matzarakis, A. and Mayer, H., 1996: Another kind of environmental stress: Thermal stress. WHO Newsletter (18), 7–10.

Matzarakis, A. and Mayer, H., 2008: Importance of urban meteorological stations - the example of Freiburg, Germany. In: Celebrating the 50 Years of the Meteorological Institute, Albert-Ludwigs-University of Freiburg, Germany (17), 119–128. Self-publishing company of the Meteorological Institute, Albert-Ludwigs-University of Freiburg, Germany.

Matzarakis, A.; Mayer, H.; and Iziomon, M. G., 1999: Applications of a universal thermal index: physiological equivalent temperature. International Journal of Biometeorology 43(2), 76–84. doi: 10.1007/s004840050119.

Matzarakis, A.; Mahlau, F.; and Mayer, H., 2000: Online-visualisierung von meteorologischen Daten im Internet — Meteorologische Stadtstation Freiburg. In: Fachtagung Mettools IV, 150–152.

Matzarakis, A.; Rutz, F.; and Mayer, H., 2007: Modelling radiation fluxes in simple and complex environments - application of the RayMan model. International Journal of Biometeorology 51(4), 323–334. doi: 10.1007/s00484-006-0061-8.

Matzarakis, A.; Röckle, R.; Richter, C.-J.; Höfl, H.-C.; Steinicke, W.; Streifeneder, M.; and Mayer, H., 2008: Planungsrelevante Bewertung des Stadtklimas – Am Beispiel von Freiburg im Breisgau. Gefahrstoffe – Reinhaltung der Luft (68), 334–340.

Matzarakis, A.; De Rocco, M.; and Najjar, G., 2009: Thermal bioclimate in Strasbourg - the 2003 heat wave. Theoretical and Applied Climatology (98), 209–220.

Matzarakis, A.; Rutz, F.; and Mayer, H., 2010: Modelling radiation fluxes in simple and complex environments: basics of the RayMan model. International Journal of Biometeorology 54(2), 131–139. doi: 10.1007/s00484-009-0261-0.

Matzarakis, A.; Muthers, S.; and Koch, E., 2011: Human-biometeorological evaluation of summer mortality in Vienna. Theoretical and Applied Climatology (105), 1–10.

Mayer, H. and Höppe, P., 1987: Thermal Comfort of Man in Different Urban Environments. Theoretical and Applied Climatology (38), 43–49.

Mayer, H.; Holst, J.; Dostal, P.; Imbery, F.; and Schindler, D., 2008: Human thermal comfort in summer within an urban street canyon in Central Europe. Meteorologische Zeitschrift 17(3), 241–250. doi: 10.1127/0941-2948/2008/0285.

Muthers, S.; Matzarakis, A.; and Koch, E., 2010: Summer climate and mortality in Vienna - a human-biometeorological approach of heat-related mortality during the heat waves in 2003. Wiener Klinische Wochenschrift 122(17-18), 525–531. doi: 10.1007/s00508-010-1424-z.

Nastos, P. and Matzarakis, A., 2012: The effect of air temperature and Physiologically Equivalent Temperature on mortality in Athens, Greece. Theoretical and Applied Climatology (108), 591–599.

Ndetto, E. L. and Matzarakis, A., 2015: Urban atmospheric environment and human biometeorological studies in Dar es Salaam, Tanzania. Air Quality Atmosphere and Health 8(2), 175–191. doi: 10.1007/s11869-014-0261-z.

Nübler, W., 1979: Konfiguration und Genese der Wärmeinsel der Stadt Freiburg. Nr. 16 in: Freiburger Geographische Hefte. Freiburg.

Oke, T. R., 1995: Boundary layer climates, 2nd edition. Routledge. London, New York.

Ottmann, T. and Widmayer, P., 2012: Algorithmen und Datenstrukturen. Springer-Verlag.

Panofsky, H. and Dutton, J., 1984: Atmospheric turbulence. Models and methods for Engineering applications. John Wiley & Sons. New York.

Pardyjak, E. R.; Brown, M. J.; and Bagal, N., 2004: Improved Velocity Deficit Parameterizations for a Fast Response Urban Wind Model. In: Proceedings of the AMS Symposium on Planning, Nowcasting, and Forecasting in the Urban Zone. Seattle, WA, USA.

Population Division, 2012: World Urbanization Prospects, the 2011 Revision. United Nations, Department of Economic and Social Affairs New York.

Press, W. H.; Teukolsky, S. A.; Vetterling, W. T.; and Flannery, B. P., 2007: Numerical recipes : the art of scientific computing. Cambridge Univ. Press. Cambridge 3rd edition.

QGIS Development Team, 2016: Qgis geographic information system.

R Development Core Team, 2008: R: A language and environment for statistical computing. Vienna, Austria.

Ratti, C.; Di Sabatino, S.; and Britter, R., 2006: Urban texture analysis with image processing techniques: winds and dispersion. Theoretical and Applied Climatology 84(1-3), 77–90. doi: 10.1007/s00704-005-0146-z.

Ratto, C. F.; Festa, R.; Romeo, C.; Frumento, O.; and Galluzzi, M., 1994: Mass-consistent models for wind fields over complex terrain: The state of the art. Environmental Software (9), 247–268.

Roache, P. J., 1982: Computational Fluid Dynamics. Hermosa Publishers. Albuquerque.

Roth, M., 2000: Review of atmospheric turbulence over cities. Quarterly Journal of the Royal Meteorological Society 126(564), 941–990. doi: 10.1256/smsqj.56408.

Rudloff, H., 1993: Beiträge zum Klima Freiburgs. Lingg Druck. Freiburg.

Röckle, R., 1990: Bestimmung der Strömungsverhältnisse im Bereich komplexer Bebauungsstrukturen. PhD thesis Fachbereich Mechanik der Technischen Hochschule Darmstadt Darmstadt.

Salata, F.; Golasi, I.; Vollaro, A. d. L.; and Vollaro, R. d. L., 2015: How high albedo and traditional buildings' materials and vegetation affect the quality of urban microclimate. A case study. Energy and Buildings 99, 32–49. doi: 10.1016/j.enbuild.2015.04.010.

Sasaki, Y., 1958: An objective analysis based on the variational method. Journ. Met. Soc. Japan 36, 77–89.

Sasaki, Y., 1970a: Some basic formalisms in numerical variational analysis. Monthly Weather Review 98(12), 875–&. doi: 10.1175/1520-0493(1970)098<0875:SBFINV>2.3.CO;2.

Sasaki, Y., 1970b: Numerical variational analysis formulated under constraints as determined by longwave equations and a low-pass filter. Monthly Weather Review 98(12), 884–&. doi: 10.1175/1520-0493(1970)098<0884:NVAFUT>2.3.CO;2.

Satterlund, D., 1979: Improved equation for estimating long-wave-radiation from the atmosphere. Water Resources Research 15(6), 1649–1650. doi: 10.1029/WR015i006p01649.

Schatzmann, M.; Lohmeyer, A.; and Ortner, G., 1987: Flue gas discharge from cooling towers. Wind tunnel investigation of building downwash effects on ground-level concentrations. Atmospheric Environment (1967) 21(8), 1713–1724. doi: 10.1016/0004-6981(87)90110-7.

Schlünzen, K. H.; Hinneburg, D.; Knoth, O.; Lambrecht, M.; López, S.; Lüpkes, C.; Panskus, H.; Renner, E.; Schatzmann, M.; Schoenemeyer, T.; Trepte, S.; and Wolke, R., 2003: Flow and Transport in the Obstacle Layer: First Results of the Micro-Scale Model MITRAS. Journal of Atmospheric Chemistry (44), 113–130.

Schoetter, R.; Grawe, D.; Hoffmann, P.; Kirschner, P.; Graetz, A.; and Schluenzen, K. H., 2013: Impact of local adaptation measures and regional climate change on perceived temperature. Meteorologische Zeitschrift 22(2), 117–130. doi: 10.1127/0941-2948/2013/0381.

Shashua-Bar, L.; Pearlmutter, D.; and Erell, E., 2011: The influence of trees and grass on outdoor thermal comfort in a hot-arid environment. International Journal of Climatology 31(10), 1498–1506. doi: 10.1002/joc.2177.

Sherman, C., 1978: Mass-consistent model for wind fields over complex terrain. Journal of Applied Meteorology 17(3), 312–319. doi: 10.1175/1520-0450(1978)017<0312:AMCMFW>2.0.CO;2.

Sievers, U., 1995: Verallgemeinerung der Stromfunktionsmethode auf drei Dimensionen (Generalization of the streamfunction-vorticity method to three dimensions). Meteorol. Z. (3), 3–15.

Sievers, U., 2012: Das kleinskalige Strömungsmodell MUKLIMO_3. Teil 1: Theoretische Grundlagen, PC-Basisversion und Validierung 240 of Berichte des Deutschen Wetterdienstes. Selbstverlag des Deutschen Wetterdienstes. Offenbach am Main.

Singh, B.; Hansen, B. S.; Brown, M. J.; and Pardyjak, E. R., 2008: Evaluation of the QUIC-URB fast response urban wind model for a cubical building array and wide building street canyon. Environmental Fluid Mechanics 8(4), 281–312. doi: 10.1007/s10652-008-9084-5.

Staiger, H. and Matzarakis, A., 2010: Evaluation of atmospheric thermal radiation algorithms for daylight hours. Theoretical and Applied Climatology 102(1-2), 227–241. doi: 10.1007/s00704-010-0325-4.

Staiger, H.; Laschewski, G.; and Graetz, A., 2012: The perceived temperature - a versatile index for the assessment of the human thermal environment. Part A: scientific basics. International Journal of Biometeorology 56(1), 165–176. doi: 10.1007/s00484-011-0409-6.

Steyn, D., 1980: The calculation of view factors from fisheye-lens photographs. Atmos-Ocean 3 (18), 245–258.

Swinbank, W. C., 1963: Long-wave radiation from clear skies. Quarterly Journal of the Royal Meteorological Society 89(381), 339–348. doi: 10.1002/qj.49708938105.

Taylor, P. A. and Salmon, J. R., 1993: A model for the correction of surface wind data for sheltering by upwind obstacles. Journal of Applied Meteorology 49, 226–239.

Thorsson, S.; Lindberg, F.; Eliasson, I.; and Holmer, B., 2007: Different methods for estimating the mean radiant temperature in an outdoor urban setting. International Journal of Climatology 27 (14), 1983–1993. doi: 10.1002/joc.1537.

Tromp, S., 1980: Biometeorology. The impact of the weather and climate on humans and their environment (animals and plants). Heyden & Son Ltd. London, Philadelphia, Rheine.

Unger, J.; Sümeghy, Z.; Szegedi, S.; Kiss, A.; and Géczi, R., 2010: Comparison and generalisation of spatial patterns of the urban heat island based on normalized values. Bio-, Agro, and Urban Climatology 35(1–2), 107–114. doi: 10.1016/j.pce.2010.03.001.

Valko, P., 1966: Die Himmelsstrahlung in ihrer Beziehung zu verschiedenen Parametern. Archiv für Meteorologie, Geophysik und Bioklimatologie, Serie B 14(3), 336–359.

VDI, 1994: VDI Guideline 3789, Part 1: Climate. Environmental meteorology, interactions between atmosphere and surface; calculation of short-and long wave radiation. Part 2: VDI/DIN- Handbuch Reinhaltung der Luft. VDI Düsseldorf.

VDI, 2008: VDI Guideline 3787, Part 2: Environmental Meteorology. Methods for the human biometeorological evaluation of climate and air quality for urban and regional planning at regional level. Part I: Climate. VDI.

VDI-Kommission Reinhaltung der Luft, 1988: Stadtklima und Luftreinhaltung: ein wissenschaftliches Handbuch für die Praxis in der Umweltplanung; mit 47 Tabellen. Springer Berlin; Heidelberg [u.a.].

Vernom, H., 1932: The measurement of radiant temperature in relation to human comfort. Journal of Industrial Hygiene (14), 95–111.

Wang, Y. S.; Williamson, C.; Garvey, D.; Chang, S.; and Cogan, J., 2005: Application of a multigrid method to a mass-consistent diagnostic wind model. Journal of Applied Meteorology 44(7), 1078–1089. doi: 10.1175/JAM2262.1.

Watson, I. and Johnson, G., 1988: Estimating person view-factors from fish-eye lens photographs. International Journal of Biometeorology 32(2), 123–128. doi: 10.1007/BF01044905.

Wieringa, J., 1993: Representative roughness parameters for homogeneous terrain. Boundary-Layer Meteorology 63(4), 323–363. doi: 10.1007/BF00705357.

Yang, S.-Q. and Matzarakis, A., 2016: Implementation of human thermal comfort information in Köppen-Geiger climate classification—the example of China. International Journal of Biometeorology, 1–5.

Yang, X.; Zhao, L.; Bruse, M.; and Meng, Q., 2013: Evaluation of a microclimate model for predicting the thermal behavior of different ground surfaces. Building and Environment 60, 93–104. doi: 10.1016/j.buildenv.2012.11.008.

변, 재영; Kim, J.; 최, 병철; and Choi, Y.-J., 2008a: Forecast and verification of perceived temperature using a mesoscale model over the Korean Peninsula during 2007 summer. Atmosphere 18(3), 237–248.

변, 재영; 김, 정식; 김, 지영; 최, 병철; 최, 영진; and Graetz, A., 2008b: A Study on the Characteristics of Perceived Temperature over the Korean Peninsula During 2007 Summer. Atmosphere 18(2), 137–146.

이, 대근; 변, 재영; 김, 규랑; and 최, 영진., 2010: Relationship between Summer Heat Stress (Perceived Temperature) and Daily Excess Mortality in Seoul during 1991~2005. Journal of Korean Society for Atmospheric Environment 26(3), 253–264.

# List of figures

# List of tables

# List of symbols

## Symbols in equations

Symbols with their corresponding SI unit (if any).

| symbol | unit | description |
|---|---|---|
| $a$ | m | attenuation coefficient |
| $a_i$ | $^\circ$ | azimuth angle |
| $a_x, a_y$ | m | semi axis of an ellipse |
| $a_{x'}, a_{y'}$ | m | semi axis of an ellipse for projected obstacles |
| $A$ | $m^2$ | area (multiple use) |
| $A_s$ | $m^2$ | surface area |
| $cc$ | octas | cloud cover |
| $clo$ | | clothing index |
| $C$ | W | flux of sensible heat |
| $C_D$ | | drag coefficient |
| $d_l$ | m | distance of a point to the next windward building |
| $d_{nw}$ | m | distance to next wall |
| $d_{sc}$ | m | distance from the upwind building. |
| $d_{SCccw}$ | m | crosswind distance to the street canyons center |
| $d_{SCw}$ | m | street canyon width |
| $d_W$ | m | distance between building and the end of the recirculation |
| $diam_g$ | m | globe thermometer diameter |
| $D$ | $\frac{W}{m^2}$ | diffuse shortwave radiation |
| $D_{aniso}$ | $\frac{W}{m^2}$ | anisotropic part of the diffuse shortwave radiation |
| $D_{iso}$ | $\frac{W}{m^2}$ | isotropic part of the diffuse shortwave radiation |
| $D_s$ | $\frac{W}{m^2}$ | diffuse reflected shortwave radiation |
| $D_0$ | $\frac{W}{m^2}$ | initial diffuse shortwave radiation |
| $E$ | W | flux of latent heat |
| $G$ | $\frac{W}{m^2}$ | global radiation |

| | | |
|---|---|---|
| $G_0$ | $\frac{W}{m^2}$ | initial global radiation |
| $\bar{h}$ | m | average height |
| $h_{eff}$ | m | effective height |
| $h_v$ | m | volumetric height |
| $h_{LR}$ | | height scaling factor |
| $i, j, k, p$ | | indexes (multiple use) |
| $I$ | $\frac{W}{m^2}$ | direct solar radiation |
| $I_0$ | $\frac{W}{m^2}$ | initial direct solar radiation |
| $l_{po}, w_{po}$ | m | length and width of an projected obstacle |
| $l_v, h_v$ | m | current length and height of the front vortex |
| $l, wi, h$ | m | streamwise length, width and height of the obstacle |
| $L_f$ | m | length of the front eddy system |
| $L_{fv}$ | m | length of the vortex zone inside the front eddy system |
| $L_R$ | m | length of the recirculation |
| $M$ | W | metabolic heat production |
| $n$ | | number of samples or items (multiple use) |
| $N_x, N_y, N_z$ | | number of grid cells in x, y, and z direction |
| $pr$ | hPa | air pressure |
| $pr_0$ | hPa | air pressure at sea level (1013 hPa) |
| $P$ | $\frac{W}{m^2}$ | radiation flux density |
| PET | °C | physiologically equivalent temperature |
| PMV | | predicted mean vote |
| $Por$ | | porosity |
| PT | °C | perceived temperature |
| $\bar{P}_h$ | $\frac{W}{m^2}$ | mean radiation flux density of a person |
| $P_{lw}$ | $\frac{W}{m^2}$ | longwave radiation flux density |
| $P_r$ | | directional projection factor |
| $P_{sw}$ | $\frac{W}{m^2}$ | shortwave radiation flux density |
| $r_{opt}$ | | relative optical air mass |
| $R$ | W | flux of radiative heat |
| $RH$ | % | relative air humidity |
| $s$ | m | height above ground (terrain following altitude) |
| $s_i$ | m | minimum distance between two obstacles to be considered independent |
| $s_s$ | m | minimum distance between two obstacles for the air current not to flow over the gap |
| $S$ | W | heat storage |
| $T_a$ | °C | air temperature |
| $T_{clo}$ | °C | clothing temperature |

| | | |
|---|---|---|
| $T_g$ | $^\circ$C | globe temperature |
| $T_L$ | | Linke turbidity factor |
| $T_{mrt}$ | $^\circ$C | mean radiant temperature |
| $T_s$ | K | surface temperature |
| $u, v, w$ | m/s | stream components in x, y, and z direction |
| $\bar{u}(z), \bar{w}(z)$ | | vertical wind profile of the horizontal and vertical stream component |
| $u_d$ | m/s | velocity deficit on the lee side of an obstacle |
| $u_n$ | m/s | velocity normal to border |
| $u_{ref}$ | m/s | wind speed at a height of $z_{ref}$. |
| $u_{rt}, v_{rt}$ | m/s | x and y component of the wind at rooftop height |
| $u^0, v^0, w^0$ | m/s | initial stream components in x, y, and z direction |
| $u_*$ | m/s | shear velocity |
| $U(H)$ | m/s | mean wind velocity at the top of the obstacle (based on upstream power law profile) |
| UTCI | $^\circ$C | universal thermal climate index |
| $v$ | m/s | wind speed |
| $VP$ | hPa | vapour pressure |
| $W$ | m | street width |
| $WD$ | $^\circ$ | wind direction |
| $Wo$ | W | mechanical work |
| $x, y, z$ | m | Cartesian coordinates in x, y, and z direction |
| $z_d$ | m | zero plane displacement height |
| $z_{ref}$ | m | reference height, $\bar{h}$ for the ABC model. |
| $z_0$ | m | roughness length |
| $\alpha$ | | ratio of weighting factors for the consideration of atmospheric stability |
| $\alpha_{abs}$ | | shortwave absorption coefficient of a person |
| $\alpha_h$, $\alpha_v$ | s/m | horizontal $\alpha_h$ and vertical $\alpha_v$ weighting factor |
| $\beta$ | $^\circ$ | elevation angle (multiple use) |
| $\delta_{opt}$ | | optical thickness of the atmosphere |
| $\Delta x, \Delta y, \Delta z$ | m | grid size in x, y, and z direction |
| $\varepsilon$ | s $^{-1}$ | divergence |
| $\varepsilon_g$ | | globe absorption coefficient |
| $\varepsilon_{lw}$ | | longwave emission coefficient |
| $\sigma$ | $\frac{W}{m^2 \cdot T_s^4}$ | Stefan-Boltzmann constant $\left(5.67 \cdot 10^{-8} \frac{W}{m^2 \cdot T_s^4}\right)$ |
| $\Gamma$ | | $0.6 * c_a^2$ |
| $\kappa$ | | Von Kármán constant |
| $\lambda$ | s | Lagrangian multiplication factor |

| | | |
|---|---|---|
| $\lambda_f$ | | frontal area density |
| $\lambda_p$ | | building plan area density |
| $\rho$ | | source term |
| $\varphi$ | $^\circ$ | zenith angle |
| $\Psi_S$ | | skyview factor |
| $\omega_{planar}$ | | correction factor to calculate planar SVF |
| $\omega_{proj}$ | | correction factor to calculate spheric SVF |

# 10 Appendix

## 10.1 Source code

### 10.1.1 Windward stagnation zone

```cpp
// Copyright 2015, Albert-Ludwigs University Freiburg
// Chair for Meteorology and Climatology
// Author: Dominik Froehlich <dominik.froehlich@meteo.uni-freiburg.de>

////////// Windwards stagnation zone ////////

#define _USE_MATH_DEFINES
#include <cmath>
#include <vector>
// #include <iostream>
#include "./Tools.h"

using std::sin;
using std::abs;
using std::pow;
using std::sqrt;

// ---------------------------------------------------------------------------
void Wind::FrontEddy() {
  // Create new temporary grids
  if (_u_FE != NULL) delete _u_FE;
  if (_v_FE != NULL) delete _v_FE;
  if (_w_FE != NULL) delete _w_FE;  // causes invalid pointer. Find out why!
  _u_FE = new ThreeDMatrix<float>(_nz, _ny, _nx+1, 9999.0);
  _v_FE = new ThreeDMatrix<float>(_nz, _ny+1, _nx, 9999.0);
  _w_FE = new ThreeDMatrix<float>(_nz+1, _ny, _nx, 9999.0);

  for (int b = 0; b < _bnum; b++) {  // calculate front eddy for each obst
    float solid = _buildings[b][7];
    float startx = _buildings[b][1] * _dx;
    float starty = _buildings[b][2] * _dy;
    float h = _buildings[b][6];
    // calculate streamwise horizontal extension of the front eddy zone after Bagal et al.
       (2004)
    float Lfx = ((1.5 * (_buildings[b][5] / h)) / (1.0 + 0.8 *
      (_buildings[b][5] / h))) * h * solid;
    // calculate streamwise horizontal extension of the front eddy zone
    float Lfy = ((1.5 * (_buildings[b][4] / h)) / (1.0 + 0.8 *
      (_buildings[b][4] / h))) * h * solid;
    if (Lfx < (0.5 * _dx) && Lfy < (0.5 * _dy)) continue;  // skip if too small
    // calculate vertical vortex extension after Bagal et al. (2004)
    float Lfzx = ((0.6 * (_buildings[b][5] / h)) / (1.0 + 0.8 *
      (_buildings[b][5] / h))) * h * solid;
    float Lfzy = ((0.6 * (_buildings[b][4] / h)) / (1.0 + 0.8 *
      (_buildings[b][4] / h))) * h * solid;
    int foot = _buildings[b][3];  // obstacles foot
    int top = round(0.6 * h / _dz) + foot;  // obstacles stagnation point
    // int stag = round((top - foot) / _dz);  // level of the stagnation point (in cells)
    if (top <= foot) continue;

    // Length of the half-axis in
    float* axx = new float[top-foot];  // x-dir  // Stagnation zone
    float* ayy = new float[top-foot];  // y-dir
    float* axxV = new float[top-foot];  // x-dir  // Vortex
    float* ayyV = new float[top-foot];  // y-dir
    float axy = (_buildings[b][5] + _dy) / 2.0;  // Second half axis is the same for both
    float ayx = (_buildings[b][4] + _dx) / 2.0;
    // Calculate semi axis of the ellyptic front eddy zone
    for (int z = 0; z < top-foot; z++) {
      axx[z] = Lfx * pow(sin(_WD * (M_PI / 180.0)), 2.0) *  // Stagnation
        // sqrt(abs(1.0 - pow(z / (0.6 * h), 2)));
        sqrt(pow(1.0 - z / (0.6 * h), 2.0));
      ayy[z] = Lfy * pow(cos(_WD * (M_PI / 180.0)), 2.0) *
        // sqrt(abs(1.0 - pow(z / (0.6 * h), 2)));
        sqrt(pow(1.0 - z / (0.6 * h), 2.0));
      axxV[z] = Lfzx * pow(sin(_WD * (M_PI / 180.0)), 2.0) *  // Vortex
        sqrt(pow(1.0 - z / (0.6 * h), 2.0));
      ayyV[z] = Lfzy * pow(cos(_WD * (M_PI / 180.0)), 2.0) *
        sqrt(pow(1.0 - z / (0.6 * h), 2.0));
    }

    // x - direction from here //
    float w = 0.0;
    float l = 0.0;
    float centerx = 0.0;
    float centery = 0.0;
    int xrange_from = 0;
    int xrange_to = 0;
    int yrange_from = 0;
    int yrange_to = 0;
    int xdir = 0;
    bool calcXdir = false;

    if (_WD > 0.0 && _WD < 180.0) {  // incident wind from +x dir
      w = _buildings[b][5];  // read out building width (m)
      l = _buildings[b][4];  // resp. length
      centerx = startx + l - _dx;  // calculate center of the ellipse
      centery = starty + 0.5 * (w - _dy);  // resp
      xrange_from = round(centerx / _dx);
      xrange_to = round((centerx + ceil(axx[0])) / _dx);
      yrange_from = round(starty / _dy);
      yrange_to = round((starty + w) / _dy) - 1;
      xdir = 1;
      calcXdir = true;
    }

    if (_WD > 180.0 && _WD < 360.0) {  // incident wind from -x dir
      w = _buildings[b][5];  // read out building width
      centerx = startx;  // calculate center of the ellipse
      centery = starty + 0.5 * (w - _dy);
      xrange_from = round((startx - ceil(axx[0])) / _dx);
      xrange_to = round(startx / _dx);
      yrange_from = round(starty / _dy);
      yrange_to = round((starty + w) / _dy) - 1;
      xdir = -1;
      calcXdir = true;
    }

    if (calcXdir) {
      // std::cout << "DEBUG: Obst. " << b << " x-dir" << std::endl;
      if (xrange_from < 0) xrange_from = 0;  // Delete Items out of range!
      if (xrange_to >= _nx) xrange_to = _nx - 1;
      if (yrange_from < 0) yrange_from = 0;
      if (yrange_to >= _ny) yrange_to = _ny - 1;
      float ay2 = pow(axy, 2.0);
      // for all points inside the z-extension of the front eddy zone,
      for (int z = foot; z < top; z++) {
        float ax2 = pow(axx[z-foot], 2.0);  // Stagnation
        float ax2V = pow(axxV[z-foot], 2.0);  // Vortex
        if (ax2 != 0.0 && ay2 != 0.0) {
          // front eddy in x-dir
          for (int y = yrange_from; y <= yrange_to; y++) {  // the y extension of the front
              eddy zone
            for (int x = xrange_from; x <= xrange_to; x++) {  // and the x- extention of the
                 front eddy zone,
              // that are inside the front eddy ellipse, and inside the calculation grid,
              if (((pow((x * _dx) - centerx, 2.0) / ax2) + (pow((y * _dy) -
                  centery, 2.0) / ay2)) <= 1.0) {  // inside Stagnation?
                float mod = 9999.0;
                // Vortex is smaller and therefore can only be inside the stagnation area
                if (((pow((x * _dx) - centerx, 2.0) / ax2V) + (pow((y * _dy) -
                    centery, 2.0) / ay2)) <= 1.0) {  // inside Vortex?
                  float vorltexlength = sqrt(abs(ax2V - (pow((y * _dy) -
                    centery, 2.0) / ay2) * ax2V)) + centerx;
                  float vortexheight = sqrt(abs(pow(0.6 * _buildings[b][6],
                    2.0) - (pow((x * _dx) - centerx, 2.0) / pow(vorltexlength,
                    2.0)) * pow(0.6 * _buildings[b][6], 2.0)));
                  // vortex horizontal
                  float vortex = ((0.6 * cos((M_PI * vortexheight) / (0.5 *
                    _buildings[b][6])) + 0.05) * (-0.6 * sin((M_PI *
                    vorltexlength) / Lfzx)));
                  mod = _ugrid->p(top, y, x) * vortex * xdir;
                  if ((abs(mod) > abs(_u_FE->p(z, y, x))) ||
                      (_u_FE->p(z, y, x) == 9999.0)) {  // write vortex if stronger than
                          present influence
                    _u_FE->p(z, y, x) = mod;
                  }
                  mod = _ugrid->p(top, y, x+1) * vortex * xdir;
                  if ((abs(mod) > abs(_u_FE->p(z, y, x+1))) ||
                      (_u_FE->p(z, y, x+1) == 9999.0)) {
                    _u_FE->p(z, y, x+1) = mod;
                  }
                  // vortex vertical
                  float wall_dist = abs(centerx - (x * _dx));
                  mod = abs(_ugrid->p(top, y, x)) * (-0.1 * cos((M_PI *
                    wall_dist) / vorltexlength) - 0.05);
                  if ((abs(mod) > abs(_wgrid->p(z+1, y, x))) ||
                      (_w_FE->p(z+1, y, x) == 9999.0)) {
                    _w_FE->p(z+1, y, x) = mod;
                  }
                } else {  // outside Vortex, so write stagnation
                  float stagnation = 0.4 * ((z * _dx) / _buildings[b][6]);
                  mod = _ugrid->p(z, y, x) * stagnation;
                  if (abs(mod) < abs(_u_FE->p(z, y, x))) {  // write stagnation if weaker
                      than present influence
                    _u_FE->p(z, y, x) = mod;
                  }
                  mod = _ugrid->p(z, y, x+1) * stagnation;
                  if (abs(mod) < abs(_u_FE->p(z, y, x+1))) {
                    _u_FE->p(z, y, x+1) = mod;
                  }
                }
              }
            }
          }
        }
      }
    }  // end of front eddy in +x direction

    // y - direction from here //
    w = 0.0;  // reset variables for safety only
    l = 0.0;
    centerx = 0.0;
    centery = 0.0;
    xrange_from = 0;
    xrange_to = 0;
    yrange_from = 0;
    yrange_to = 0;
    int ydir = 0;
    bool calcYdir = false;

    if (_WD > 90.0 && _WD < 270.0) {  // incident wind from -y dir
      w = _buildings[b][4];  // read out building width
      centerx = startx + 0.5 * (w - _dx);  // calculate center of the ellipse
      centery = starty;
      xrange_from = round(startx / _dx);
      xrange_to = round((startx + w) / _dx) - 1;
      yrange_from = round((starty - ceil(ayy[0])) / _dy);
      yrange_to = round(starty / _dy);
      ydir = 1;
      calcYdir = true;
    }

    if (_WD > 270.0 || (_WD > 0.0 && _WD < 90.0)) {  // incident wind from +y dir
      w = _buildings[b][4];  // read out building width
      l = _buildings[b][5];  // resp. length
      centerx = startx + 0.5 * (w - _dx);  // calculate center of the ellipse
      centery = starty + l - _dy;
      xrange_from = round(startx / _dx);
      xrange_to = round((startx + w) / _dx) -1;
      yrange_from = round(centery / _dy);
      yrange_to = round((starty + l + ceil(ayy[0])) / _dy);
      ydir = -1;
      calcYdir = true;
    }
    if (calcYdir) {
      // std::cout << "DEBUG: Obst. " << b << " y-dir" << std::endl;
      if (xrange_from < 0) xrange_from = 0;  // Delete Items out of range!
      if (xrange_to >= _nx) xrange_to = _nx - 1;
      if (yrange_from < 0) yrange_from = 0;
      if (yrange_to >= _ny) yrange_to = _ny - 1;
      float ax2 = pow(ayx, 2.0);
      // for all points inside the z-extension of the front eddy zone,
      for (int z = foot; z < top; z++) {
        // front eddy in y-dir
        float ay2 = pow(ayy[z-foot], 2.0);  // Stagnation
        float ay2V = pow(ayyV[z-foot], 2.0);  // Vortex
        if (ax2 != 0.0 && ay2 != 0.0) {
          for (int y = yrange_from; y <= yrange_to; y++) {  // the y extension of the front
              eddy zone
            float wall_dist = abs(centery - (y * _dy));
            for (int x = xrange_from; x <= xrange_to; x++) {  // and the x- extention of the
                 front eddy zone,
              // that are inside the front eddy ellipse, and inside the calculation grid,
              if (((pow((y * _dy) - centery, 2.0) / ay2) + (pow((x * _dx) -
                  centerx, 2.0) / ax2)) <= 1.0) {
                float mod = 9999.0;
                // Vortex is smaller and therefore can only be inside the stagnation area
                if (((pow((y * _dy) - centery, 2.0) / ay2V) + (pow((x * _dx) -
                     centerx, 2.0) / ax2)) <= 1.0) {  // inside Vortex?
                  float vorltexlength = sqrt(abs(ay2V - pow((x * _dx) - centerx,
                    2.0) / ax2 * ay2V)) + centery;  // length of the vortex at given x and
                        height (m)
                  float vortexheight = sqrt(abs(pow(0.6 * _buildings[b][6],
                    2.0) - (pow((y * _dy) - centery, 2.0) / pow(
                    vorltexlength, 2.0)) * pow(0.6 * _buildings[b][6], 2.0)));
                  // vortex horizontal
                  float vortex = ((0.6 * cos((M_PI * vortexheight) / (0.5 *
                    _buildings[b][6])) + 0.05) *  (-0.6 * sin((M_PI *
                    vorltexlength) / Lfzy)));
                  mod = _vgrid->p(top, y, x) * vortex * ydir;
                  if ((abs(mod) > abs(_v_FE->p(z, y, x))) ||
                      (_v_FE->p(z, y, x) == 9999.0)) {  // write vortex if stronger than
                          present influence
                    _v_FE->p(z, y, x) = mod;
                  }
                  mod = _vgrid->p(top, y+1, x) * vortex * ydir;
                  if ((abs(mod) > abs(_v_FE->p(z, y+1, x))) ||
                      (_v_FE->p(z, y+1, x) == 9999.0)) {
                    _v_FE->p(z, y+1, x) = mod;
                  }
                  // vortex vertical
                  // float wall_dist = abs(centery - (y * _dy));  // moved upwards for
                      increased performance
                  // mod = abs(_vgrid->p(top, y, x)) * (-0.1 * cos((M_PI * wall_dist) / Lfzy
                      ) - 0.05);
                  mod = abs(_vgrid->p(top, y, x)) * (-0.1 * cos((M_PI *
                    wall_dist) / vorltexlength) - 0.05);
                  if ((abs(mod) > abs(_wgrid->p(z+1, y, x))) ||
                      (_w_FE->p(z+1, y, x) == 9999.0)) {
                    _w_FE->p(z+1, y, x) = mod;
                  }
                } else {  // outside Vortex, so write stagnation
                  float stagnation = 0.4 * ((z * _dy) / _buildings[b][6]);
                  mod = _vgrid->p(z, y, x) * stagnation;
                  if (abs(mod) < abs(_v_FE->p(z, y, x))) {  // write stagnation if weaker
                      than present influence
                    _v_FE->p(z, y, x) = mod;
                  }
                  mod = _vgrid->p(z, y+1, x) * stagnation;
                  if (abs(mod) < abs(_v_FE->p(z, y+1, x))) {
                    _v_FE->p(z, y+1, x) = mod;
                  }
                }
              }
            }
          }
        }
      }
    }
    // clear stack
    delete[] axx;
    delete[] ayy;
    delete[] axxV;
    delete[] ayyV;
  }
  // return nothing as function is void now
}
```

### 10.1.2 Lee-side recirculation

```cpp
// Copyright 2016, Albert-Ludwigs University Freiburg
// Chair for Meteorology and Climatology
// Author: Dominik Froehlich <dominik.froehlich@venus.uni-freiburg.de>

// calculates the modification of the recirculation, modified after Roeckle 1990

// requires: WD, dx, dy, ugrid, vgrid, buildings, nx, ny, nz
// u_CW = array(0, dim=c(nx+1,ny,nz)) // velocity grid in x dir (with zeros)
// v_CW = array(0, dim=c(nx,ny+1,nz)) // velocity grid in x dir (with zeros)

#include <stdlib.h>
#include <cmath>
#include <algorithm>
#include <vector>
// #include <iostream>  // DEBUG, remove!
#include "./Tools.h"
#include "./max4.h"

using std::abs;
using std::pow;
using std::sqrt;
using std::min;
using std::max;
using std::min_element;
using std::max_element;

void Wind::CloseWake() {
  // Create new temporary grids
  if (_u_CW != NULL) delete _u_CW;
  if (_v_CW != NULL) delete _v_CW;
  _u_CW = new ThreeDMatrix<float>(_nz, _ny, _nx+1, 999.0);
  _v_CW = new ThreeDMatrix<float>(_nz, _ny+1, _nx, 999.0);
  /** // make sure grid is empty
  for (int z = 0; z < _nz; z++) {
    for (int y = 0; y < _ny; y++) {
      for (int x = 0; x <= _nx; x++) {
        _u_CW->p(z, y, x) = 999.0;
      }
    }
  }
  // make sure grid is empty
  for (int z = 0; z < _nz; z++) {
    for (int y = 0; y <= _ny; y++) {
      for (int x = 0; x < _nx; x++) {
        _v_CW->p(z, y, x) = 999.0;
      }
    }
  } **/

  // float minLR = 1000.0;                    // DEBUG, remove!
  // float maxLR = 0.0;
  // float minLH = 1000.0;
  // float maxLH = 0.0;

  // for perpendicular incident flow
  if (_WD == 90.0 ||  _WD == 180.0 || _WD == 270.0 ||  _WD == 360.0) {
    for (int b = 0; b < _bnum; b++) {  // calculate for each building
      float solid = _buildings[b][7];
      float bf = _buildings[b][3] * _dz;  // read out foot hight of the building (metric)
      float h = _buildings[b][6];  // read out building hight (metric)
      int foot = _buildings[b][3];  // cells
      int top = round((bf + h - _dz) / _dz);  // cells
      float startx = _buildings[b][1] * _dx;  // lower left corner of the building
      float starty = _buildings[b][2] * _dy;
      if (_WD == 90.0 || _WD == 270.0) {  // for perpendicular incident flow in x dir
        float w = _buildings[b][5];  // read out building width
        float l = _buildings[b][4];  // lenght resp.
        float LH = l / h;   // check if l / h in valid range of 0.3 to 3.0
        // if (LH < minLH) { minLH = LH; }                  // DEBUG, remove!
        // if (LH > maxLH) { maxLH = LH; }
        if (LH < 0.3) { LH = 0.3; }
        if (LH > 3.0) { LH = 3.0; }
        // float h_LR = 20.0 * (1.0 - exp(-0.02 * h));  // apply function for limited growth
             for building height, border is 20.0m, dom: 21.10.2014
        float h_LR = 5.0 * (1.0 - exp(-0.2 * h));  // limit height
        float LR = ((1.8 * (w / h_LR)) / (pow(l/h_LR, 0.3) * (1.0 + 0.24 *
          (w / h_LR)))) * h_LR * solid;
        // if (minLR > LR) { minLR = LR; }                  // DEBUG, remove!
        // if (maxLR < LR) { maxLR = LR; }
        _buildings[b][8] = LR;  // add LR to building parameters, 15.08.2014
        if (LR < (0.5 * _dx) && LR < (0.5 * _dy)) continue;  // Skip if too short
        double* ax = new double[top - foot + 1];
        double* ay = new double[top - foot + 1];
        for (int z = 0; z <= top-foot; z++) {
          ax[z] = LR * sqrt(1.0 - pow((foot+z)*_dz / h, 2.0)) - (l / 2.0);  // after
              Pardyjak et al. (2004), dom: 15.09.2014
          if (ax[z] < 0.0) {ax[z] = 0.0; }
          ay[z] = w / 2.0;
        }
        if (_WD == 90.0) {               // for perpendicular flow from +x dir
          // calculate center of the ellipse (x)
          float centerx = startx - 0.5 * _dx;
          float centery = starty - 0.5 * _dy + 0.5 * w;  // (y)
          int xrange_from = round(startx) - ceil(ax[0]);
          int xrange_to = round(startx);
          int yrange_from = round(starty);
          int yrange_to = round(starty + w);
          if (xrange_from < 0) xrange_from = 0;  // Delete Items out of range!
          if (xrange_to >= _nx) xrange_to = _nx - 1;
          if (yrange_from < 0) yrange_from = 0;
          if (yrange_to >= _ny) yrange_to = _ny - 1;

          // for all points inside the z-extension of the recirculation zone,
          for (int z = foot; z <= top; z++) {
            float ax2 = pow(ax[z-foot], 2.0);
            float ay2 = pow(ay[z-foot], 2.0);
            for (int y = yrange_from; y <= yrange_to; y++) {  // the y extension
              // and the x- extention of the close wake
              for (int x = xrange_from; x <= xrange_to; x++) {
                // that are inside the front eddy ellipse,
                // and inside the calculation grid,
                if (((pow(x - centerx, 2.0) / ax2) + (pow(y - centery, 2.0)
                    / ay2)) <= 1.0) {
                  float dl = abs(x - centerx);  // points distance to the wall
                  // walls distance to the end of the wake zone
                  float dw = sqrt(abs(ax2 - pow(y - centery, 2.0) / ay2 * ax2));
                  if (dl > 0.0 && dw > 0.0) {
                    // calculate modifications for this cell
                    float mod = -1.0 * _ugrid->p(top, y, x) * pow(1.0 - dl/dw,
                      2.0);
                    if (_u_CW->p(z, y, x) == 999.0 || abs(_u_CW->p(z, y, x)) <
                      abs(mod)) {
                      _u_CW->p(z, y, x) = mod;
                    }
                  }
                }
              }
            }
            }
          }
        }
        if (_WD == 270.0) {         // for perpendicular incident flow from -x dir
          // calculate center of the ellipse (x)
          float centerx = startx * _dx + l - _dx;  // startx - 0.5 * _dx + l;
          float centery = starty - 0.5 * _dy + 0.5 * w;  // (y)
          int xrange_from = round(centerx);  // round(startx + l - 1);
          int xrange_to = round(centerx) + ceil(ax[0]);
          int yrange_from = round(starty);
          int yrange_to = round(starty + w);
          if (xrange_from < 0) xrange_from = 0;  // Delete Items out of range!
          if (xrange_to >= _nx) xrange_to = _nx - 1;
          if (yrange_from < 0) yrange_from = 0;
          if (yrange_to >= _ny) yrange_to = _ny - 1;

          // for all points inside the z-extension of the recirculation zone,
          for (int z = foot; z <= top; z++) {
            float ax2 = pow(ax[z-foot], 2.0);
            float ay2 = pow(ay[z-foot], 2.0);
            for (int y = yrange_from; y <= yrange_to; y++) {  // the y extension
              // and the x- extention of the close wake
              for (int x = xrange_from; x <= xrange_to; x++) {
                // that are inside the front eddy ellipse, and inside the
                // calculation grid,
                if (((pow(x - centerx, 2.0) / ax2) + (pow(y - centery, 2.0) /
                    ay2)) <= 1.0) {
                  // dl = abs(x - startx)   // points distance to the wall
                  float dl = abs(x - centerx);    // points distance to the wall
                  // walls distance to the end of the wake zone
                  float dw = sqrt(abs(ax2 - pow(y-centery, 2.0) / ay2 * ax2));
                  if (dl > 0.0 && dw > 0.0) {
                    // calculate modifications for this cell
                    float mod = -1.0 * _ugrid->p(top, y, x+1) * pow(1.0 - dl/dw,
                      2.0);
                    if (_u_CW->p(z, y, x+1) == 999.0 || abs(_u_CW->p(z, y,
                      x+1)) < abs(mod)) {
                      _u_CW->p(z, y, x+1) = mod;
                    }
                  }
                }
              }
            }
          }
        }
        delete ax;
        delete ay;
      } else {                    // for perpendicular incident flow in y dir
        float w = _buildings[b][4];  // read out building width
        float l = _buildings[b][5];
        float LH = l / h;
        // if (LH < minLH) { minLH = LH; }                   // DEBUG, remove!
        // if (LH > maxLH) { maxLH = LH; }
        if (LH < 0.3) { LH = 0.3; }
        if (LH > 3.0) { LH = 3.0; }
        // float h_LR = 20.0 * (1.0 - exp(-0.02 * h));  // apply function for limited growth
             for building height, border is 20.0m, dom: 21.10.2014
        float h_LR = 5.0 * (1.0 - exp(-0.2 * h));  // limit height
        float LR = ((1.8 * (w / h_LR)) / (pow(l/h_LR, 0.3) * (1.0 + 0.24 *
          (w / h_LR)))) * h_LR * solid;
        // if (minLR > LR) { minLR = LR; }                   // DEBUG, remove!
        // if (maxLR < LR) { maxLR = LR; }
        _buildings[b][8] = LR;  // add LR to building parameters, 15.08.2014
        if (LR < 0.5 * _dx && LR < 0.5 * _dy) continue;  // Skip if too short
        double* ax = new double[top - foot + 1];
        double* ay = new double[top - foot + 1];
        for (int z = 0; z <= top-foot; z++) {
          ax[z] = w / 2.0;
          ay[z] = LR * sqrt(abs(1.0 - pow((foot+z)*_dz / h, 2.0))) - (1 / 2.0);  // after
              Pardyjak et al. (2004), dom: 15.09.2014
          if (ay[z] < 0.0) {ay[z] = 0.0; }
        }

        if (_WD == 360.0) {                // for perpendicular flow from +y dir
          // calculate center of the ellipse (x)
          float centerx = startx - 0.5 * _dx + 0.5 * w;
          float centery = starty - 0.5 * _dy;  // (y)
          int xrange_from = round(startx);
          int xrange_to = round(startx + w - 1);
          int yrange_from = round(starty) - ceil(ay[0]);
          int yrange_to = round(starty);
          if (xrange_from < 0) xrange_from = 0;  // Delete Items out of range!
          if (xrange_to >= _nx) xrange_to = _nx - 1;
          if (yrange_from < 0) yrange_from = 0;
          if (yrange_to >= _ny) yrange_to = _ny - 1;
          // for all points inside the z-extension of the recirculation zone
          for (int z = foot; z <= top; z++) {
            float ax2 = pow(ax[z-foot], 2.0);
            float ay2 = pow(ay[z-foot], 2.0);
            for (int y = yrange_from; y <= yrange_to; y++) {  // the y extension
              // and the x- extention of the close wake
              for (int x = xrange_from; x <= xrange_to; x++) {
                // that are inside the front eddy ellipse, and inside the
                // calculation grid,
                if (((pow(x-centerx, 2.0) / ax2) + (pow(y-centery, 2.0) /
                    ay2)) <= 1.0) {
                  float dl = abs(y - centery);   // points distance to the wall
                  // points distance to the end of the wake zone
                  // (+dl as at wrong side of ellipse)
                  float dw = sqrt(abs(ay2 - pow(x-centerx, 2.0) / ax2 * ay2));
                  if (dl > 0.0 && dw > 0.0) {
                    // calculate modifications for this cell
                    float mod = -1.0 * _vgrid->p(top, y+1, x) * pow(1.0 - dl/dw,
                      2.0);
                    if (_v_CW->p(z, y+1, x) == 999.0 || abs(_v_CW->p(z, y+1,
                      x)) < abs(mod)) {
                      _v_CW->p(z, y+1, x) = mod;
                    }
                  }
                }
              }
            }
          }
        }

        if (_WD == 180.0) {                // for perpendicular flow from +y dir
          // calculate center of the ellipse (x)
          float centerx = startx - 0.5 * _dx + 0.5 * w;
          float centery = starty - 0.5 * _dy + 1;  // (y)
          int xrange_from = round(startx);
          int xrange_to = round(startx + w - 1);
          int yrange_from = round(starty - 0.5 * _dy + 1);
          int yrange_to = round(starty - 0.5 * _dy + 1) + ceil(ay[0]);
          if (xrange_from < 0) xrange_from = 0;  // Delete Items out of range!
          if (xrange_to >= _nx) xrange_to = _nx - 1;
          if (yrange_from < 0) yrange_from = 0;
          if (yrange_to >= _ny) yrange_to = _ny - 1;
          // for all points inside the z-extension of the recirculation zone,
          for (int z = foot; z <= top; z++) {
            float ax2 = pow(ax[z-foot], 2.0);
            float ay2 = pow(ay[z-foot], 2.0);
            for (int y = yrange_from; y <= yrange_to; y++) {  // the y extension
              // and the x- extention of the close wake
              for (int x = xrange_from; x <= xrange_to; x++) {
                // that are inside the front eddy ellipse, and inside the
                // calculation grid,
                if (((pow(x-centerx, 2.0) / ax2) + (pow(y-centery, 2.0) /
                    ay2)) <= 1.0) {
                  float dl = abs(y - centery);   // points distance to the wall
                  float dw = sqrt(abs(ay2 - pow(x-centerx, 2.0) / ax2 * ay2));
                  if (dl > 0.0 && dw > 0.0) {
                    // calculate modifications for this cell
                    float mod = -1.0 * _vgrid->p(top, y, x) * pow(1.0 - dl/dw,
                      2.0);
                    if (_v_CW->p(z, y, x) == 999.0 || abs(_v_CW->p(z, y, x))
                      < abs(mod)) {
                      _v_CW->p(z, y, x) = mod;
                    }
                  }
                }
              }
            }
          }
        }
        delete ax;
        delete ay;
      }
    }
    // no return as function is void now
  } else {  ///////////////////////////////// for non-perpendicular incident flow
      /////////////////////////
    // DEG:sin(12)=0.20791169, RAD:sin(12)=-0.5365729, GRA:sin(12)=0.187381314
    float WD1 = (-90.0 - _WD) * (M_PI / 180.0);  // to achieve incident flow from -x (-90.0-
        _WD)
    float rotmat[2][2];  // create rotation matrix
    rotmat[0][0] = cos(WD1);
    rotmat[0][1] = sin(WD1);
    rotmat[1][0] = -1.0 * sin(WD1);
    rotmat[1][1] = cos(WD1);

    // factor to reduce the length of the recirculation in case of diagonal incident flow.
        dom: 2014-11-13
    double elipse_extension_factor = max(abs(rotmat[0][1]), abs(rotmat[0][0]));  // max(abs(
        sin(WD1)), abs(cos(WD1)));

    for (int b = 0; b < _bnum; b++) {  // calculate recirculation for each building
      float solid = _buildings[b][7];
      // project building to achieve perpendicular flow
      double p_low_left_x = (_buildings[b][1] - 0.5) * _dx;  // lower left corner (in
          cartesian coord. system)
      double p_low_left_y = (_buildings[b][2] - 0.5) * _dy;
      double p_up_left_x = (_buildings[b][1] - 0.5) * _dx;  // upper left corner
      double p_up_left_y = (_buildings[b][2] * _dy) + _buildings[b][5] - _dy;
      double p_low_right_x = (_buildings[b][1] * _dx) + _buildings[b][4] - _dx;  // lower
          right corner
      double p_low_right_y = (_buildings[b][2] - 0.5) * _dy;
      double p_up_right_x = (_buildings[b][1] * _dx) + _buildings[b][4] - _dx;  // upper
          right corner
      double p_up_right_y = (_buildings[b][2] * _dy) + _buildings[b][5] - _dy;
      // I have to do matrix multiplication manually??? Come on...
      double pr_low_left_x = rotmat[0][0] * p_low_left_x + rotmat[0][1] *
        p_low_left_y;
      double pr_low_left_y = rotmat[1][1] * p_low_left_y + rotmat[1][0] *
        p_low_left_x;
      double pr_up_left_x = rotmat[0][0] * p_up_left_x + rotmat[0][1] *
```

```
        p_up_left_y;
      double pr_up_left_y = rotmat[1][1] * p_up_left_y + rotmat[1][0] *
        p_up_left_x;
      double pr_low_right_x = rotmat[0][0] * p_low_right_x + rotmat[0][1] *
        p_low_right_y;
      double pr_low_right_y = rotmat[1][1] * p_low_right_y + rotmat[1][0] *
        p_low_right_x;
      double pr_up_right_x = rotmat[0][0] * p_up_right_x + rotmat[0][1] *
        p_up_right_y;
      double pr_up_right_y = rotmat[1][1] * p_up_right_y + rotmat[1][0] *
        p_up_right_x;

      double A = _buildings[b][4] * _buildings[b][5];  // Area covered by the building in
          square meters
      double h = _buildings[b][6];  // hight of the building in meters
      double bf = _buildings[b][3] * _dz;  // read out foot hight of the building
      double w_proj = max4(pr_low_left_y, pr_up_left_y, pr_low_right_y,
        pr_up_right_y) - min4(pr_low_left_y, pr_up_left_y, pr_low_right_y,
        pr_up_right_y);
      double l_proj = A / w_proj;

      int foot = round(bf);  // zrange = bf:(bf + h-1)
      int top = round(bf + h - _dz);

      float LH = l_proj / h;
      if (LH < 0.3) { LH = 0.3; }
      if (LH > 3.0) { LH = 3.0; }
      // float h_LR = 20.0 * (1.0 - exp(-0.02 * h));  // apply function for limited growth
          for building height, border is 20.0m, dom: 21.10.2014
      float h_LR = 5.0 * (1.0 - exp(-0.2 * h));  // limit height
      double LR = ((1.8 * (w_proj/h_LR)) / (pow(l/h_LR, 0.3) * (1.0 + 0.24 *
        (w_proj / h_LR)))) * h_LR * solid;
      _buildings[b][8] = LR;  // add LR to building parameters, 15.08.2014
      if (LR < (0.5 * _dx) && LR < (0.5 * _dy)) continue;  // Skip if too short

      // replacement for for-loop above, preparation
      double max_x = max4(pr_low_left_x, pr_up_left_x, pr_low_right_x,
        pr_up_right_x);
      double max_y = max4(pr_low_left_y, pr_up_left_y, pr_low_right_y,
        pr_up_right_y);
      double min_y = min4(pr_low_left_y, pr_up_left_y, pr_low_right_y,
        pr_up_right_y);

      // First part of former loop (set middlepoint, located on ellipse's ay-line)
      double middlep_y;
      double middlep_rx;  // r = rotated!
      double middlep_ry;
      if (abs(pr_low_left_x - max_x) <= 0.01) {
        middlep_y = p_low_left_y;
        middlep_rx = pr_low_left_x;
        middlep_ry = pr_low_left_y;
      }
      if (abs(pr_up_left_x - max_x) <= 0.01) {
        middlep_y = p_up_left_y;
        middlep_rx = pr_up_left_x;
        middlep_ry = pr_up_left_y;
      }
      if (abs(pr_low_right_x - max_x) <= 0.01) {
        middlep_y = p_low_right_y;
        middlep_rx = pr_low_right_x;
        middlep_ry = pr_low_right_y;
      }
      if (abs(pr_up_right_x - max_x) <= 0.01) {
        middlep_y = p_up_right_y;
        middlep_rx = pr_up_right_x;
        middlep_ry = pr_up_right_y;
      }

      // Second part of the former loop (set upperp, point with highest y value)
      double upperp_x;
      double upperp_rx;
      double upperp_ry;
      if (abs(pr_low_left_y - max_y) <= 0.01) {
        upperp_x = p_low_left_x;
        upperp_rx = pr_low_left_x;
        upperp_ry = pr_low_left_y;
      }
      if (abs(pr_up_left_y - max_y) <= 0.01) {
        upperp_x = p_up_left_x;
        upperp_rx = pr_up_left_x;
        upperp_ry = pr_up_left_y;
      }
      if (abs(pr_low_right_y - max_y) <= 0.01) {
        upperp_x = p_low_right_x;
        upperp_rx = pr_low_right_x;
        upperp_ry = pr_low_right_y;
      }
      if (abs(pr_up_right_y - max_y) <= 0.01) {
        upperp_x = p_up_right_x;
        upperp_rx = pr_up_right_x;
        upperp_ry = pr_up_right_y;
      }

      // Third part of the former loop (set lowerp, point with lowest y value)
      double lowerp_x;
      double lowerp_rx;
      double lowerp_ry;
      if (abs(pr_low_left_y - min_y) <= 0.01) {
        lowerp_x = p_low_left_x;
        lowerp_rx = pr_low_left_x;
        lowerp_ry = pr_low_left_y;
      }
      if (abs(pr_up_left_y - min_y) <= 0.01) {
        lowerp_x = p_up_left_x;
        lowerp_rx = pr_up_left_x;
        lowerp_ry = pr_up_left_y;
      }
      if (abs(pr_low_right_y - min_y) <= 0.01) {
        lowerp_x = p_low_right_x;
        lowerp_rx = pr_low_right_x;
        lowerp_ry = pr_low_right_y;
      }
      if (abs(pr_up_right_y - min_y) <= 0.01) {
        lowerp_x = p_up_right_x;
        lowerp_rx = pr_up_right_x;
        lowerp_ry = pr_up_right_y;
      }

      // upper quarter-ellipse
      double* ax1 = new double[top - foot + 1];
      // double* ay1 = new double[top - foot + 1];
      double ay1 = (upperp_ry - middlep_ry);  // constant, no array needed! dom: 2014-11-14
      // lower quarter-ellipse
      double* ax2 = new double[top - foot + 1];
      // double* ay2 = new double[top - foot + 1];
      double ay2 = (middlep_ry - lowerp_ry);

      // calculate shortened LR, dom: 2014-11-13
      double upper_LR = 0.0;
      double lower_LR = 0.0;
      if (upperp_rx >= lowerp_rx) {
        upper_LR = LR * elipse_extension_factor;
        lower_LR = (LR * elipse_extension_factor) + upperp_rx - lowerp_rx;
      } else {
        upper_LR = (LR * elipse_extension_factor) + lowerp_rx - upperp_rx;
        lower_LR = LR * elipse_extension_factor;
      }
      // Calculate half-axis
      for (int z = 0; z <= top-foot; z++) {
        ax1[z] = upper_LR * sqrt(1.0 - pow((foot + z) * _dz /
          h, 2.0)) - (l_proj / 2.0);  // after Pardyjak et al. (2004), dom: 15.09.2014
        if (ax1[z] < 0.0) { ax1[z] = 0.0; }
        ax2[z] = lower_LR * sqrt(1.0 - pow((foot + z) * _dz /
          h, 2.0)) - (l_proj / 2.0);  // after Pardyjak et al. (2004), dom: 15.09.2014
        if (ax2[z] < 0.0) { ax2[z] = 0.0; }
      }

      // calculate center of the ellipse(s) (x)
      double centerx1 = upperp_rx;  // Only for readability, will be replaced by compiler
          optimization anyway.
      double centerx2 = lowerp_rx;
      double centery = middlep_ry;
      double center_repr1_x = upperp_x;
      double center_repr2_x = lowerp_x;
      double center_repr_y = middlep_y;  // always the same

      int xrange_from = 0;
      int xrange_to = _nx - 1;
      int yrange_from = 0;
      int yrange_to = _ny - 1;
      if (_WD > 0.0 && _WD < 90.0) {  // set x and y limits for points on the main grid to
          be considered in the close wake
        xrange_from = min(center_repr1_x, center_repr2_x) -
          max3(ay1, ay2, LR);
        xrange_to = max3(p_low_right_x, (center_repr1_x + ay1),
          center_repr2_x + ay2);
        yrange_from = center_repr_y - max3(LR, ay1, ay2);
        yrange_to = max3(p_up_left_y, (center_repr_y + ay1),
          (center_repr_y + ay2));
      }
      if (_WD > 90.0 && _WD < 180.0) {
        xrange_from =min(center_repr1_x, center_repr2_x) -
          max3(LR, ay1, ay2);
        xrange_to = max3(p_up_right_x, (center_repr1_x + ay1),
          (center_repr2_x + ay2));
        yrange_from = min3(p_low_left_y, (center_repr_y - ay1),
          (center_repr_y - ay2));
        yrange_to = center_repr_y + max3(LR, ay1, ay2);
      }
      if (_WD > 180.0 && _WD < 270.0) {
        xrange_from = min3(p_low_left_x, (center_repr1_x - ay1),
          center_repr2_x - ay2);
        xrange_to = max(center_repr1_x, center_repr2_x) +
          max3(LR, ay1, ay2);  // ay is always the same, LR is ~max(ax)
        yrange_from = min3(p_low_right_y, (center_repr_y - ay1),
          center_repr_y - ay2);
        yrange_to = center_repr_y + max3(LR, ay1, ay2);
      }
      if ((_WD > 270.0 && _WD < 360.0) && (_WD > 0.0 && _WD < 90.0)) {
        xrange_from = min3(p_low_left_x, (center_repr1_x - ay1),
          (center_repr2_x - ay2));
        xrange_to = max(center_repr1_x, center_repr2_x) +
          max3(LR, ay1, ay2);
        yrange_from = center_repr_y - max3(LR, ay1, ay2);
        yrange_to = max3(p_up_right_y, center_repr_y + ay1,
          center_repr_y + ay2);
      }
      xrange_from = round(xrange_from / _dx);
      xrange_to = round(xrange_to / _dx);
      yrange_from = round(yrange_from / _dy);
      yrange_to = round(yrange_to / _dy);

      // Cut ranges if they are out of model area
      if (xrange_from < 0) xrange_from = 0;
      if (yrange_from < 0) yrange_from = 0;
      if (xrange_to >= _nx) xrange_to = _nx - 1;
      if (yrange_to >= _ny) yrange_to = _ny - 1;

      double ay1_sq = pow(ay1, 2.0);  // only needs to be calculated once
      double ay2_sq = pow(ay2, 2.0);

      for (int z = foot; z <= top; z++) {  // for all points inside the z-extension of the recirculation zone,
        double ax1_sq = pow(ax1[z-foot], 2.0);  // reduce computation time by evaluating only once
        double ax2_sq = pow(ax2[z-foot], 2.0);

        for (int y = yrange_from; y <= yrange_to; y++) {  // the y extension
          for (int x = xrange_from; x <= xrange_to; x++) {  // and the x- extention of the recirculation
            double x_pr = rotmat[0][0] * (x * _dx) + rotmat[0][1] * (y * _dy);
            double y_pr = rotmat[1][1] * (y * _dy) + rotmat[1][0] * (x * _dx);

            // upper quarter ellipse
            // that are inside the recirculation ellipse
            if ((((pow(x_pr - centerx1, 2.0) / ax1_sq) +
              (pow(y_pr - centery, 2.0) / ay1_sq)) <= 1.0) &&
              (x_pr - centerx1 >= 0.0) && (y_pr - centery >= 0.0)) {
              double dl = 0.0;
              double dw = 0.0;
              // bool not_covered = true;
              if (y_pr >= middlep_ry) {  // upper wall is reference
                if (x_pr - ((upperp_ry - y_pr) / (upperp_ry - middlep_ry) *
                  (middlep_rx - upperp_rx) + upperp_rx) > 0.0) {
                  // points distance to the wall
                  double wall_x_pos = upperp_rx;  // use virtual wall inside Obstacle to get upper recirculation strength and maybe even stream reattachment, dom: 23.10.2014. Not enough, dom: 2014-11-13
                  dl = x_pr - wall_x_pos;
                  if (dl > 0.0) {
                    // walls distance to the end of the wake zone
                    dw = sqrt(abs(ax1_sq - pow(y_pr - centery, 2.0) /
                     ay1_sq * ax1_sq));  // + abs(wall_x_pos)  // removed as this makes no sence at all. dom: 2014-11-14

                    if (dw > 0.0) {  // this usually does not happen
                     double mod = 0.0;
                      // calculate modifications for this cell
                      mod = -1.0 * _ugrid->p(top, y, x) * pow(1.0 - dl/dw, 2.0);
                      if (_u_CW->p(z, y, x) == 999.0 || abs(_u_CW->p(z, y, x))
                       < abs(mod)) {
                       _u_CW->p(z, y, x) = mod;
                      }

                      // calculate modifications for this cell
                      mod = -1.0 * _vgrid->p(top, y, x) * pow(1.0 - dl/dw, 2.0);
                      if (_v_CW->p(z, y, x) == 999.0 || abs(_v_CW->p(z, y, x))
                       < abs(mod)) {
                       _v_CW->p(z, y, x) = mod;
                      }
```

```
                    }
                  }
                }
              }
            }

            // lower quarter ellipse
            // that are inside the wake ellipse
            if ((((pow(x_pr - centerx2, 2.0) / ax2_sq) +
              (pow(y_pr - centery, 2.0) / ay2_sq)) <= 1.0) &&
              (x_pr - centerx2 >= 0.0) && (y_pr - centery < 0.0)) {
              double dl = 0.0;
              double dw = 0.0;

              if (y_pr <= middlep_ry) {  // lower wall is reference
                if (x_pr - ((y_pr - lowerp_ry) / (middlep_ry - lowerp_ry) *
                  (middlep_rx - lowerp_rx) + lowerp_rx) > 0.0) {
                  // points distance to the wall
                  double wall_x_pos = lowerp_rx;  // use virtual wall inside Obstacle to get lower recirculation strength and maybe even stream reattachment, dom: 23.10.2014
                  dl = x_pr - wall_x_pos;
                  if (dl > 0.0) {
                    // walls distance to the end of the wake zone
                    dw = sqrt(abs(ax2_sq - pow(y_pr - centery, 2.0) /
                      ay2_sq * ax2_sq));  // abs removed

                    if (dw > 0.0) {
                      double mod = 0.0;
                      mod = -1.0 * _ugrid->p(top, y, x) * pow(1.0 - dl/dw, 2.0);
                      if (_u_CW->p(z, y, x) == 999.0 || abs(_u_CW->p(z, y, x))
                        < abs(mod)) {
                        _u_CW->p(z, y, x) = mod;
                      }
                      // calculate modifications for this cell
                      mod = -1.0 * _vgrid->p(top, y, x) * pow(1.0 - dl/dw, 2.0);
                      if (_v_CW->p(z, y, x) == 999.0 || abs(_v_CW->p(z, y, x))
                        < abs(mod)) {
                        _v_CW->p(z, y, x) = mod;
                      }
                    }  // end of if(dl > 0 & dw > 0)
                  }
                }
              }
            }
          }
        }
      }  // end for(z in zrange)
      delete[] ax1;
      delete[] ax2;
      // delete[] ay1;
      // delete[] ay2;
    }
  }
  // std::cout << "minLR = " << minLR << ", maxLR = " << maxLR << ", minLH = " << minLH << ", maxLH = " << maxLH << std::endl;
}
```

### 10.1.3 Successive over-relaxation

```
// Copyright 2015, Chair of Meteorology and Climatology Freiburg
// Author: Dominik Froehlich <dominik.froehlich@venus.uni-freiburg.de>


// #include "stdafx.h"  // Windows only!
// #include <ppl.h>
// #include <mutex>

// #include <math.h>
#include <time.h>  // for timer
#include <cmath>  // for abs
#include <algorithm>  // for max
#include <iostream>   // to output status
#include "./Tools.h"

using std::abs;
using std::pow;
using std::sqrt;

// ----------------------------------------------------------------------------
void Wind::SOR() {
  std::cout << "SOR routine called" << std::endl;
  std::cout.flush();
  // Start clock timer
  clock_t start = clock();
  // set abort criterion
  double const tardivg = pow(10.0, -5.0);  // was: pow(10.0, -5.0);

  // Init array for initial divgergence
  ThreeDMatrix<float> * divg;
  divg = new ThreeDMatrix<float>(_nz, _ny, _nx, 0.0);  // divg = new ThreeDMatrix<float>(_nz
      , _ny, _nx);

  // Init grid to store lambdas
  ThreeDMatrix<float> * lambdagr;
  lambdagr = new ThreeDMatrix<float>(_nz, _ny, _nx, 0.0);

  // Factor for the calculation of lambda 1/(Area of cell in x-dir)
  float const fact_dx = 1.0 / pow(_dx, 2.0);
  float const fact_dy = 1.0 / pow(_dy, 2.0);
  float const fact_dz = pow((_alpha_hor / _alpha_vert), 2.0) / pow(_dz, 2.0);  // as long as
       dz is fix (no change of dz with height)
  float const fact_dx2 = fact_dx * 2.0;  // to only have to calculate it once
  float const fact_dy2 = fact_dy * 2.0;
  float const fact_dz2 = fact_dz * 2.0;
  float const cell_fact_free = 1.0 / (fact_dx2 + fact_dy2 + fact_dz2);

  // Calculate initial divgergence
  for (int k = 0; k < _nz; k++) {
    for (int j = 0; j < _ny; j++) {
      for (int i = 0; i < _nx; i++) {
        if (_build->p(k, j, i) != 1.0) {  // No divgergence inside solid obstacles, no if
            needed as already 0 in flux grids!
          divg->p(k, j, i)= ((_ugrid->p(k, j, i+1) - _ugrid->p(k, j, i)) / _dx+ // write
              right side of the differential equation
                            (_vgrid->p(k, j+1, i) - _vgrid->p(k, j, i)) / _dy +
                            (_wgrid->p(k+1, j, i) - _wgrid->p(k, j, i)) / _dz)
                            * (-2.0) * pow(_alpha_hor, 2.0);
        }
      }
    }
  }

  // check _build and _bordergr dimensions (catch mismatch due to invalid file)
  _build->inRange(_nz-1, _ny-1, _nx-1, "_build");
  _bordergr->inRange(_nz-1, _ny-1, _nx-1, "_bordergr");
```

```

// Find best value for omega, refer to Press, W., 2007: Numerical Recipes, 3rd ed. p 1062 ff
float const dx_divg_dy_sq = pow((_dx / _dy), 2.0);

double const pi = atan(1.0) * 4.0;

// rho_Jacobi2 is only used as rho square, thus already saved like that
double const rho_Jacobi2 = pow(((cos(pi/_nx) + dx_divg_dy_sq * cos(pi/_ny)) / pow((
  1.0 + dx_divg_dy_sq), 2.0)), 2.0);

// parameter to store the local modification of lambda at a given point
// float lambda = 0.0;
float sum_delta_lambda = 0.0;
double abort = 0.0;  // Abort criteria
// Count total change in lambda per iteration as abort criteria
double sum_lambda_new  = 0.0;
size_t num_iter = 0;
double omega = 1.0;  // Initial over-relaxation parameter

// variable declaration in single CPU mode
double delta = 0.0;  // Set variable to store local difference
int borders = 0;  // create temporary border variable
double cell_fact = 0.0;
int ir_rbSOR = 0;  // Control range of x values
int lastk_p;  // store pre-variables
int nextk_p;
int k_p;
int lastj_p;
int nextj_p;
int j_p;
int lastk;  // store array positions
int nextk;
int lastj;
int nextj;
int lasti;
int nexti;
int pos;

// std::mutex mutex;  // only useful if called in parallel


// main iteration from here
// for (int iter = 1; iter <= _maxitr; iter++) {
for (int iter = 0; iter < _maxitr; iter++) {
  // reset total difference in lambda
  sum_delta_lambda = 0.0;
  // Control red black SOR
  // concurrency::parallel_for (int(0), 2, [&](int black) {
  for (int black = 0; black < 2; black++) {  // there is no true/false iteration in cpp
    /**
    // variable declaration in multicore mode
    double delta = 0.0;  // Set variable to store local difference
    int borders = 0;  // create temporary border variable
    double cell_fact = 0.0;
    int ir_rbSOR = 0;  // Control range of x values
    int lastk_p;  // store pre-variables
    int nextk_p;
    int k_p;
    int lastj_p;
    int nextj_p;
    int j_p;
    int lastk;  // store array positions
    int nextk;
    int lastj;
```

```
      int nextj;
      int lasti;
      int nexti;
      int pos;
      **/

      // Iterate over all heights
      for (int k = 0; k < _nz; k++) {
        // To only have this calculated once: constantes for this iteration
        k_p = k*_nx*_ny;
        lastk_p = (k - 1)*_nx*_ny;
        nextk_p = (k + 1)*_nx*_ny;
        // Iterate over y
        for (int j = 0; j < _ny; j++) {
          // Constants for y
          j_p = j * _nx;
          lastj_p = (j - 1) * _nx;
          nextj_p = (j + 1) * _nx;

          // Control range of x values
          ir_rbSOR = (black + j) % 2;  // should be way faster doing exactly the same

          // Iterate over every second cell according to ir_rbSOR
          for (int i = ir_rbSOR; i < _nx; i += 2) {
            // calculate position of (k, j, i)th element only once
            pos = k_p + j_p + i;
            // make sure point is not located inside a solid obstacle
            if (_build->p(pos) >= 1.0) continue;
            // calculate neighbouring points positions
            lasti = k_p + j_p + i - 1;
            nexti = k_p + j_p + i + 1;
            lastj = k_p + lastj_p + i;
            nextj = k_p + nextj_p + i;
            lastk = lastk_p + j_p + i;
            nextk = nextk_p + j_p + i;

            // Reset delta
            delta = 0.0;
            borders = _bordergr->p(pos);  // create temporary border variable

            // SOR for a non-border situation, taken from Roeckle (1990) p. 57
            if (borders == 0) {
              // calculate delta-lambda for all points
              delta = (cell_fact_free *
                ((lambdagr->p(lasti) + lambdagr->p(nexti)) *
                fact_dx +
                (lambdagr->p(lastj) + lambdagr->p(nextj)) *
                fact_dy +
                (lambdagr->p(lastk) + lambdagr->p(nextk)) *
                fact_dz -
                divg->p(pos)) -
                lambdagr->p(pos)) *
                omega;    // include over-relaxation
            } else {
              // if cell is border cell
              cell_fact = 0.0;

              // ### if  border in -x or +x direction ###
              if (borders >= 10000) {
                // if open or closed border in -x direction
                if (borders >= 100000) {
                  if (borders < 200000) {  // if closed border
                    cell_fact = cell_fact + fact_dx;  // normal notation said to be faster
                      than +=
                    borders = borders - 100000;
                  } else {
                    cell_fact = cell_fact + fact_dx2;
                    borders = borders - 200000;
                  }
                  if (borders >= 10000) {  // if also border in +x direction
                    if (borders < 20000) {  // if closed border
                      // reduce cell factor by fact_dx again, to have 0 in total
                      cell_fact = cell_fact - fact_dx;
                      borders = borders - 10000;
                    } else {                 // if open border
                      borders = borders - 20000;
                    }
                  } else {  // end of if border in -x and +x dir // if only border in -x dir
                    delta = delta + lambdagr->p(nexti) * fact_dx;
                  }
                } else {  // end of if border in -x and +x dir // if only border in +x dir
                  if (borders < 20000) {  // if closed border
                    cell_fact = cell_fact + fact_dx;
                    borders = borders - 10000;
                  } else {                 // if open border
                    cell_fact = cell_fact + fact_dx2;
                    borders = borders - 20000;
                  }
                  delta = delta + lambdagr->p(lasti) * fact_dx;
                }
              } else {   // if no border in xdirection
                delta = delta + (lambdagr->p(lasti) +
                  lambdagr->p(nexti)) * fact_dx;
                cell_fact = cell_fact + fact_dx2;
              }  // ### end of x border section ###

              // ### if  border in -y or +y direction ###
              if (borders >= 100) {
                // if open or closed border in -y direction
                if (borders >= 1000) {
                  if (borders < 2000) {  // if closed border in -y dir
                    cell_fact = cell_fact + fact_dy;
                    borders = borders - 1000;
                  } else {                 // if open border in -y dir
                    cell_fact = cell_fact + fact_dy2;
                    borders = borders - 2000;
                  }
                  if (borders >= 100) {  // if also border in +y direction
                    if (borders < 200) {  // if closed border
                      // reduce cell factor by fact_dy again, to have 0 in total
                      cell_fact = cell_fact - fact_dy;
                      borders = borders - 100;
                    } else {                 // if open border
                      borders = borders - 200;
                    }
                  } else {  // end of if border in -y and +y dir # if only border in -y dir
                    delta = delta + lambdagr->p(nextj) * fact_dy;
                  }
                } else {  // end if border in -y dir # if border in +y direction
                  if (borders < 200) {  // if closed border in +y dir
                    cell_fact = cell_fact + fact_dy;
                    borders = borders - 100;
                  } else {                 // if open border in +y dir
                    cell_fact = cell_fact + fact_dy2;
                    borders = borders - 200;
                  }
                  delta = delta + lambdagr->p(lastj) * fact_dy;
                }
              } else {   // if no border in y direction
                delta = delta + (lambdagr->p(lastj) +
                  lambdagr->p(nextj)) * fact_dy;
                cell_fact = cell_fact + fact_dy2;
              }  // ### end of y border section ###

              //  ### if  border in -z or +z direction ###
              if (borders >= 1) {
                // if open or closed border in -z direction
                if (borders >= 10) {
                  if (borders < 20) {  // if closed border
                    cell_fact = cell_fact + fact_dz;
                    borders = borders - 10;
                  } else {                  // if open border
                    cell_fact = cell_fact + fact_dz2;
                    borders = borders - 20;
                  }
                  if (borders >= 1) {  // if also border in +z direction
                    if (borders < 2) {  // if closed border
                      cell_fact = cell_fact - fact_dz;  // reduce cell factor by fact_dz
                          again, to have 0 in total
                      borders = borders - 1;
                    } else {                  // if open border
                      borders = borders - 2;
                    }
                  } else {  // end of if border in -z and +z dir # if only border in -z dir
                    delta = delta + lambdagr->p(nextk) * fact_dz;
                  }
                } else {   // end if border in -z dir # if border in +z direction
                  if (borders < 2) {  // if closed border
                    cell_fact = cell_fact + fact_dz;
                    borders = borders - 1;
                  } else {                  // if open border
                    cell_fact = cell_fact + fact_dz2;
                    borders = borders - 2;
                  }
                  delta = delta + lambdagr->p(lastk) * fact_dz;
                }
              } else {   // if no border in z direction
                delta = delta + (lambdagr->p(lastk) +
                  lambdagr->p(nextk)) * fact_dz;
                cell_fact = cell_fact + fact_dz2;
              }  // ### end of z border section ###

              // Calculate changes for this cell
              delta = ((delta - divg->p(pos)) / cell_fact -
                lambdagr->p(pos)) * omega;
            }  // end if cell is border cell

            // Write modification to lambdagr
            if (delta != 0.0) {
              // std::cout << "delta=" << abs(delta) << std::endl;
              // the following must be thread safe
              // mutex.lock();  // only one thread at a time from here
              lambdagr->p(pos) = lambdagr->p(pos) + delta;
              sum_delta_lambda = sum_delta_lambda + abs(delta);  // total delta lambda
              sum_lambda_new = sum_lambda_new + abs(lambdagr->p(pos));
              // mutex.unlock();  // and go for the next thread
            }

          }
        }  //  end of for all of the grid points
      }
    // });  // end of parallel_for black-red iteration
    }  // end of black-red iteration single core

    // ### new abort crietria after Roeckle 1990 ###
    // std::cout << "sum_delta_lambda=" << sum_delta_lambda << ", sum_lambda_new=" <<
        sum_lambda_new << std::endl;
    abort = sum_delta_lambda / sum_lambda_new;

    // std::cout << "SOR step #" << iter << ", abort criteria: " << abort << " < "
    //    << tardivg << ", omega=" << omega << std::endl;
    if (abort != abort) {  // stop SOR if abort becomes NaN!
      std::cout << "SOR failed in step #" << iter << ", invalid abort criteria: " << abort
          << std::endl;
      std::cout.flush();
      break;
    }
    if (abort < tardivg) {
        num_iter = iter + 1;  // save number of iterations used for display
        break;  // break iteration if abort criteria becomes TRUE
      }

    if (iter == 1) {   // include Chebyshev accerleration
      omega = 1.0 / (1.0 - 0.5 * rho_Jacobi2);
    }
    if (iter > 1) {   // again increase omega to final value
      omega = 1.0 / (1.0 - 0.25 * rho_Jacobi2 * omega);
    }
  }  // end of iteration cycle

  // stop timer
  clock_t stop = clock();  // total time (double) = stop - start

  // Do something with time
  _SORtime = 1000.0 * (stop - start) / CLOCKS_PER_SEC;
  _SORsteps = num_iter;
  // std::cout << "Computing time for SOR (" << num_iter << " steps) = " <<
  //    _SORtime << " ms\n";
  // std::cout.flush();


  // Write results respecting obstacles
  // x-direction
  double stabx = 1.0 / (2.0 * pow(_alpha_hor, 2.0) * _dx);
  int nx1 = _nx - 1;
  for (int k = 0; k < _nz; k++) {
    for (int j = 0; j < _ny; j++) {
      // left border
      if (_build->p(k, j, 0) < 0.7) {  // is there not an obstacle?
        _ugrid->p(k, j, 0) = _ugrid->p(k, j, 0) + stabx *
          lambdagr->p(k, j, 0);
      }
      // right border
      if (_build->p(k, j, nx1) < 0.7) {  // is there not an obstacle?
        _ugrid->p(k, j, _nx) = _ugrid->p(k, j, _nx) + stabx *
          lambdagr->p(k, j, nx1);
      }
      // main grid
      for (int i = 1; i <= nx1; i++) {
        if ((_build->p(k, j, i) < 0.7) && (_build->p(k, j, i-1) < 0.7)) {  // is there not
            an obstacle?
          _ugrid->p(k, j, i) = _ugrid->p(k, j, i) + stabx *
            (lambdagr->p(k, j, i) - lambdagr->p(k, j, i-1));
        }
      }
    }
  }

  // y-direction
  double staby = 1.0 / (2.0 * pow(_alpha_hor, 2.0) * _dy);
```

```
int ny1 = _ny - 1;
for (int k = 0; k < _nz; k++) {
  for (int i = 0; i < _nx; i++) {
    // front border
    if (_build->p(k, 0, i) < 0.7) {  // is there not an obstacle?
      _vgrid->p(k, 0, i) = _vgrid->p(k, 0, i) + staby *
        lambdagr->p(k, 0, i);
    }
    // rear border
    if (_build->p(k, ny1, i) < 0.7) {  // is there not an obstacle?
      _vgrid->p(k, _ny, i) = _vgrid->p(k, _ny, i) + staby *
        lambdagr->p(k, ny1, i);
    }
    // main grid
    for (int j = 1; j < ny1; j++) {
      if ((_build->p(k, j, i) < 0.7) && (_build->p(k, j-1, i) < 0.7)) {  // is there not
          an obstacle?
        _vgrid->p(k, j, i) = _vgrid->p(k, j, i) + staby *
          (lambdagr->p(k, j, i) - lambdagr->p(k, j - 1, i));
      }
    }
  }
}

// z-direction
double stabz = 1 / (2 * pow(_alpha_vert, 2.0) * _dz);
int nz1 = _nz - 1;
for (int i = 0; i < _nx; i++) {
  for (int j = 0; j < _ny; j++) {
    // upper border
    if (_build->p(nz1, j, i) < 0.7) {  // is there not an obstacle?
      _wgrid->p(_nz, j, i) = _wgrid->p(_nz, j, i) + stabz *
        lambdagr->p(nz1, j, i);
    }
    // main grid
    for (int k = 1; k < nz1; k++) {
      if ((_build->p(k, j, i) < 0.7) && (_build->p(k-1, j, i) < 0.7)) {  // is there not
          an obstacle?
        _wgrid->p(k, j, i) = _wgrid->p(k, j, i) + stabz *
          (lambdagr->p(k, j, i) - lambdagr->p(k-1, j, i));
      }
    }
  }
}


// Calculate remaining divergence
float max_div_left = 0.0;
float total_div_left = 0.0;
float local_div = 0.0;
for (int k = 0; k < _nz; k++) {
  for (int j = 0; j < _ny; j++) {
    for (int i = 0; i < _nx; i++) {
      if (_build->p(k, j, i) != 1.0) {  // No divgergence inside solid obstacles
        local_div = (_ugrid->p(k, j, i+1) - _ugrid->p(k, j, i)) / _dx +
                    (_vgrid->p(k, j+1, i) - _vgrid->p(k, j, i)) / _dy +
                    (_wgrid->p(k+1, j, i) - _wgrid->p(k, j, i)) / _dz;
        total_div_left += abs(local_div);
        if (abs(local_div) > max_div_left) max_div_left += abs(local_div);
      }
    }
  }
}

// print remaining divergence
```

```
  std::cout << "Maximum local divergence remaining: " << max_div_left << std::endl;
  std::cout.flush();
  std::cout << "Total grid divergence remaining: " << total_div_left << std::endl;
  std::cout.flush();

  delete divg;
  delete lambdagr;
}
```

## 10.2 Distribution maps

Distribution maps for all three areas of input (Place of the old synagogue in both designs (compare to section 6.8.1) as well as the Institutes Quarter(section 6.8.2)) are created for all the meteorological conditions contained by the applied meteorological data input file (see tab. 6.4). To keep the size of the results section limited, only the most interesting results are presented there. Other distribution maps, showing information useful for comparison, are placed in the following section in the Appendix.

### 10.2.1 Institutes Quarter

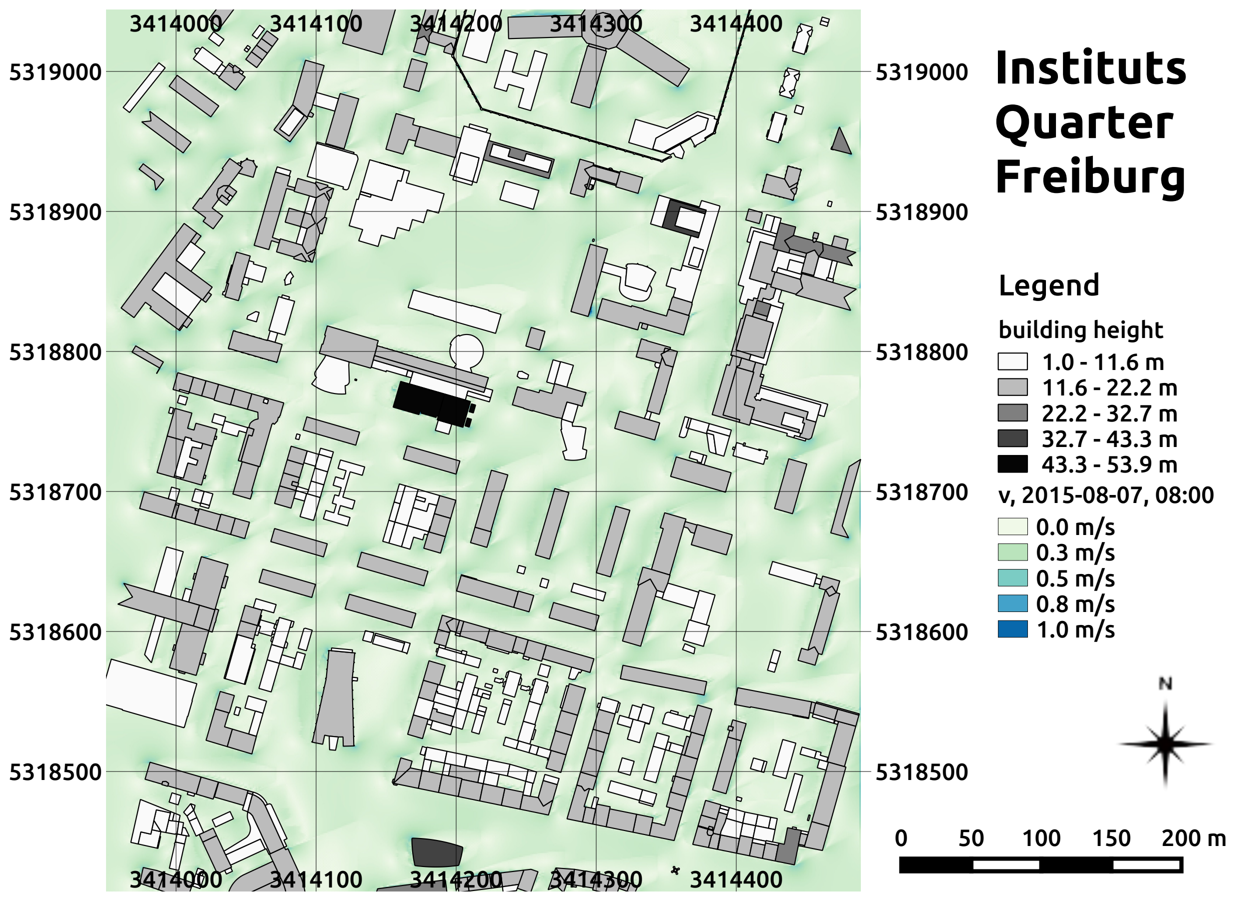


**fig. 10.1:** Spatial distribution of wind speed for the 07th of August 2015 at 08:00 in 1.5 m height at the Institutes Quarter in Freiburg, south-west Germany calculated by the SkyHelios model. Incident wind in 10 m height is 1.2 m/s from 248°. Building heights above ground (m) as well as tree properties and positions (compare to fig. 6.15) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

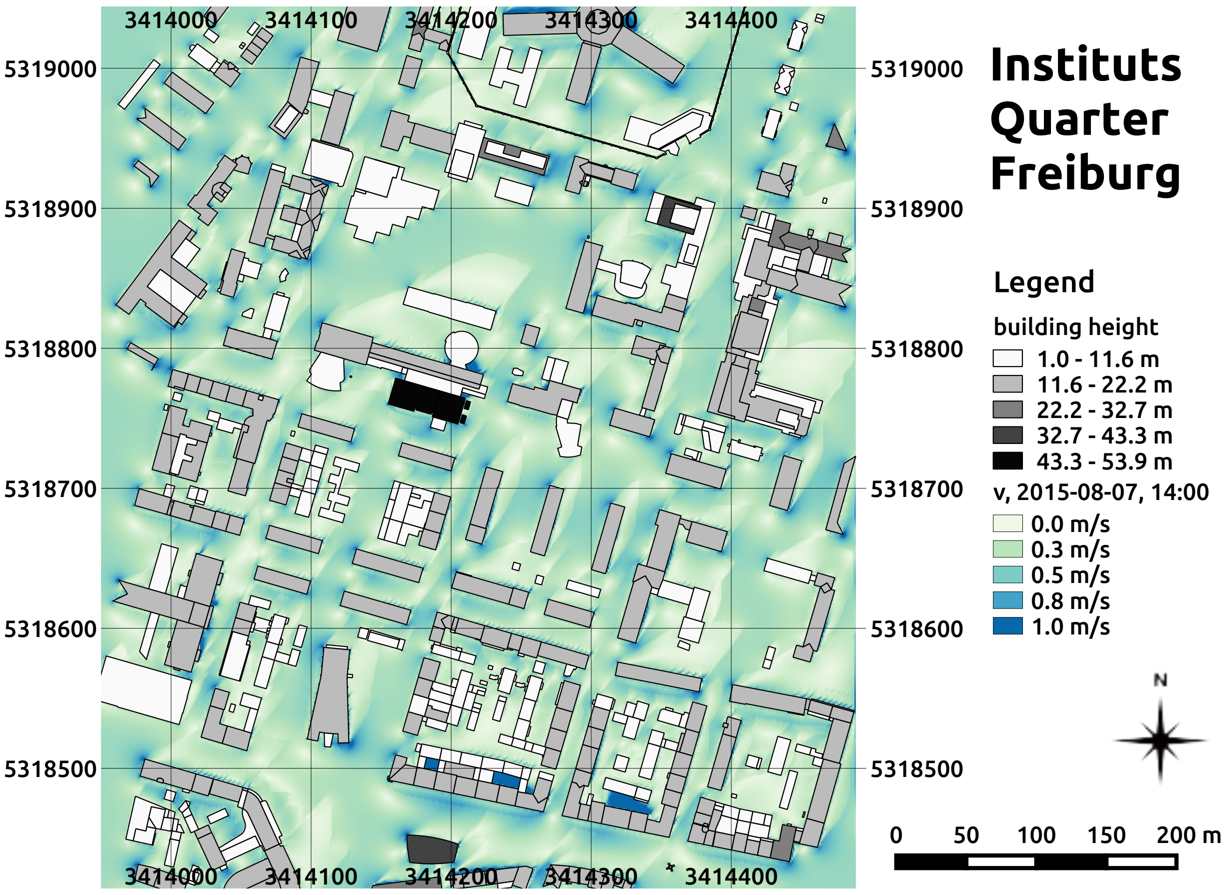


**fig. 10.2:** Spatial distribution of wind speed for the 07$^{th}$ of August 2015 at 14:00 in 1.5 m height at the Institutes Quarter in Freiburg, south-west Germany calculated by the SkyHelios model. Incident wind in 10 m height is 3.0 m/s from 221°. Building heights above ground (m) as well as tree properties and positions (compare to fig. 6.15) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

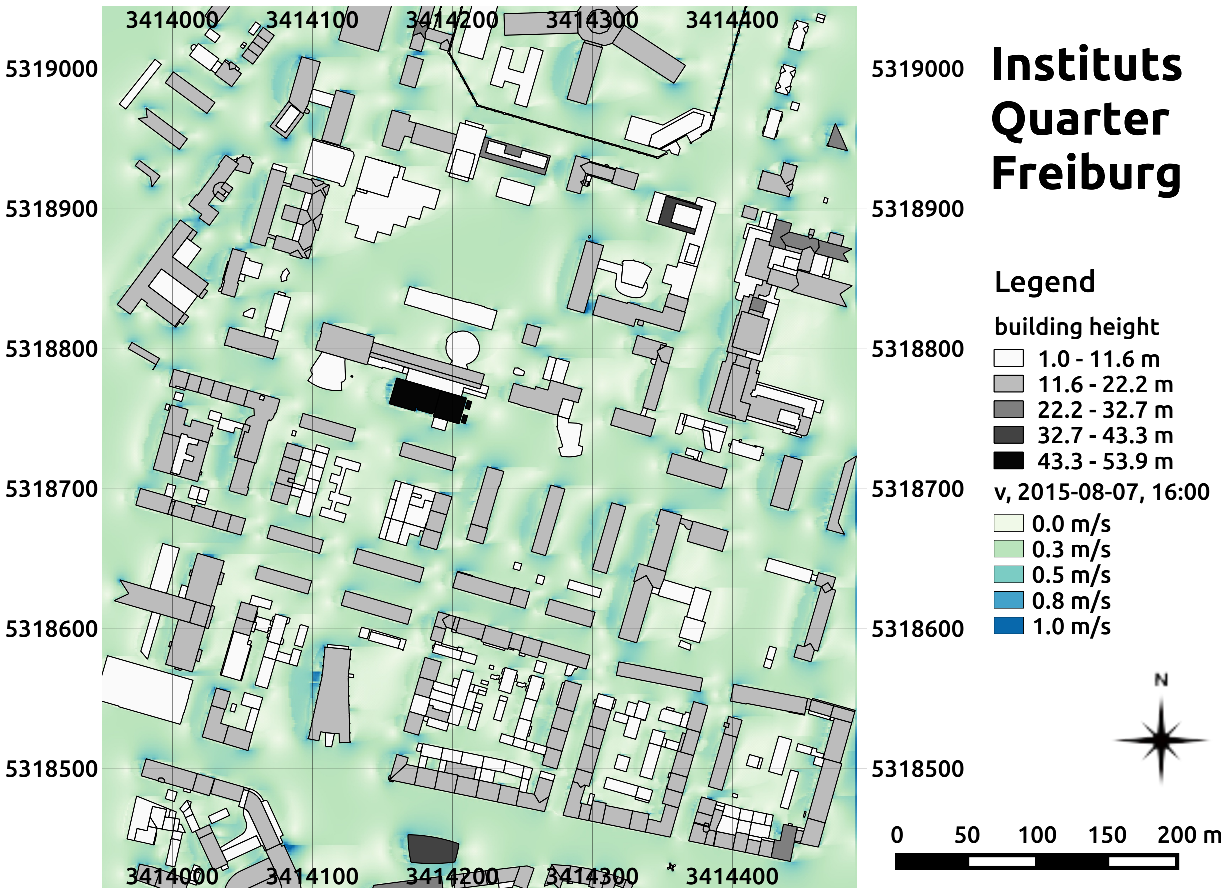


**fig. 10.3:** Spatial distribution of wind speed for the 07th of August 2015 at 16:00 in 1.5 m height at the Institutes Quarter in Freiburg, south-west Germany calculated by the SkyHelios model. Incident wind in 10 m height is 2.0 m/s from 269°. Building heights above ground (m) as well as tree properties and positions (compare to fig. 6.15) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

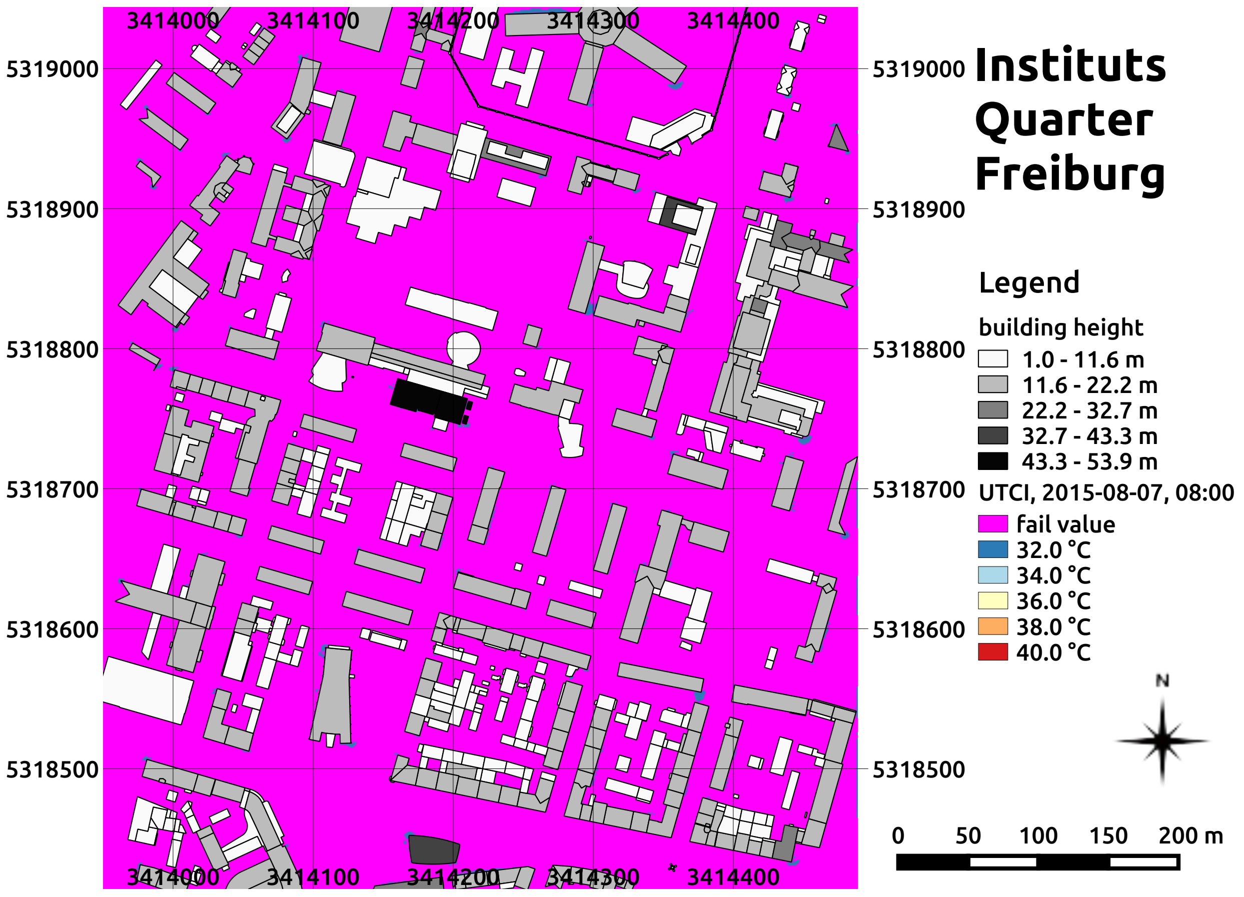


**fig. 10.4:** UTCI on the 07th of August 2015 at 08:00 at the Institutes Quarter in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

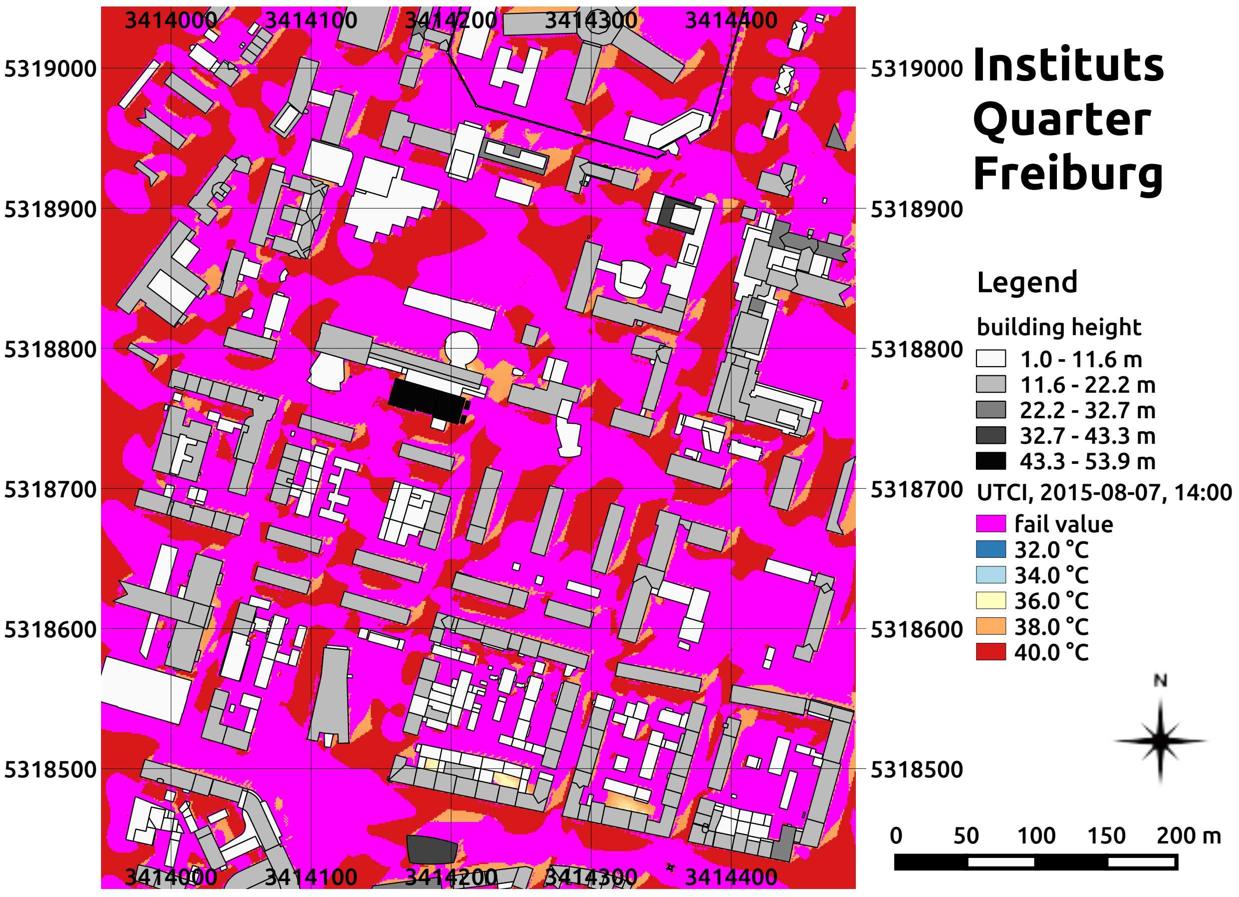


**fig. 10.5:** UTCI on the 07th of August 2015 at 14:00 at the Institutes Quarter in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

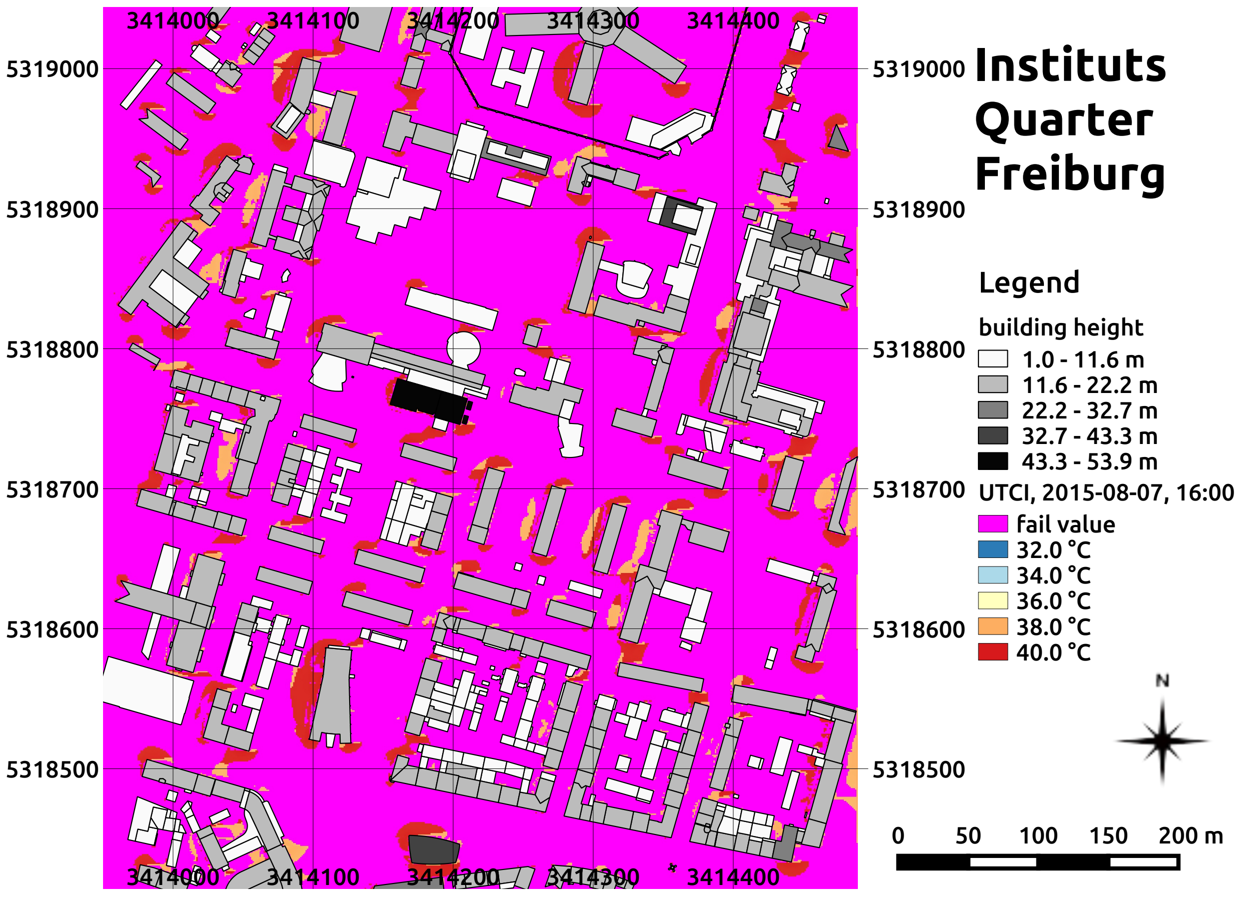


**fig. 10.6:** UTCI on the 07th of August 2015 at 16:00 at the Institutes Quarter in Freiburg, south-west Germany calculated by the SkyHelios model. Building heights above ground (m) are provided by the municipality of Freiburg. The coordinates are based on the projected coordinate system DHDN / Gauss-Krüger zone 3 (EPSG: 31467).

## 10.3 Comparison to measurements

Not all of the comparisons made to assess the accuracy of the advanced SkyHelios model can be presented in the results for not putting strain on the readability of the chapter. To maintain the sufficiency of this dissertation, the graphs not presented in the results section can be found in the following subsections.

### 10.3.1 Wind speed

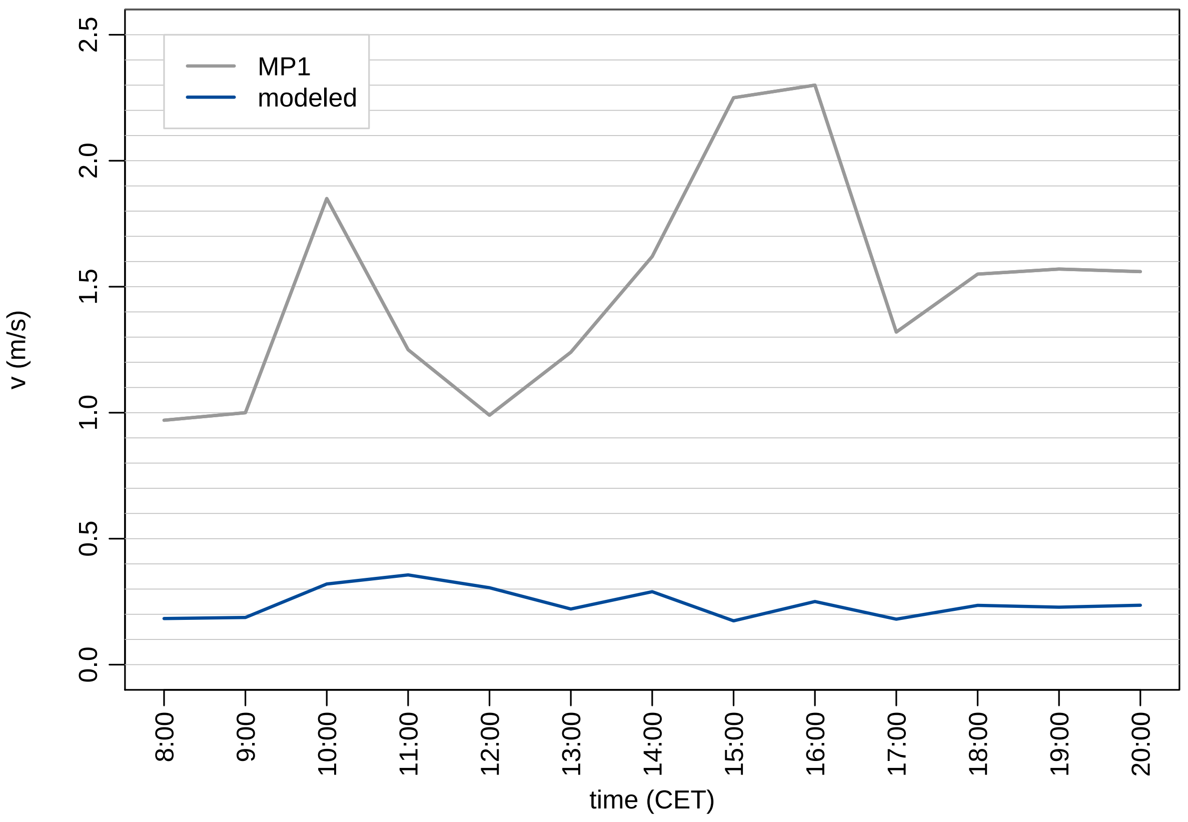


**fig. 10.7:** Comparison of wind speed ($v$) on the place of the old synagogue on the 01$^{st}$ of July 2008 as measured at measuring point 1 (MP1) by a mobile biometeorological station (grey) and calculated by the SkyHelios model (blue).

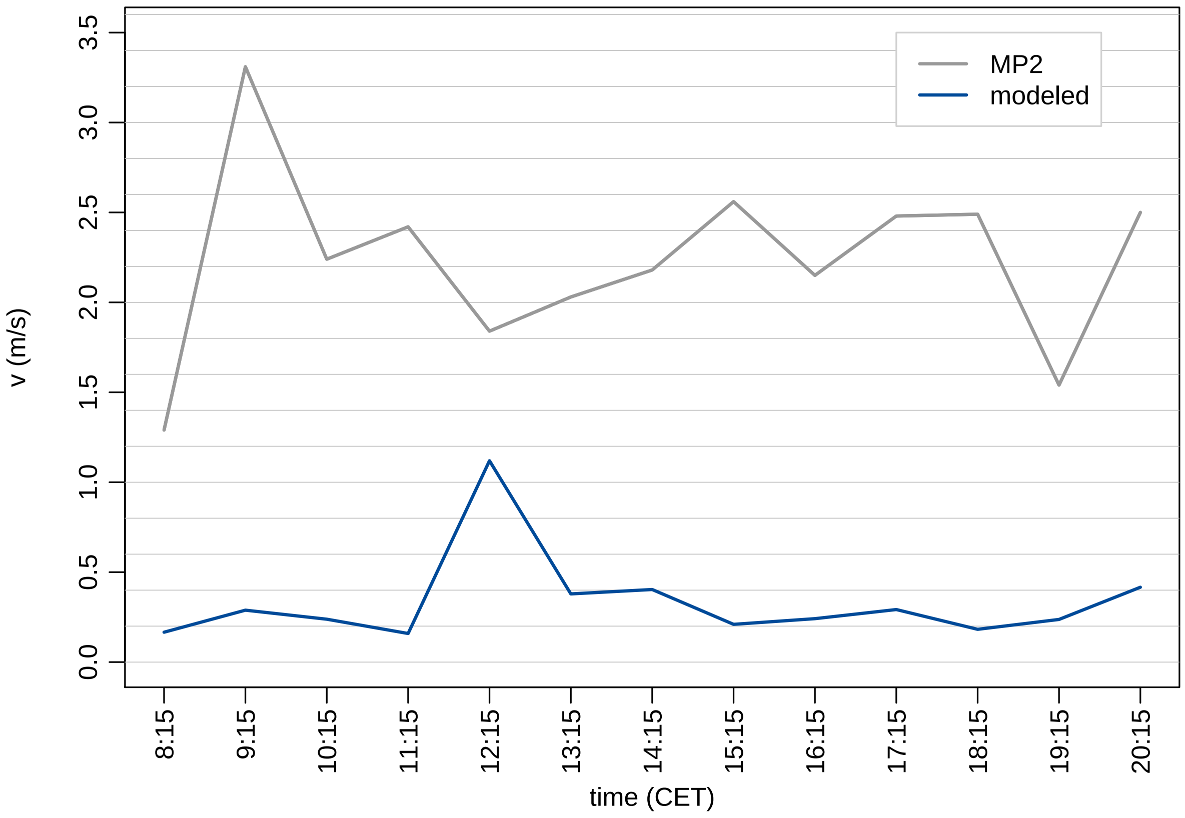


**fig. 10.8:** Comparison of wind speed ($v$) on the place of the old synagogue on the 01$^{st}$ of July 2008 as measured at measuring point 2 (MP2) by a mobile biometeorological station (grey) and calculated by the SkyHelios model (blue).

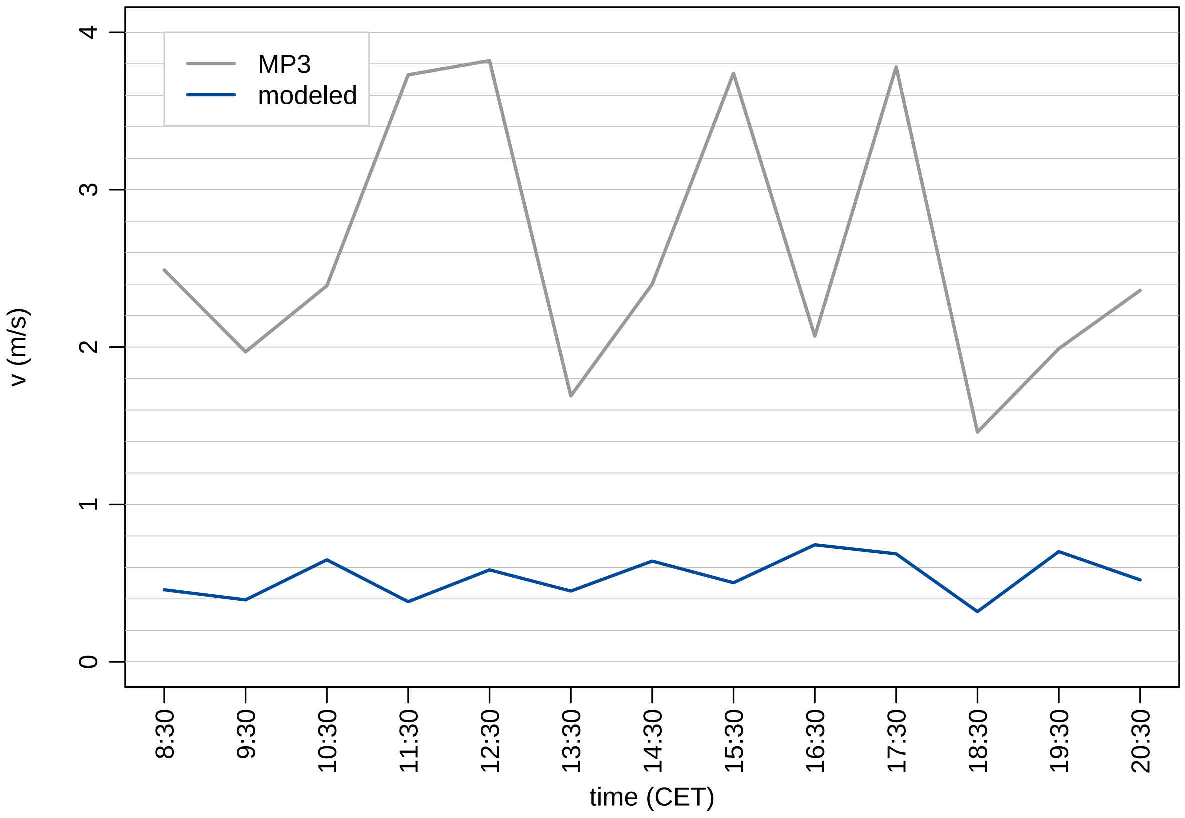


**fig. 10.9:** Comparison of wind speed ($v$) on the place of the old synagogue on the 01$^{st}$ of July 2008 as measured at measuring point 3 (MP3) by a mobile biometeorological station (grey) and calculated by the SkyHelios model (blue).

### 10.3.2 Wind direction

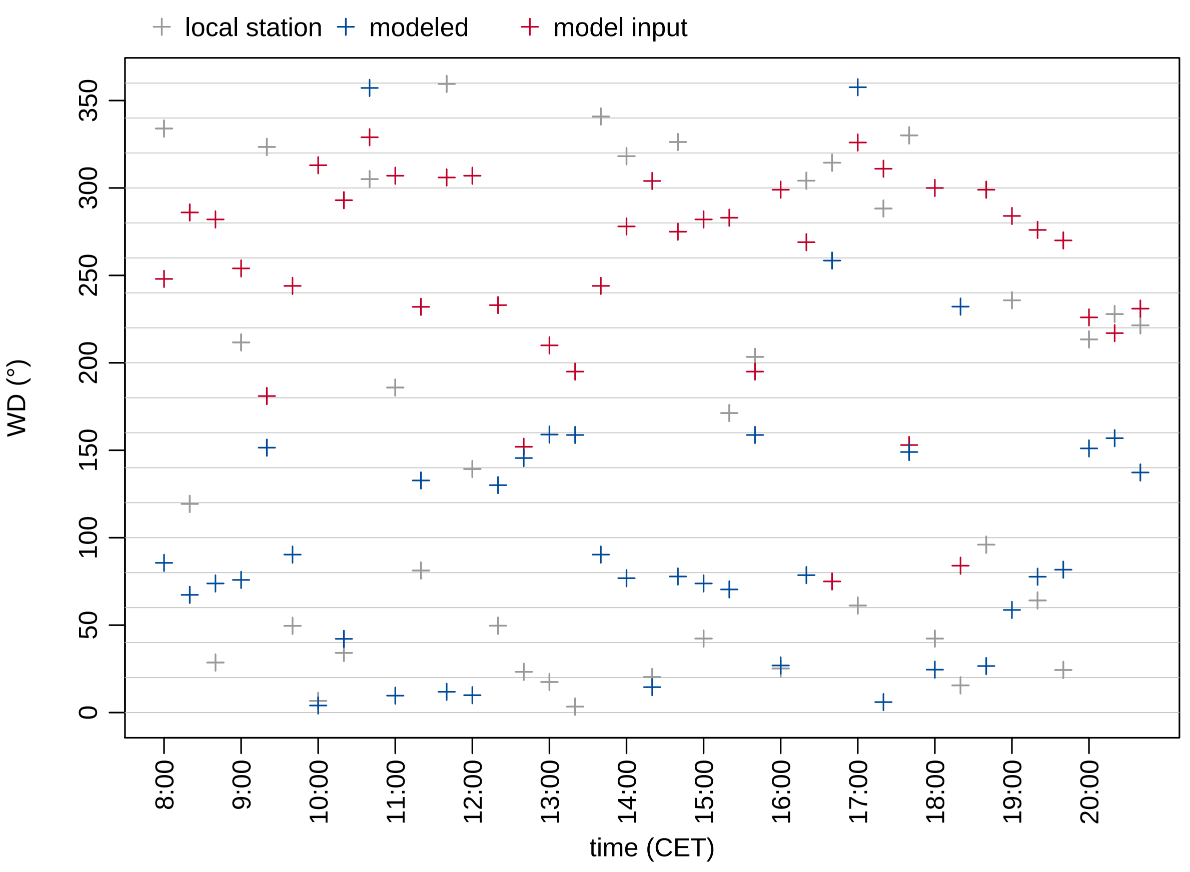


**fig. 10.10:** Comparison of wind direction ($WD$) on the place of the old synagogue on the 01$^{st}$ of July 2008 by a local urban climate station (grey) and calculated by the SkyHelios model (blue). The input values for the advanced SkyHelios model are given by the red crosses for comparison.

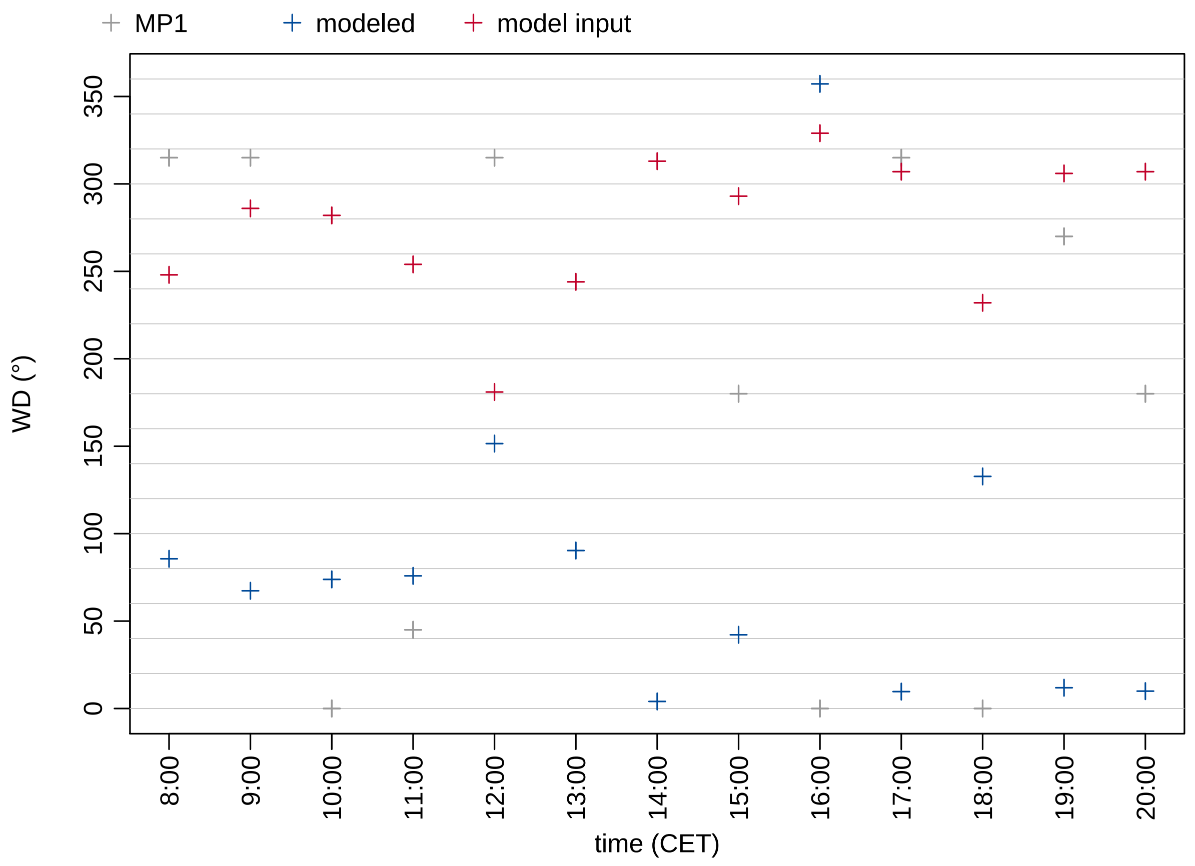


**fig. 10.11:** Comparison of wind direction ($WD$) on the place of the old synagogue on the 01$^{st}$ of July 2008 as measured at measuring point 1 (MP1) by a mobile biometeorological station (grey) and calculated by the SkyHelios model (blue). The input values for the advanced SkyHelios model are given by the red crosses for comparison.

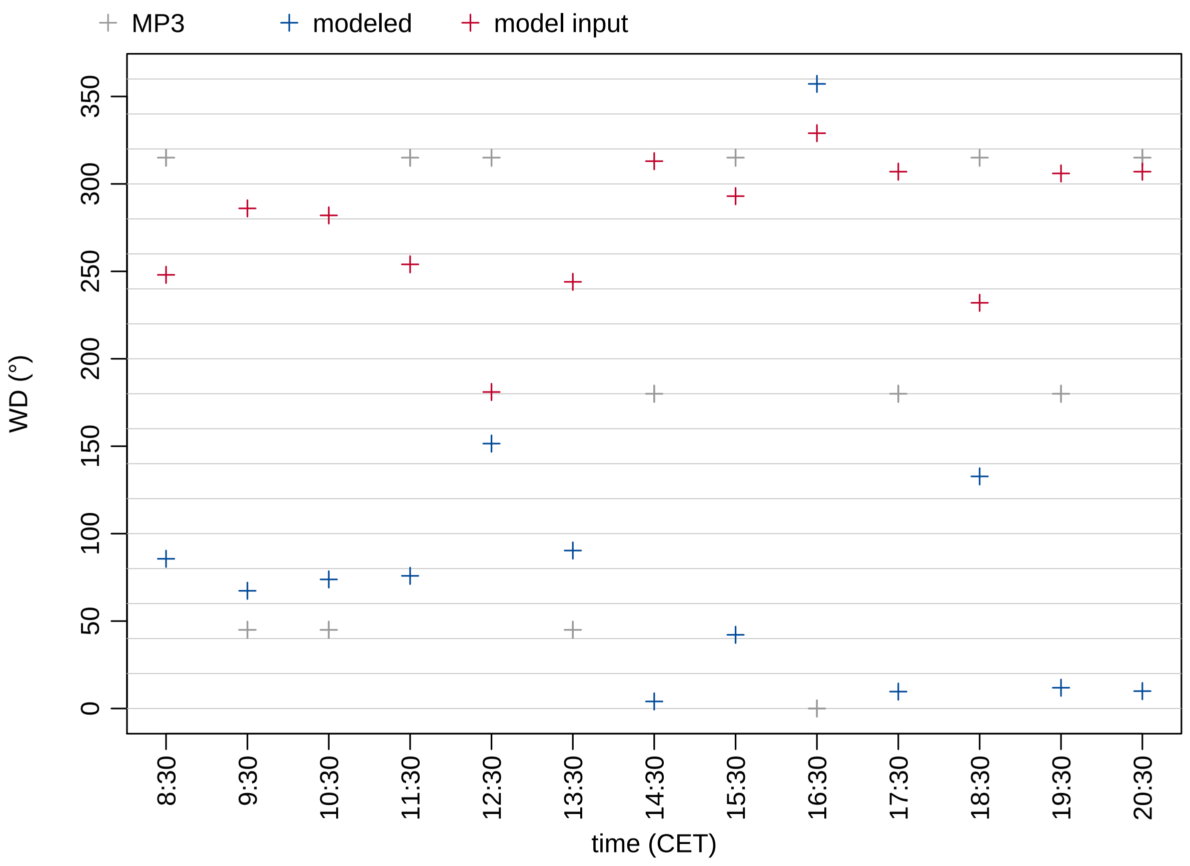


**fig. 10.12:** Comparison of wind direction ($WD$) on the place of the old synagogue on the 01$^{st}$ of July 2008 as measured at measuring point 3 (MP3) by a mobile biometeorological station (grey) and calculated by the SkyHelios model (blue). The input values for the advanced SkyHelios model are given by the red crosses for comparison.

### 10.3.3 Mean radiant Temperature

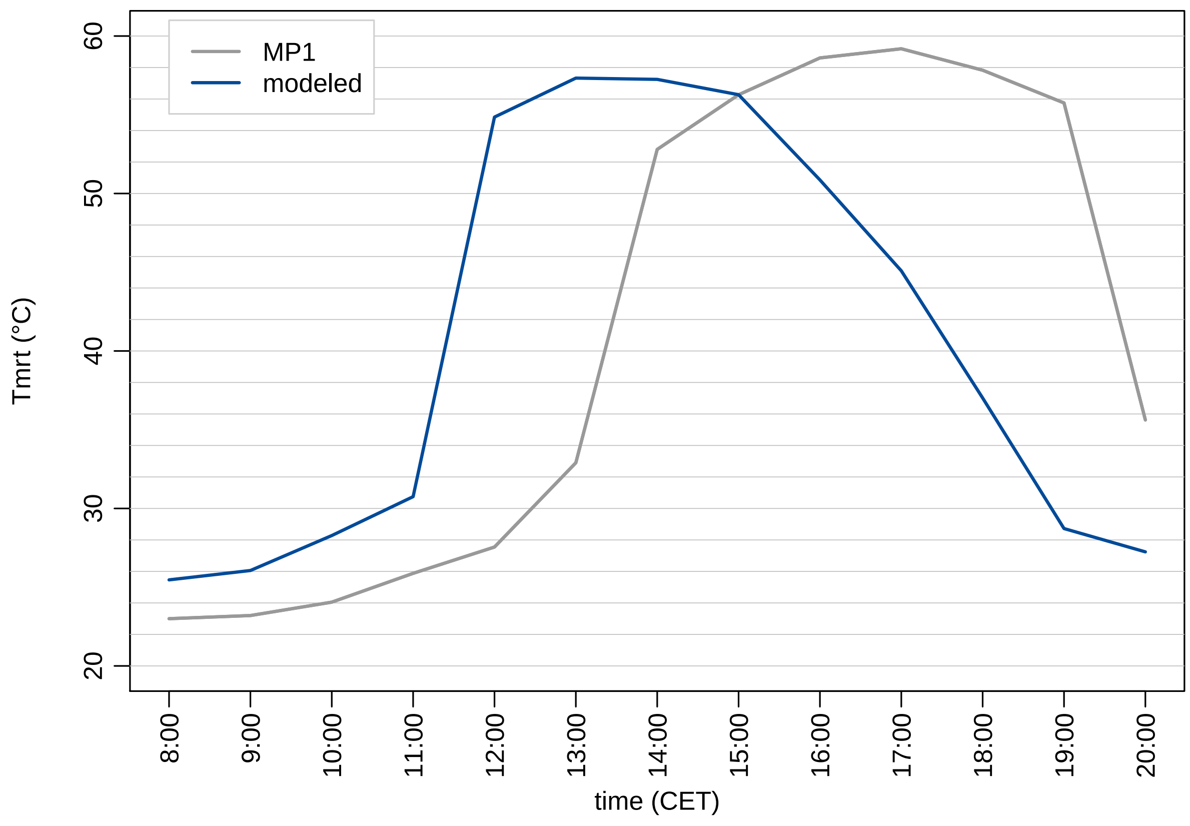


**fig. 10.13:** Comparison of mean radiant temperature ($T_{mrt}$) on the place of the old synagogue on the 01$^{st}$ of July 2008 as measured at measuring point 1 (MP1) by a mobile biometeorological station (grey) and calculated by the SkyHelios model (blue).

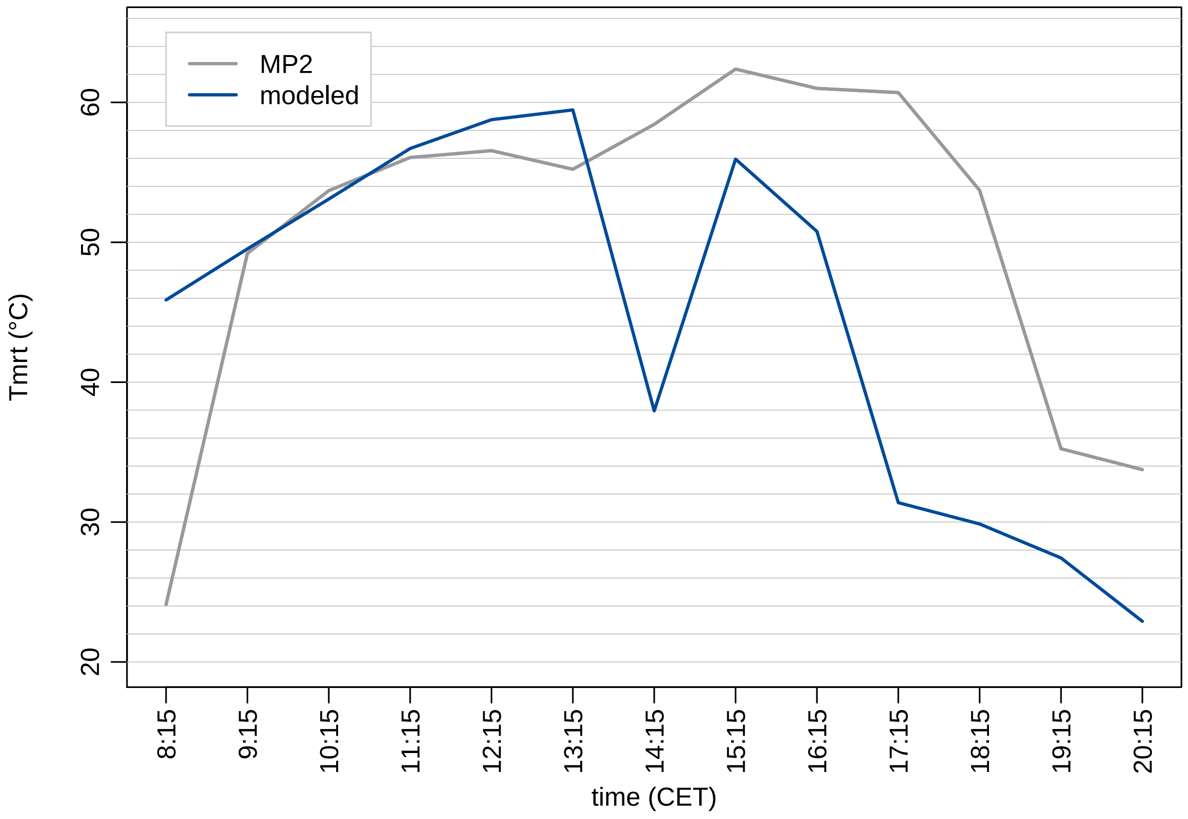


**fig. 10.14:** Comparison of mean radiant temperature ($T_{mrt}$) on the place of the old synagogue on the 01$^{st}$ of July 2008 as measured at measuring point 2 (MP2) by a mobile biometeorological station (grey) and calculated by the SkyHelios model (blue).

### 10.3.4 Physiologically Equivalent Temperature

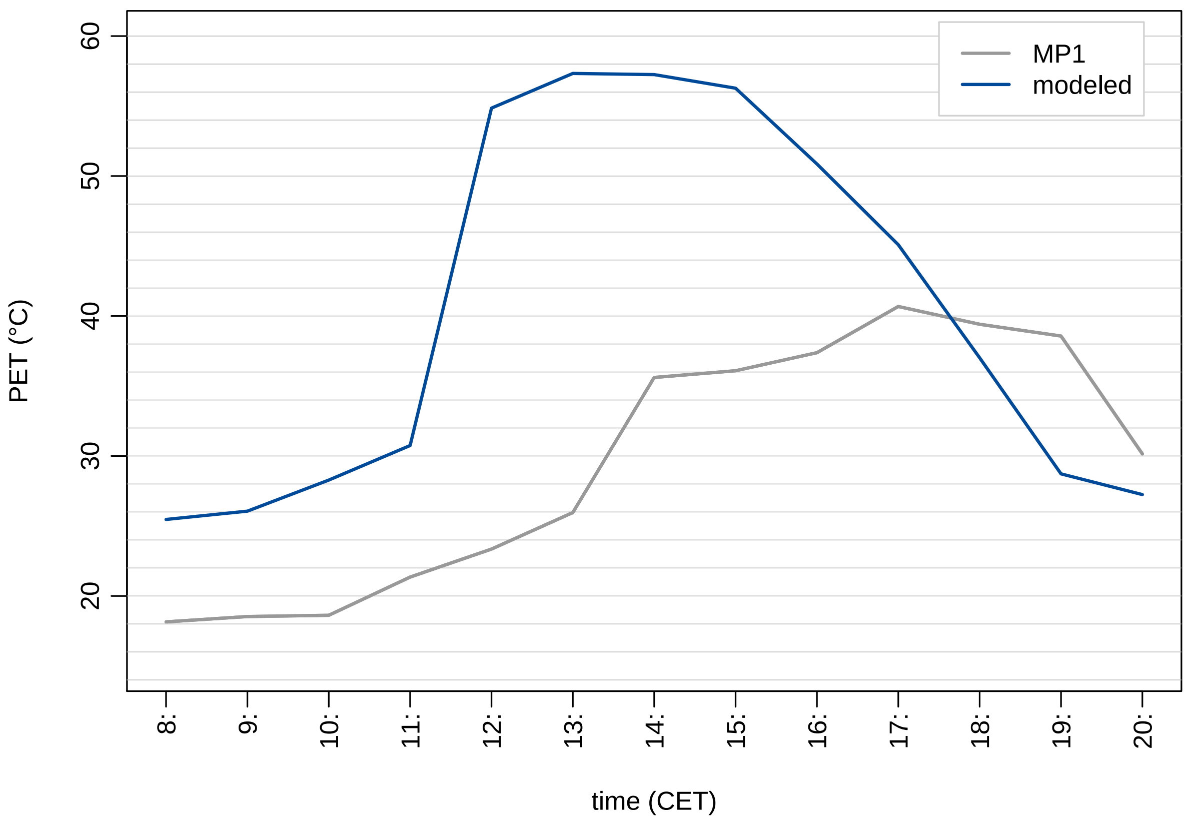


**fig. 10.15:** Comparison of Physiologically Equivalent Temperature (PET) on the place of the old synagogue on the 01*st* of July 2008 as as calculated based on measurements by a mobile biometeorological station at measuring point 1 (MP1) (grey) and calculated by the SkyHelios model (blue).

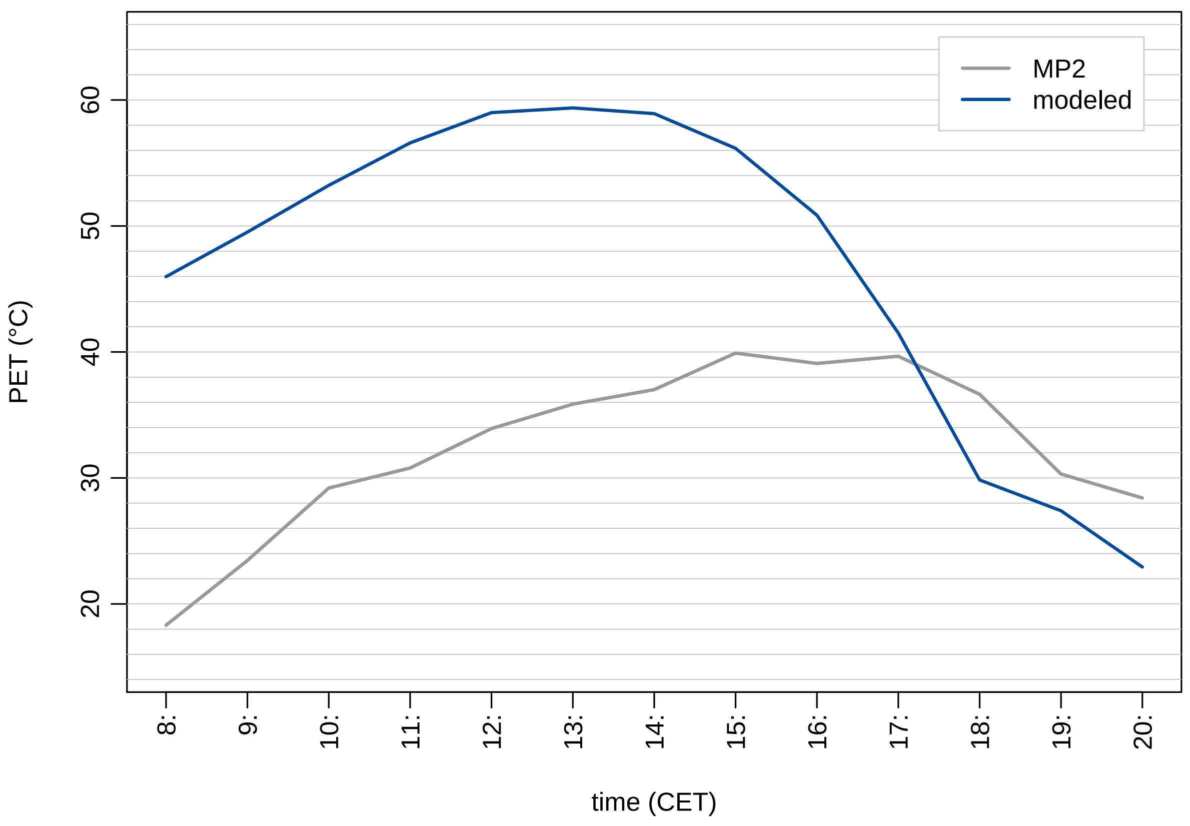


**fig. 10.16:** Comparison of Physiologically Equivalent Temperature (PET) on the place of the old synagogue on the 01$^{st}$ of July 2008 as as calculated based on measurements by a mobile biometeorological station at measuring point 2 (MP2) (grey) and calculated by the SkyHelios model (blue).

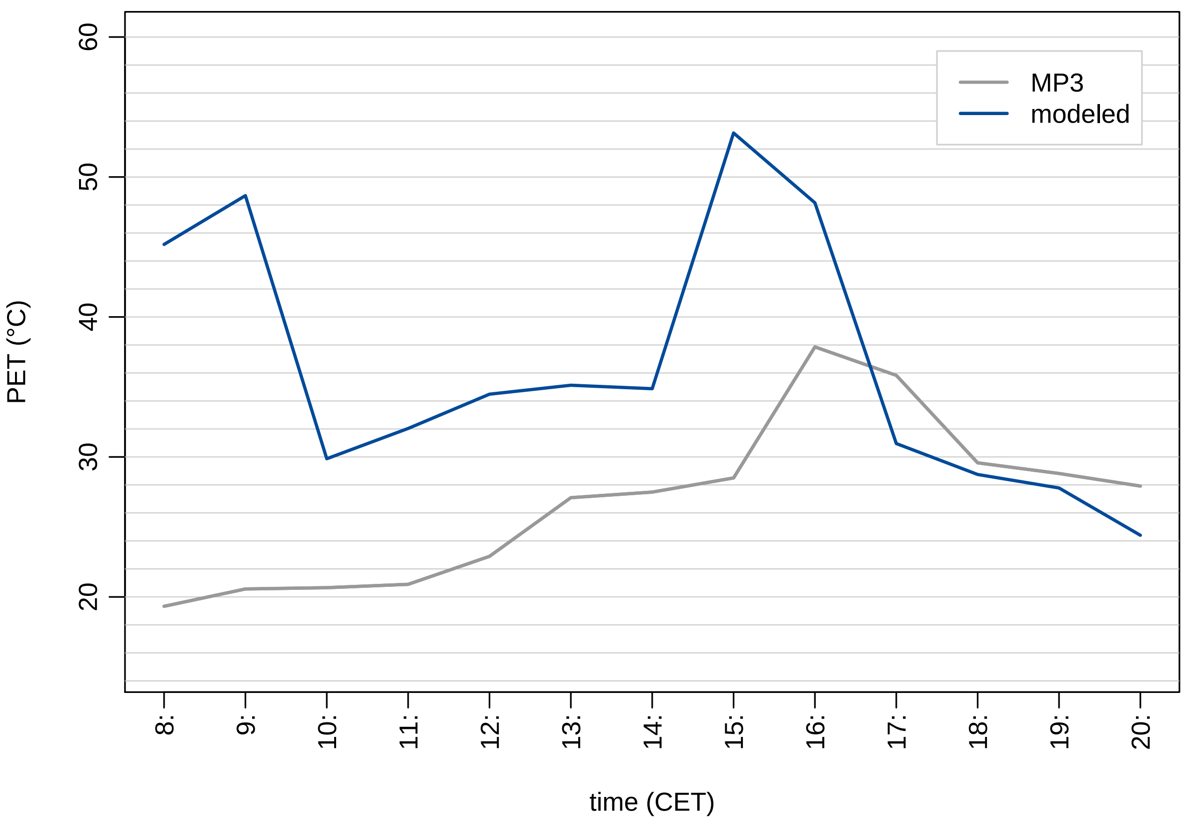


**fig. 10.17:** Comparison of Physiologically Equivalent Temperature (PET) on the place of the old synagogue on the 01st of July 2008 as as calculated based on measurements by a mobile biometeorological station at measuring point 3 (MP3) (grey) and calculated by the SkyHelios model (blue).

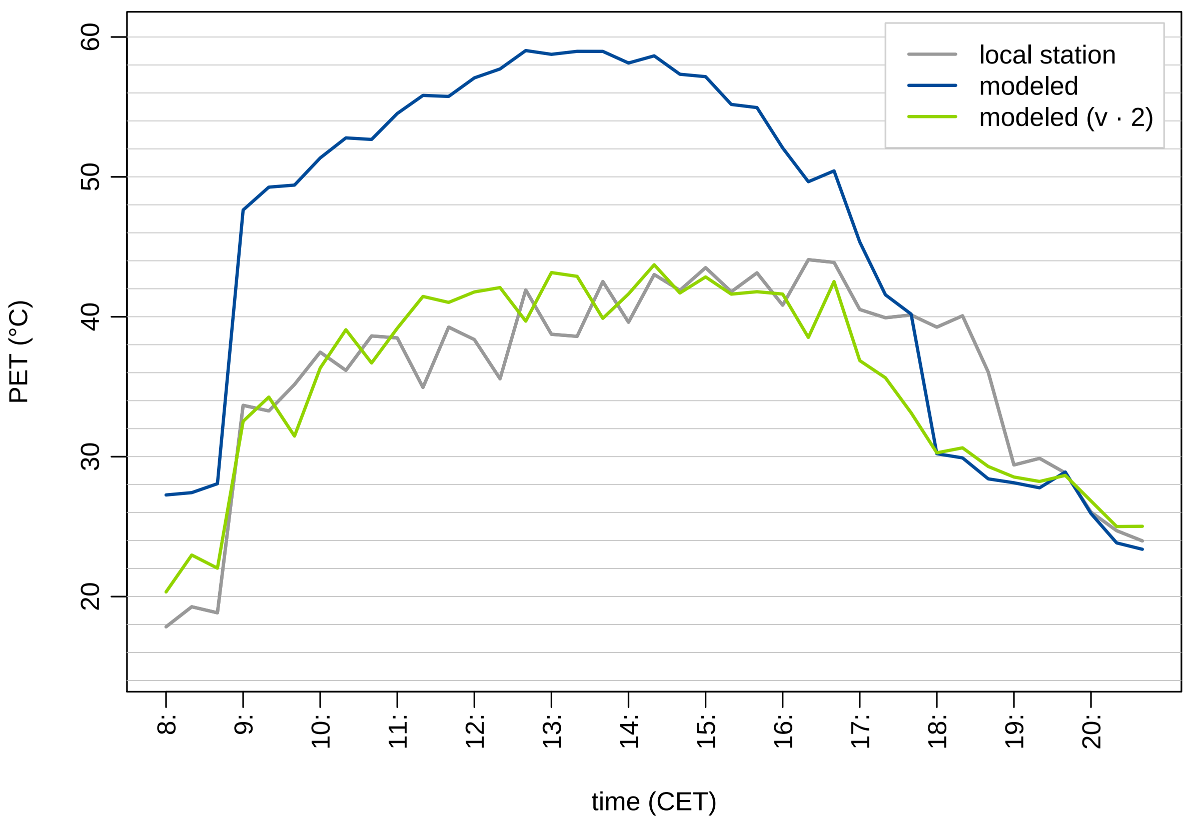


**fig. 10.18:** Comparison of Physiologically Equivalent Temperature (PET) on the place of the old synagogue on the 01$^{st}$ of July 2008 as calculated based on measurements by a local biometeorological station (grey) and calculated by the SkyHelios model (blue). PET calculated by the model with v corrected by factor 2 is depicted in green.

### 10.3.5 Perceived Temperature

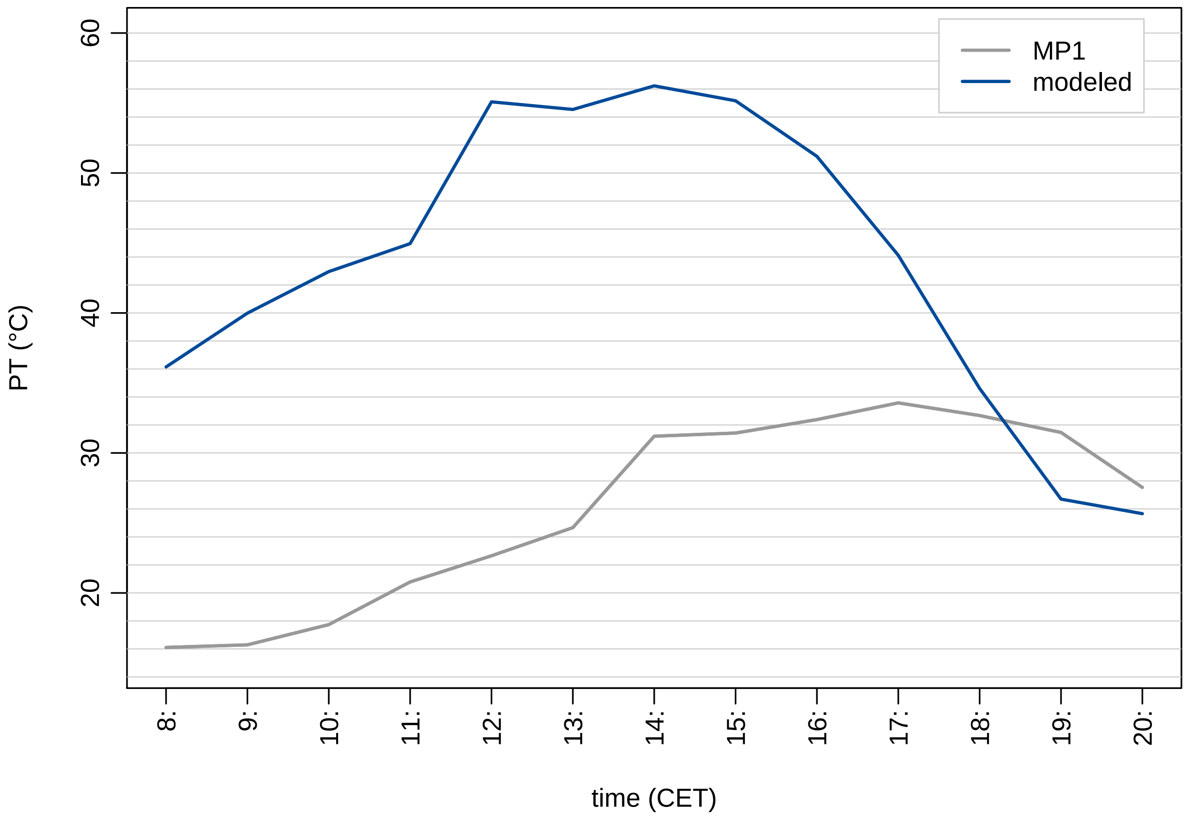


**fig. 10.19:** Comparison of Perceived Temperature (PT) on the place of the old synagogue on the 01$^{st}$ of July 2008 as as calculated based on measurements by a mobile biometeorological station at measuring point 1 (MP1) (grey) and calculated by the SkyHelios model (blue).

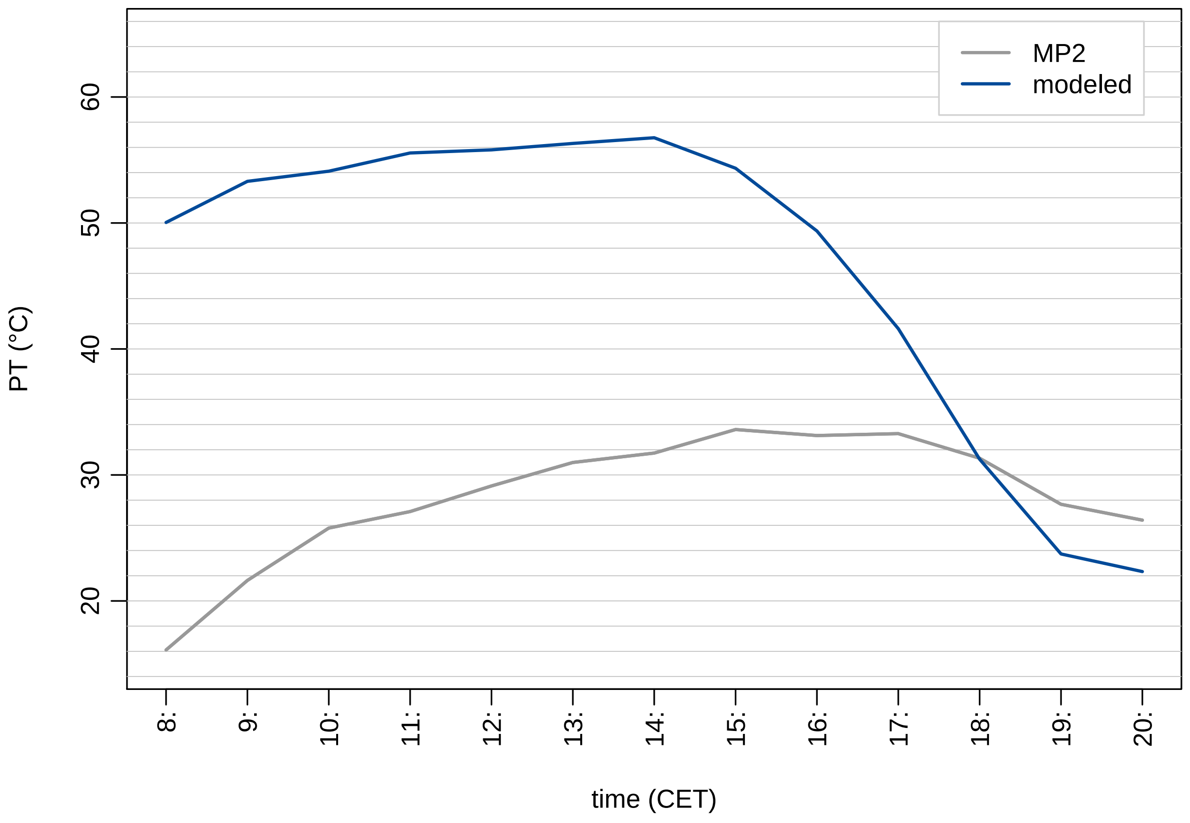


**fig. 10.20:** Comparison of Perceived Temperature (PT) on the place of the old synagogue on the 01st of July 2008 as as calculated based on measurements by a mobile biometeorological station at measuring point 2 (MP2) (grey) and calculated by the SkyHelios model (blue).

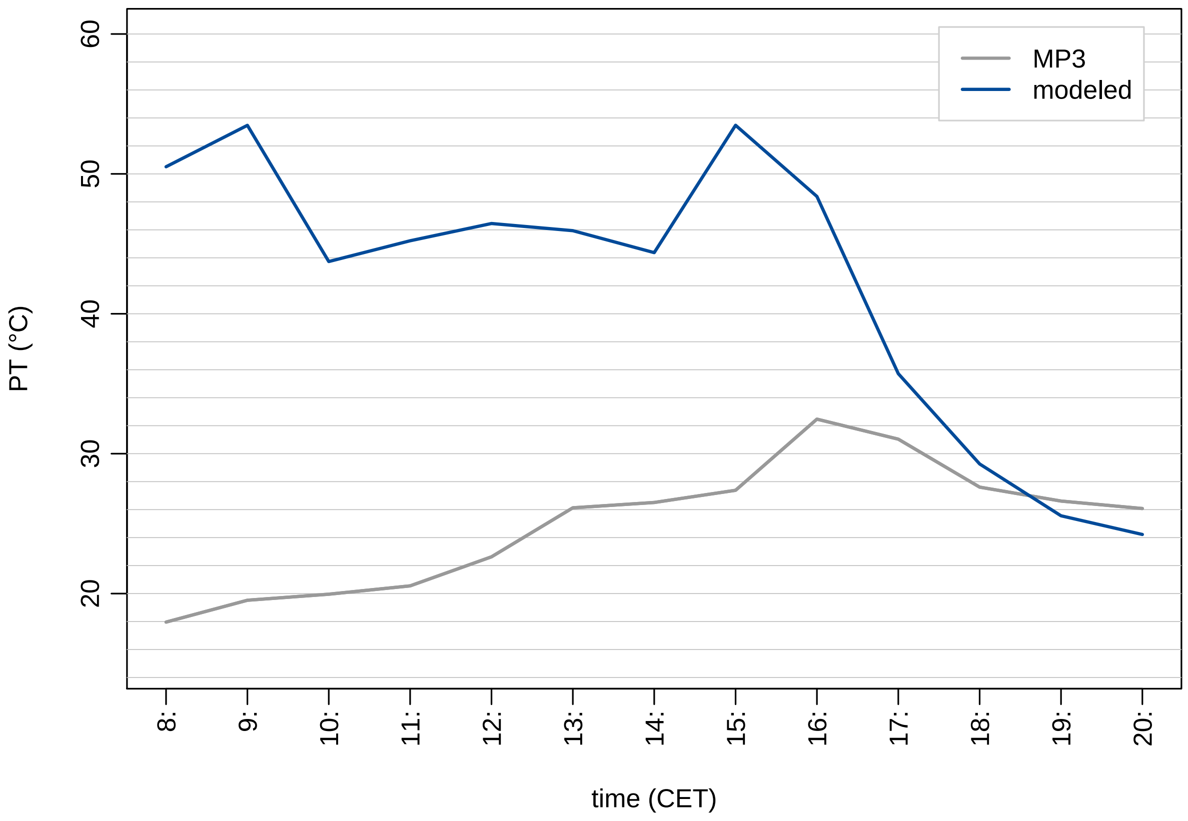


**fig. 10.21:** Comparison of Perceived Temperature (PT) on the place of the old synagogue on the 01*st* of July 2008 as as calculated based on measurements by a mobile biometeorological station at measuring point 3 (MP3) (grey) and calculated by the SkyHelios model (blue).

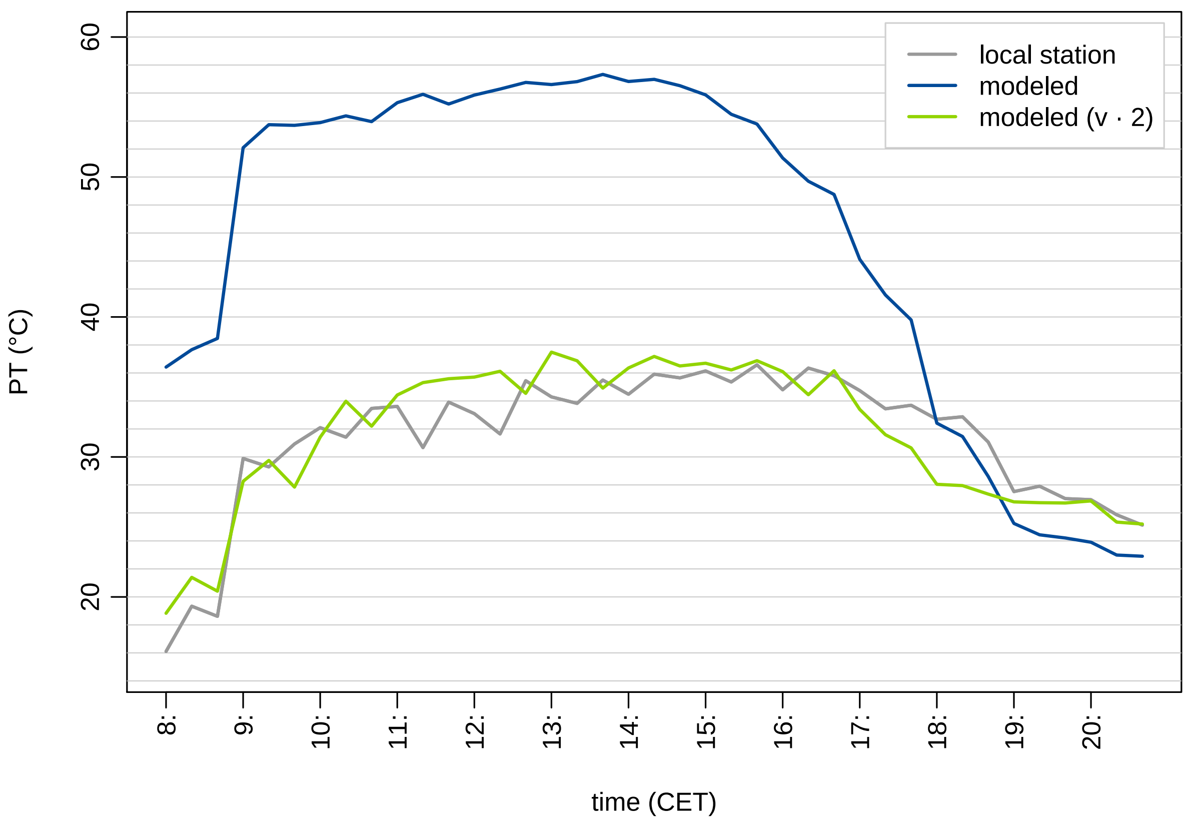


**fig. 10.22:** Comparison of Perceived Temperature (PET) on the place of the old synagogue on the 01$^{st}$ of July 2008 as calculated based on measurements by a local biometeorological station (grey) and calculated by the SkyHelios model (blue). PT calculated by the model with v corrected by factor 2 is depicted in green.

# Erklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Insbesondere habe ich hierfür nicht die entgeltliche Hilfe von Vermittlungs- bzw. Beratungsdiensten (Promotionsberaterin / -berater oder anderer Helferinnen / Helfer) in Anspruch genommen.

Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form einer anderen Prüfungsbehörde vorgelegt.

Die Dissertation wurde in keiner Form bereits anderweitig als Prüfungsarbeit verwendet oder einer anderen Fakultät als Dissertation vorgelegt. Es gab keine früheren Promotionen oder Promotionsversuche.

Ort, Datum Unterschrift