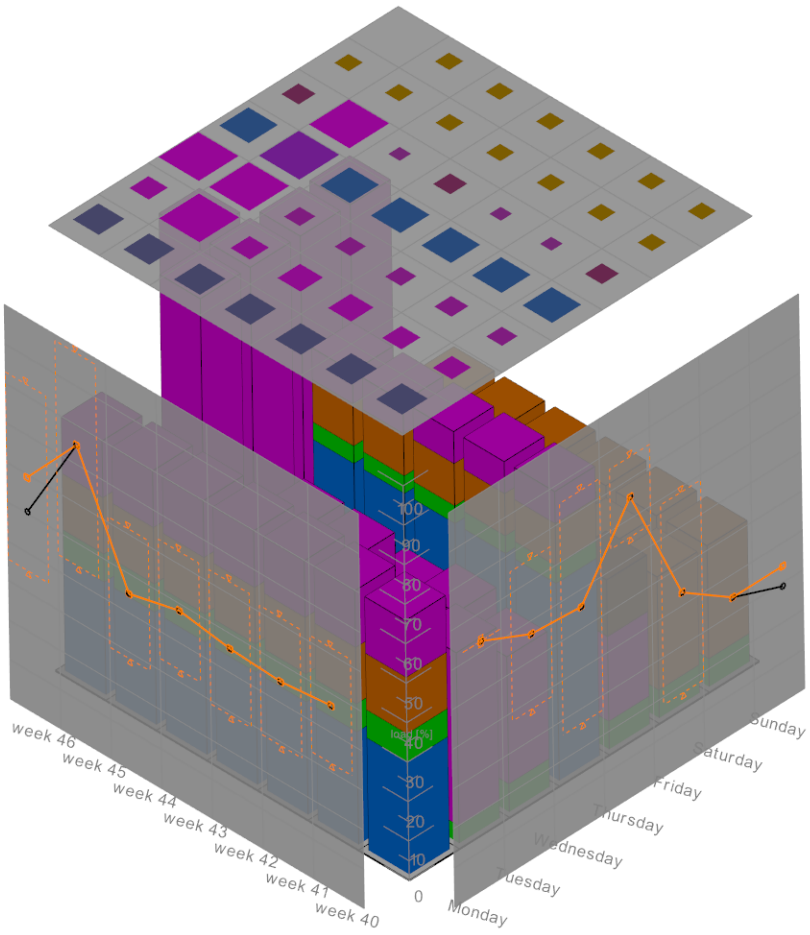
Philipp Hartl

Visualization of Calendar Data

Diploma Thesis



supervised by

Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Eduard Gröller

Dipl.-Ing. Dr.techn. Matej Mlejnek

Institute of Computer Graphics and Algorithms

Vienna University of Technology

Abstract

Since many centuries, calendars are used to organize appointments, events and tasks. In this thesis interaction and visualization techniques for calendar data are presented, which do not only support the organization and analysis, but also facilitate and improve them. The basis of the solution is a 3D heightfield visualization, which displays the workloads of time slots over periods of time in a compact manner. Thereon, interaction and visualization techniques are used to investigate the data set for regularities and irregularities. The comparison of data sets while planning events is just as important as the integration of fuzzy tasks into one's schedule and their manipulation. The visualization and exploration process is completed by statistical representations showing trends and patterns. The use of these techniques and the combination of them are presented with the help of real examples.

Kurzfassung

Seit Anbeginn der Zeit werden Kalender benutzt, um Aufgaben und Termine zu organisieren. In dieser Diplomarbeit werden Interaktions- und Visualisierungstechniken für Kalenderdaten präsentiert, die die Organisation und Analyse nicht nur unterstützen sondern auch erleichtern und verbessern. Die Lösung basiert auf einer 3D Höhenfelddarstellung, welche die Auslastungen von Zeitabschnitten über Zeiträume darstellt. Darauf aufbauend kommen Interaktions- und Visualisierungstechniken zum Einsatz, die es ermöglichen, den Datensatz auf Regelmäßigkeiten und Unregelmäßigkeiten hin zu erforschen. Der Vergleich von Datensätzen beim Planen von Terminen ist dabei genau so wichtig wie das Einbringen von zeitlich nicht genau festgelegten Aufgaben in einen Terminplan und deren Manipulation. Abgerundet wird die Visualisierung und Analyse durch Verwendung von statistischen Auswertungen, die Tendenzen und Muster erkennen lassen. Der Einsatz dieser Techniken und deren Zusammenwirken wird anhand von Beispielen präsentiert.

Contents

Chapter 1

Introduction

For a long time, calendars are used to visualize and organize tasks and events. There are many domains, such as timelines (Tufte [32]) that depict large time periods, for example, to visualize the genesis of our earth, project schedules for business-processes, and so on. The origin and history of different calendar systems can be found at Tøndering's website [31], while ''A Walk Through Time'' can be made at the National Institute of Standards and Technology (NIST) [15].

In comparison to paper calendars, electronic calendars provide more flexibility, such as visualization and interaction techniques. These techniques are used to easily create and delete events, build one's own appointment-lists, set reminders for special events and synchronize the schedule with other devices. In the past years, collaborative calendars have experienced a particular popularity. They are used to exchange the information on calendar data with other users for example in order to create invitations to a meeting.

On the other hand default calendar applications show fixed size views with limited visualization techniques. The one-dimensional non-hierarchical calendar data is usually visualized in static views. These views are neither customizable for personal needs nor for a certain task. For example, one can see a small calendar overview on the left hand side showing only the current selection, while in a main big view, events and tasks can be handled. The time period often can be switched between day-, month- and year-views.

The main task of most common calendar applications is the visualization and organization of events and tasks. It is often possible to set the category of an event, for example, meeting, birthday, and so on. This helps to find events that belong to a special category. However, sometimes the beginning of an appointment depends on the completion of others. Whereas, events of the same category can be found and visualized, there is no possibility, to show dependencies between calendar entries.

Tasks or to-do's are a special kind of appointment. They are always shown in a different view and are not integrated into one's schedule. The reason for this is that they often do not possess well defined time attributes. A task is said to be fuzzy, if it has

at least one missing time attribute, for example start and/or end time. In this thesis a technique to schedule such uncertainties is presented. It is possible to combine tasks and events into one schedule for organization as well as for analysis.

In many calendar applications the analysis of calendar data is currently not supported or very insufficient. However, the exploration and analysis of calendar data is very important in order to get insight into the data and to schedule them more efficiently. In addition to handle dependency and fuzziness, the motivation of this thesis is to develop sophisticated visualization and interaction techniques to represent the calendar data in such a manner, that enables its analysis. A different representation of calendar data is used.

In contrast to common calendar applications, the main focus of this thesis lies on the visualization of and interaction with workloads. The workload represents the total amount of work for a period of time, for example, forty hours a work week. To display the sequence of workloads over large time periods, the time is split into slots and each slot represents a workload. For example, the work week is now represented by five days, with eight hours on each day. The workloads are shown in a 3D heightfield to support the visualization and exploration of calendar data. This basic visual metaphor, the heightfield, is used among others, to simplify the recognition of workload distribution, to find structures in the data and to schedule fuzzy entries. A prototype has been developed to identify the limitations and to find opportunities in the representation of workloads.

This diploma thesis is structured as follows: chapter 2 reviews the state of the art and related work about visualization of time-oriented data and calendar software applications. In chapter 3, heightfield visualization of calendar data as workload distributions is explained. The usage of color for distinction between categories, showing detailed information of abstract data and the representation of groups and dependencies is also presented. Chapter 4 describes novel interaction possibilities on the heightfield. This includes: highlighting of interested objects, an importance visualization view of occluded objects and a sophisticated technique to schedule fuzzy entries. Chapter 5 focuses on advanced statistic views to get more insight into the calendar data, to recognize patterns and trends. Chapter 6 introduces the implementation of the prototype for this thesis and frameworks that have been used. In chapter 7, the major components are tested with real examples and the results are presented. Chapter 8 gives a summary of the completed work and finally, in section 9, the conclusion is drawn and some examples for future work are provided.

Chapter 2

State of the Art

The intention of this thesis is to present new interaction methods and visualization techniques for calendar visualization. Many techniques and possibilities are available nowadays and have already been published. The following sections will give an overview of visualization techniques and applications interesting for calendar visualization.

First, the term information visualization is defined. Then, some techniques to visualize time-oriented data are described. Calendar data are a subset of time-oriented data. Therefore, the superset is explained first, to understand the structure of the data and what is important for visualization. Current software applications for visualizing calendar data are introduced afterwards. The last section discusses the main features of the presented applications and techniques. The results of the discussion are some open issues.

2.1 Information Visualization

The Oxford English Dictionary (1989) defines ''to visualize'' as ''to form a mental vision, image, or picture of (something not visible or present to the sight, or of an abstraction); to make visible to the mind or imagination''. Visualization is a cognitive process and its goal is to give a user a specific insight into some data. In the field of computer graphics, visualization is amplified by the use of computer-based, interactive, visual representations of data [8].

Information visualization, in short InfoVis, is a special type of visualization and is used when abstract, non-physical data should be examined. The main challenges of InfoVis are to find a graphical representation for n-dimensional data, often known as visual metaphor, and to develop an appropriate user interaction for exploring and manipulating the data. Both should help to understand, present, and analyze the data. For example to find structures, patterns, outliers and more. There are many techniques in InfoVis to visualize and to interact with different types of data [29].

2.2 Visualization of Time-Oriented Data

Visualization of time-oriented data is a challenging task. A lot of methods and tools exist to achieve this goal. Aigner et al. [1] specify three aspects to visualize time-oriented data. These are

1. the characteristic of the temporal structure
2. the data itself, which relate to time
3. the visual representation of time-oriented data

First, the data relate in some way to time. While some data depend on points in time, which means that they are instantaneous in time, others have an extent in time and relate to a period. Furthermore, the relation between the data and the time can be linear from the past to the future or in a cyclic manner, for example recurring each day.

Second, the context or the nature of the data has to be considered. Time-oriented data can be acquired in an abstract or a spatial context. This distinction is important for processing the data. While information visualization deals with abstract, non-physical data, scientific visualization deals with real world measurements. Furthermore, the time-oriented data can have more than one attribute or variable. The number of variables defines the dimensionality of the data, from one dimensional data, i.e., univariate data, to multi dimensional data, i.e., multivariate data. Often, not all variables can be represented at the same time. If large data sets with many variables have to be visualized, the use of abstraction is inevitable, in contradiction to the principle of Tufte: ''above all else show the data'' [32].

Third, finding an appropriate visual representation of time-oriented data depends on several factors. The nature of the presentation can be distinguished in static and dynamic representations. For example, a dynamical system, such as a flow visualization [25], can be presented in still images, but it is often analyzed if the visualization is animated. In contrast, static systems, such as calendar applications, use static representations to concentrate on the schedule. Furthermore, the space of representation, such as 2D and 3D, offers different visualization possibilities. For example, a 2D visualization can be extended by one more dimension, to a 3D visualization, to encode another information. As a consequence, advanced interaction techniques must be developed.

These three aspects, with further sub-aspects, determine the visualization possibilities. Often, compromises must be made. In the next sub-sections a brief overview of sophisticated techniques to visualize time-oriented data is given. The first four examples of applications and techniques deal with calendar-data, a subset of time-oriented data. Then time-series on spirals are introduced. Clustered data with calendar based

visualization is shown next. Afterwards, two project management visualizations are presented. At the end a short summary on visualizing time-oriented data is given.

2.2.1 DateLens

Many people need a small portable calendar outside their home or office, as they do not want to carry along their big desktop calendars. Applications for mobile devices such as PDAs have gained their popularity in the last couple of years. Bederson et al. [5] developed an information visualization technique called DateLens, a fisheye calendar interface for PDAs. The design goal was to scale down calendars to smaller devices such as mobile assistants and scale up to larger devices such as desktop displays by using the fisheye distortion technique. An easy navigation and a good representation on handheld devices are very important, because there is not enough screen space to show all the information. Therefore, DateLens offers several different views, to show the schedule, while recognizing patterns and outliers, when searching for some information (see figure 2.1(b)). This is done with semantic aspects, it decides automatically which view is the best to fit into the available screen space and how many information of the calendar data is shown (see figure 2.1(d)). Each row in the view shows one week, while each column represents days of the week. Furthermore, a selection, of how many rows are shown simultaneously (see figure 2.1(a)), can also be done, to show more or less detailed information. The blue boxes in each picture represent the schedule of each day. By tapping a cell, another detailed view expands (see figure 2.1(c)). The transitions between these views are animated. These compact overviews assist the user in planning and analyzing the schedule.

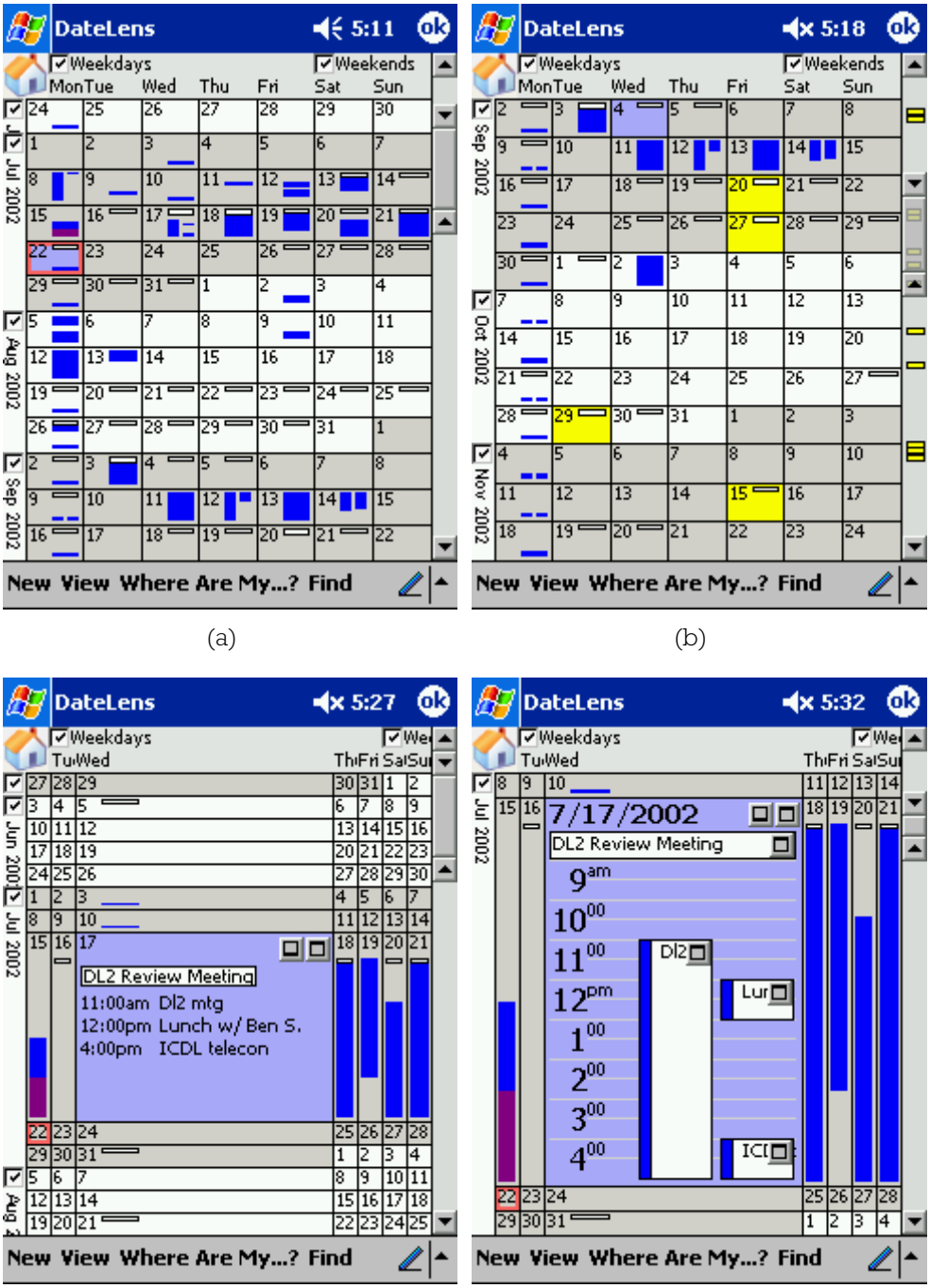


Figure 2.1: DateLens: A Fisheye Calendar Interface for PDAs: (a) overview of more than one month, (b) recurring or searched events are highlighted, (c) zoomed view into one day, (d) more detailed view on that day [5].

2.2.2 Perspective Wall

The Perspective Wall is a focus+context visualization technique. This means, it shows an overview (i.e., context) and detail information (i.e., focus) simultaneously, while both views are combined within a single, dynamic display [8]. The Perspective Wall supports efficient space utilization for 2D layouts of linear data with wide aspect ratios. This is possible through folding a 2D layout and projecting it onto a 3D wall. The technique results in a distortion view with detailed information on the center plane, while keeping an overview on the side walls [20].

The Perspective Wall is typically used for the visualization of time lines. The time is mapped to the horizontal dimension (x-axis) and the categorical data is defined through the vertical dimension (y-axis). The ratio of detail to context as well as the position in time can be intuitively adjusted by user interaction with smooth animation. Figure 2.2 shows an example of the demo application "TimeWall" from Business Objects [4], making use of the Perspective Wall.

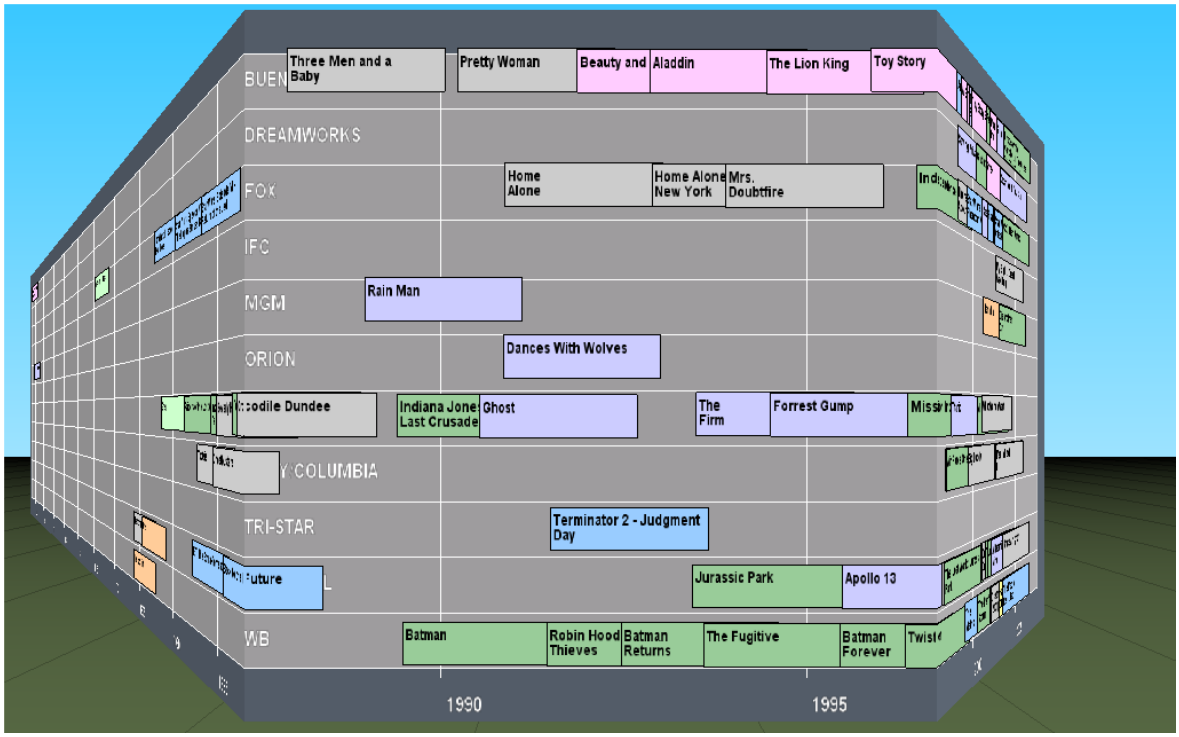


Figure 2.2: Time line visualization realized with a Perspective Wall implementation showing a movie database [17].

2.2.3 Spiral Calendar

Mackinlay et al. [21] developed another focus+context technique by placing objects in an 3D spiral. The time is treated in a hierarchical manner to visualize the calendar data. In figure 2.3, an example of a Spiral Calendar is shown. Each level of granularity partitions the time into more detailed resolutions, i.e., from year to months, from months to weeks and so on. A time section can be expanded into a more detailed view, by user interaction, while the hierarchy of the calendar data (context) spirals into the background. The different levels are visually linked by translucent truncated pyramids, to emphasize the time walk. The Spiral Calendar was designed for the manipulation of one individual's daily schedule. Although the spiral technique makes efficient use of screen space, it is a very time consuming task requiring to walk through each view. Therefore, it is not optimized for navigating to next calendar entries [9].

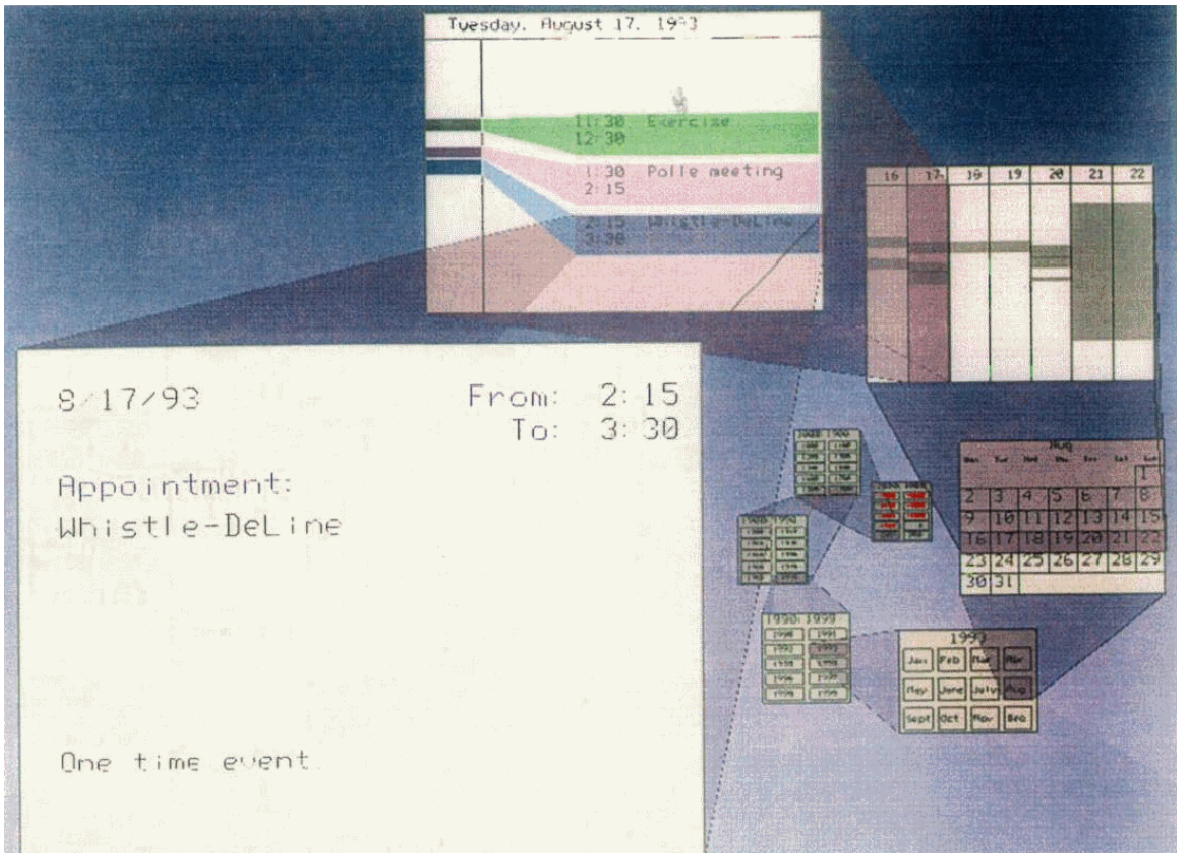


Figure 2.3: A Spiral Calendar with an expanded view of a day schedule [21].

2.2.4 Time Lattice

During the development of calendar visualization techniques for the "Information Visualizer" [21], a tool for visualizing group calendars has been produced. Most electronic calendars have a 2D layout, as they are often based on existing paper versions. A 2D week view is normally represented by days and hours on the horizontal and vertical axis. Respectively, if several week calendars are put on top of each other, a 3D object arises. This is what Time Lattice [21] does. It is a combination of weekly calendars of a group of persons to align them into a 3D object for comparison. Figure 2.4 shows an example of the Time Lattice calendar. The depth axis represents individuals, which are colored randomly for distinction. Each box represents a scheduled event. A shadow metaphor is used to support the interaction. With shadows it is possible to project data in each direction of the 3D object more or less efficiently onto a wall. Each of these three shadow projections interprets the data in a different manner. The left wall represents the individual calendars to see time slots for scheduling meetings. The right wall shows how busy a person is at a particular time within a week. The shadow on the floor stands for the daily workload.

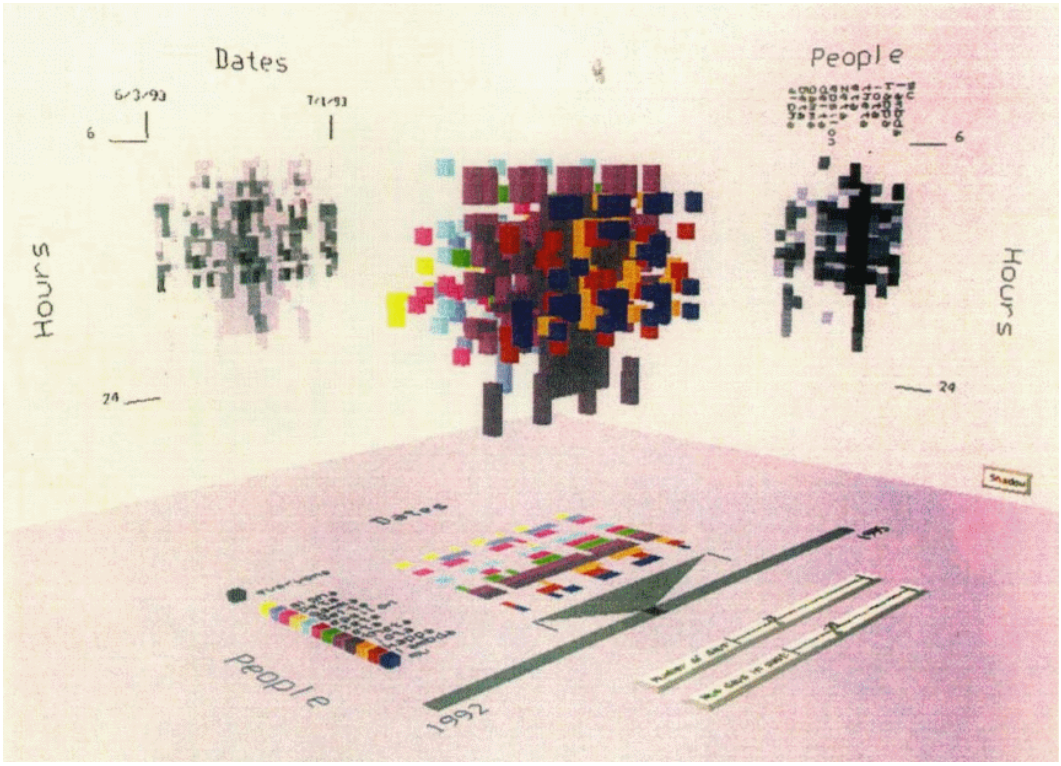
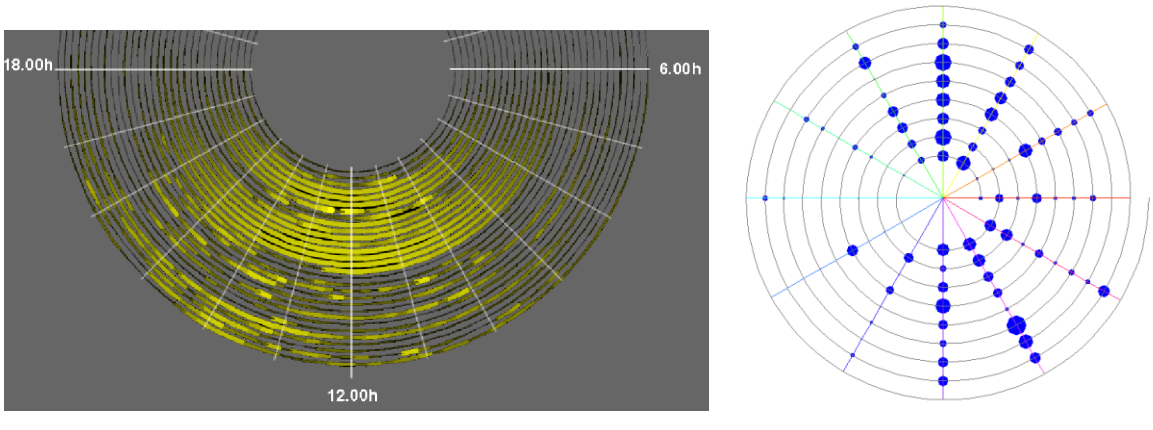


Figure 2.4: The Time Lattice represents week schedules of several people in a complex 3D object. Shadows project the data on walls [21].

2.2.5 Time-Series on Spirals

Time-series data are a special kind of time-oriented data, they may appear in a cyclic manner. Weber et al. [39] presented a method to use spirals, for analyzing and confirming periodic behavior of time-series data. Multiple data sets are compared simultaneously for data points and cycles, see figure 2.5(a). The data is mapped onto the spirals by using colors and line thickness. The visualization of abstract data is also possible. Therefore a similar visualization metaphor is used. Large data sets are visualized in a 3D helix, extended by zooming. The 2D and 3D views are linked, hence the selection of the data in the helix is then visualized in the 2D spiral.

Another spiral technique, to explore serial and periodic attributes of time-series data simultaneously, is described by Carlis and Konstan [10]. The time-series data are displayed on two directions of expansion. The time serial information of the data is displayed on the spiral, while the periodic data is placed radially. The data can be explored in 2D as well as in 3D space. The figure 2.5(b) shows an example of monthly food consumption of a special sort of food by chimpanzees during the period 1980 till 1988. One lap of the spiral is equivalent to one year and each spoke equals one month. The size of a blue blot corresponds the food consumption during a month. The extension in 3D space shows many categories of food at the same time. Each category lies in its own z-range, for example ordered alphabetically.



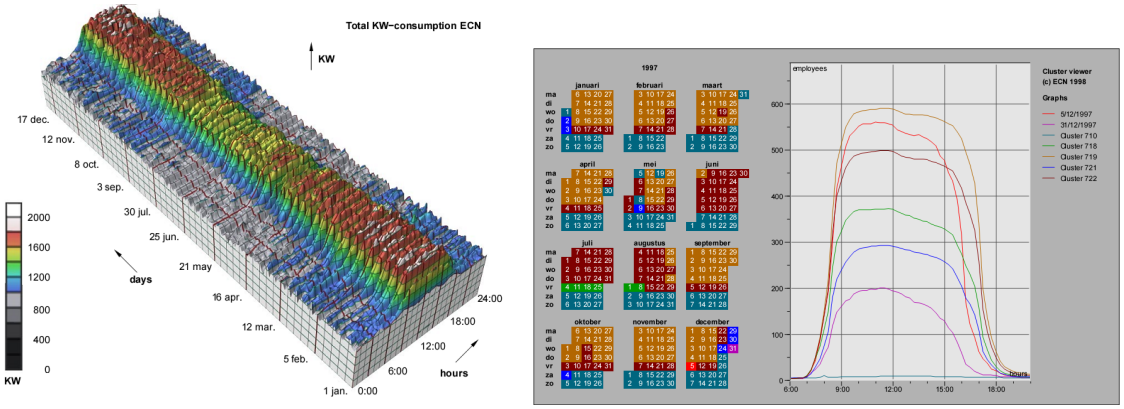
(a) (b)

Figure 2.5: Visualization of time-series data on spirals: (a) identification of periodic behavior [39], (b) monthly food consumption of a special sort of food by chimpanzees during the period 1980-1988 [10].

2.2.6 Cluster

Cluster and calendar based visualization of univariate time-series data has been presented by Wijk and Selow [34]. It is possible to identify patterns and trends on multiple time scales simultaneously. One possibility to show such data is to use the dependency on time scales explicitly. The data is treated in a 2D manner, for example as a function of day and hour. These two time scales are mapped on different axes, while the data is visualized in the third dimension through color. A possible result could be a heightfield (see figure 2.6(a)).

To detect daily patterns and their distribution within a week or year, another more sophisticated approach is used. Similar data are clustered together by any time scale, such as days, weeks and so on. The averaged data out of cluster analysis is visualized in graphs, while the corresponding days are shown in a calendar. The graphs help identifying patterns in more detailed time scales and the corresponding clusters are highlighted by colors on a calendar (see figure 2.6(b)).



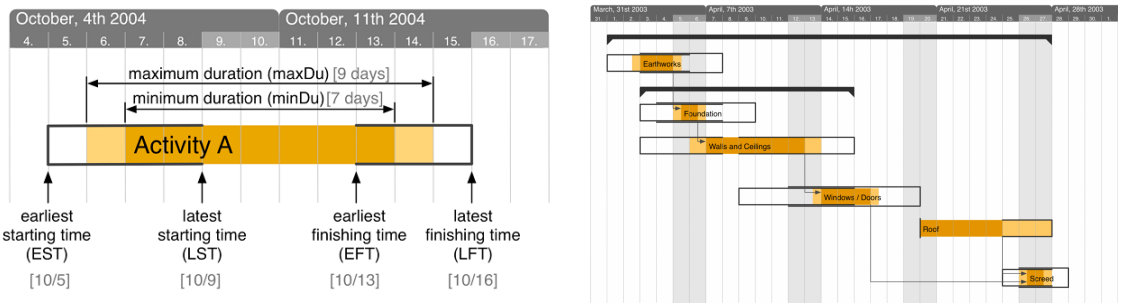
(a) (b)

Figure 2.6: Cluster and Calendar based visualization of time-series data: (a) power demand by ECN displayed as a function of hours and days, (b) calendar view showing the number of employees in clustered graphs [34].

2.2.7 Planning Lines

The key to success in project management is the time domain. Many methods and tools have been developed for project management to deal with resources, deadlines and other challenges. Gantt charts [2], a special type of bar charts, and other typical project management instruments can be used for planning and illustrating project schedules as of bars on a time line. Normally, a Gantt chart does not support temporal uncertainties, which means, that tasks can only be scheduled with exact knowledge of start and end time. A lot of experience is needed to estimate the duration of different tasks, as well as their beginning and completion date.

Aigner et al. [2] presented a visualization technique called PlanningLines, that uses special glyphs (i.e., graphical objects) to represent temporal uncertainties. These glyphs should help to plan and control tasks by defining temporal attributes, such as starting- and ending-time and a possible duration. Each attribute is defined as an interval, so there are six points in time: earliest starting-time, latest starting-time, earliest finishing-time, latest finishing-time, minimum duration and maximum duration (see figure 2.7).



(a) (b)

Figure 2.7: PlanningLines: (a) a glyph showing the time attributes, (b) a small Gantt chart example with PlanningLines [2].

2.2.8 Event Tunnel

The Event Tunnel framework presented by Suntinger et al. [30] offers an interesting visualization technique to support complex business process monitoring and analysis. It provides an interactive visualization of event streams, in distinct multiple views. The visual metaphor is a cylindrical tunnel. There are two important views, the top and the side view (see figure 2.8). The top view is an axial projection of the time line into 2D. To simulate perspective projection the inner circles are displayed smaller than the outer ones. This is a simple but efficient focus+context technique to represent recent events with bigger circles at the outer rings. A spherical glyph represents an event.

Data attributes are encoded in the design of the glyph, such as color, size, position and many more. Many different policies for the placement of these event circles as well as a clustering mechanism on different data attributes for large data sets exist. Dependencies of process behavior patterns and relationships can be easily recognized. To estimate absolute time values the side view provides an adequate representation. The main focus is on the visualization of temporal relationships between correlated event sequences, therefore, they are plotted on horizontal lines in temporal order.

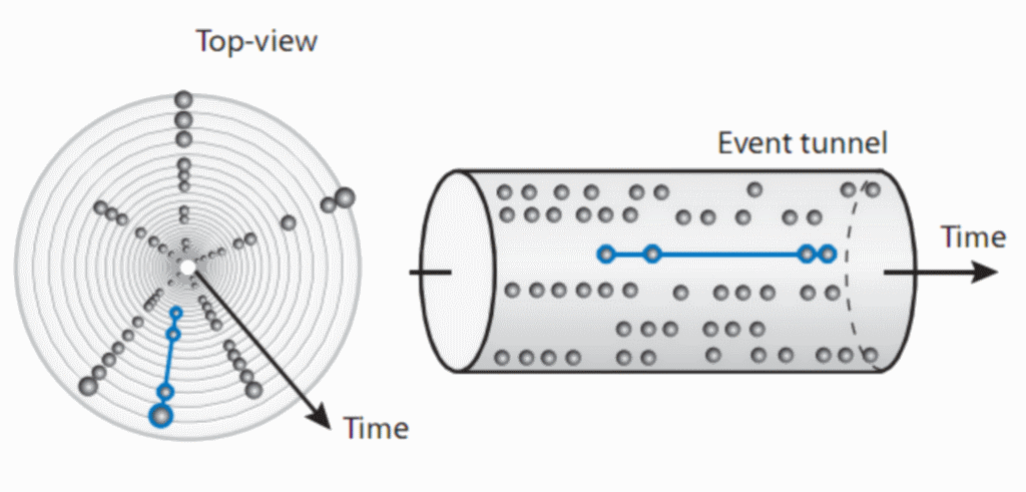


Figure 2.8: Schematic Event Tunnel visualization with top- and side-view. Process sequences are drawn as lines and correlations are connected [30].

2.2.9 Summary

The visualization of time-oriented data is still a challenge in information visualization. There are many solutions for different application areas. All of them use interaction and visualization techniques on familiar objects, to help thinking about the data by means of a real world example.

Information visualization techniques, such as focus+context and others, can help to visualize large periods of time-oriented data as well as calendar data. Different views show different representations of the data and therefore, give an insight into them. Calendar data expand usually linearly, however, daily or seasonal patterns are almost not visible. Therefore, spirals are used to analyze serial and periodic behavior. The visualization of multivariate data can be supported by the usage of color, sometimes in combination with abstraction of the data through clustering. The representation of time-oriented data in 3D space is also used to compare several data or to explore large data sets. Finding relationships and other dependencies are important for calendar data in 2D as well as in 3D visualizations. To recognize patterns and trends, analyzing techniques to explore the data and their temporal structure are used.

2.3 Software Applications

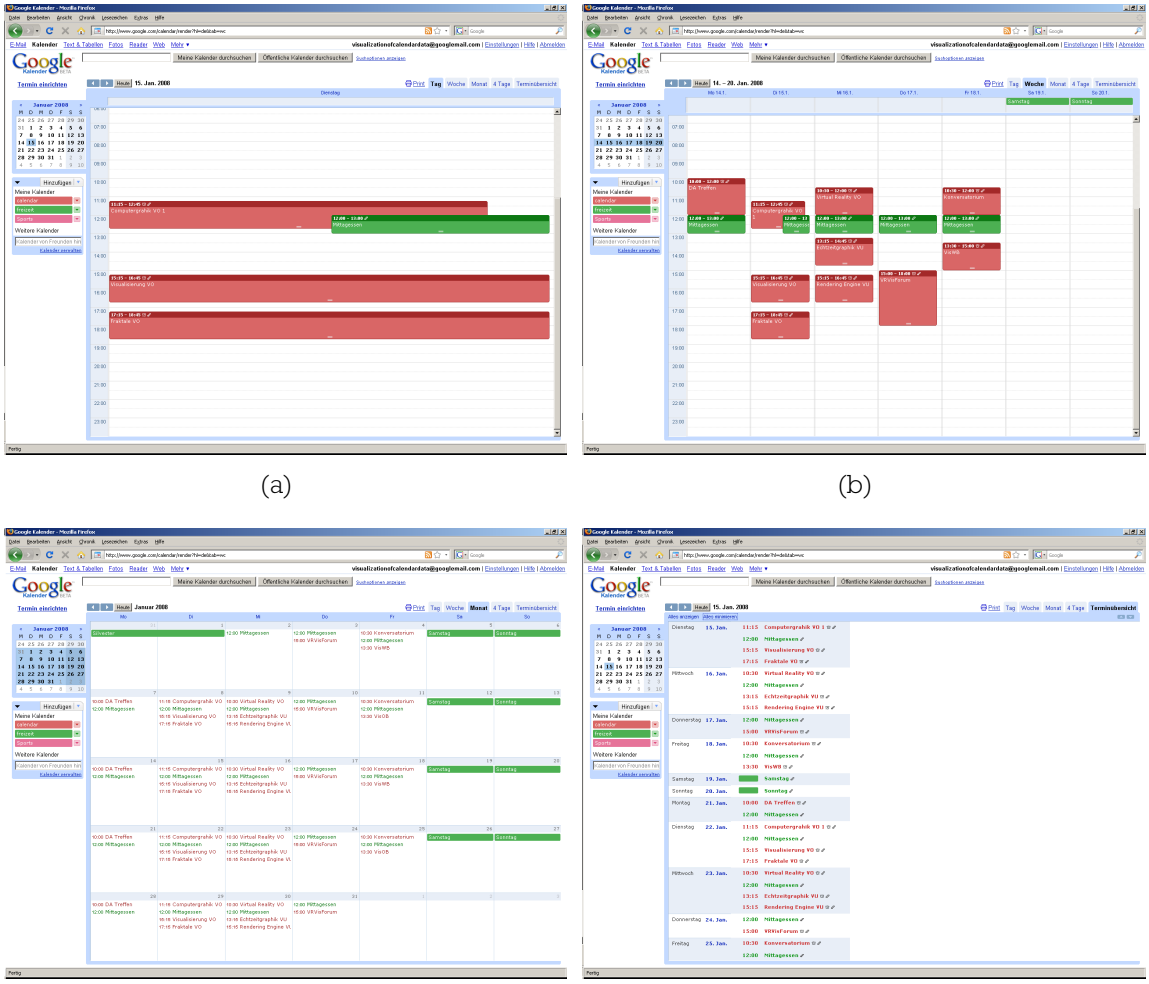
Nowadays, calendar software applications are widespread. Many devices have the ability to show a calendar and to synchronize ones schedule. With software applications, one is able to organize and visualize much faster than with a paper calendar. Electronic calendar applications provide much more flexibility than printed ones. They all offer many visualization and interaction techniques, obviously to easily create and delete events and tasks. It is possible to build to-do lists. Reminders for special events, which should not be forgotten, can also be set. Moreover, sharing of calendars with other users experiences quite a popularity.

There are many calendar applications on the software market. In the next subsections, some examples, of what calendars currently support, are presented. At the end a short summary is given.

2.3.1 Google

Recently Google [16] has introduced, besides many other online services, also a calendar application. The online calendar should help to manage all activities and appointments, easy and fast, from a central point. The data can be published to the public or to specific groups of people. Events can be discussed with others. Invitations can also be sent.

A small calendar overview is displayed on the left hand side, which shows the current selection and appointments, if there are any. In the main big view (see figure 2.9) the user is able to handle entries, switchable between daily, weekly, monthly and other views. In the daily and weekly views there is a small bar at the top for daily entries. To distinguish more easily between different calendar data, each calendar can be assigned to a different color. Each calendar can easily be shown or hidden by selection. A simple interaction through point and click with the mouse, is implemented. Detailed information of an event is shown by selecting it with the mouse.



(a) (b)

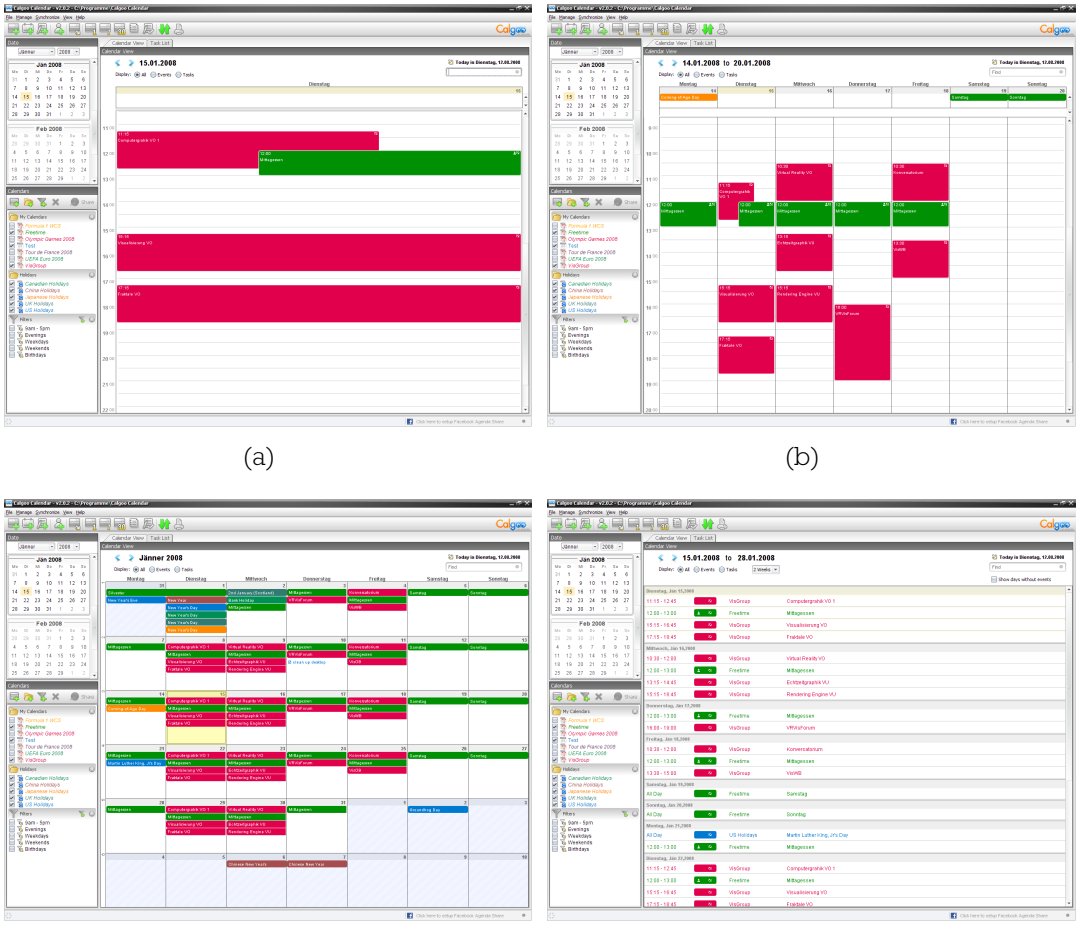
(c) (d)

Figure 2.9: Google calendar views: (a) daily, (b) weekly, (c) monthly, (d) agenda view [16].

2.3.2 Calgoo

The Calgoo Calendar [28] offers the possibility to use Google and other online calendars as a desktop calendar with offline support. It has a very simple user interface, is free and supports many features, for example, the compatibility with other programs and platforms for synchronization.

It offers the common views, such as daily, weekly, monthly and agenda views (see figure 2.10). On the left side the user can see the month overviews and all calendars. Each calendar can have its own color for better recognition and collaboration. There is no detailed information in a view, until an interested event is selected. Also tasks can be created, but they are listed in a separate table view, with priority, status and completeness properties. Filters are a feature, to search for text, times, attendees and defined tags in the calendar data set.



(a) (b)

(c) (d)

Figure 2.10: Calgoo calendar views: (a) daily, (b) weekly, (c) monthly, (d) agenda view [28].

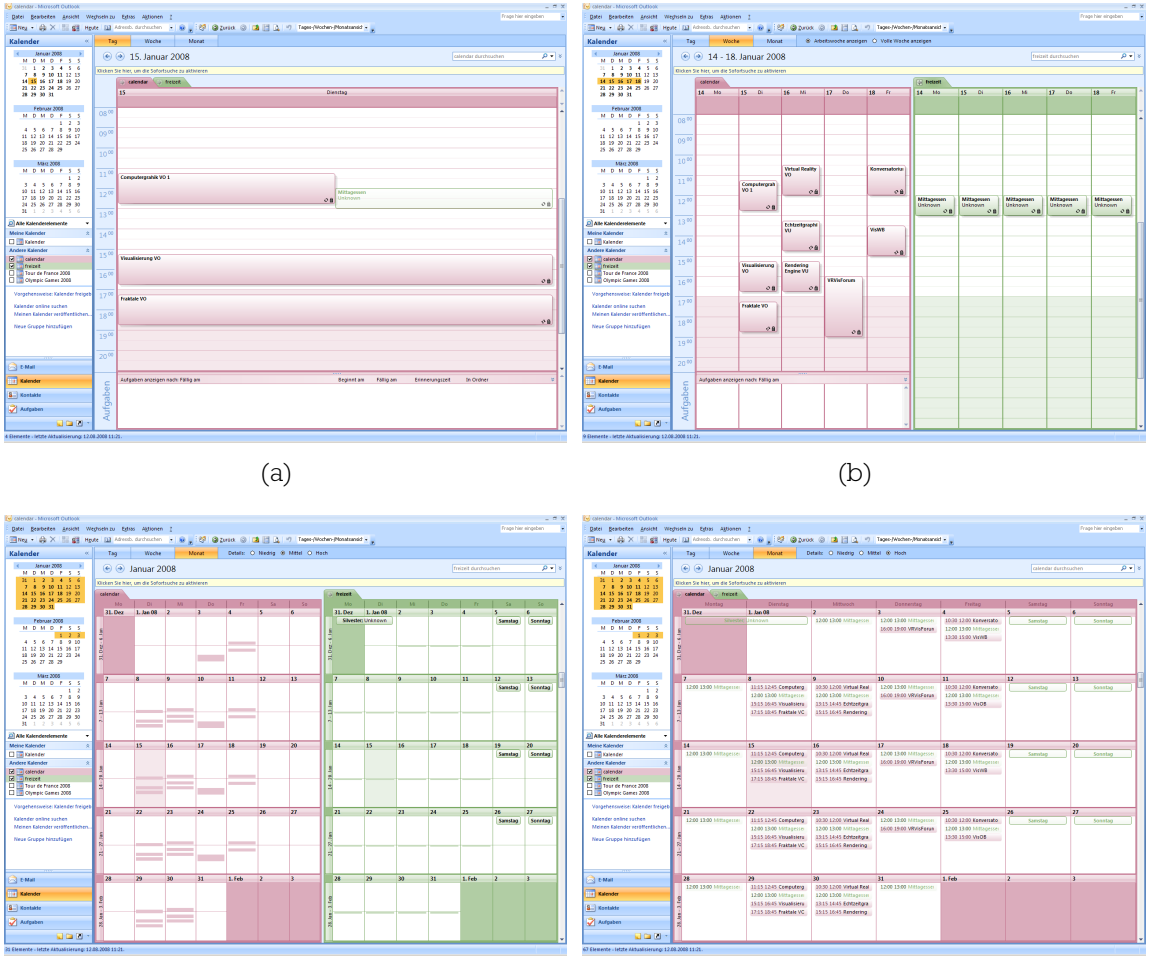
2.3.3 Outlook

Microsoft sells a powerful calendar within its Office Suite as a part of Outlook [23]. It is easy to share calendars and make some appointments in collaboration. Outlook offers all the common views, such as daily, weekly, monthly views and some more (see figure 2.11). Similar to other calendar software, there is an overview as well. Each calendar has its own color, with an opportunity to categorize each calendar entry additionally with a special color. In the weekly view it is possible to switch between a work week (see figure 2.11(b)) and a full week (see figure 2.11(c)) view. Tasks are shown permanently at the bottom of the daily and weekly view (see figure 2.11(a)). The monthly view has three information steps with varying detail. The lower one shows only daily-dates. The middle one is the most interesting view concerning information visualization (see figure 2.11(c)). It shows the scheduling in solid blocks, depending on its start and end time. Each daily cell is divided by a thin line which represents noon. The position and the size of such blocks in each cell helps to imagine the scheduled data. The highest step shows more detailed information, start and end times and the first letters of the summary (see figure 2.11(d)).

The shared calendars are shown side by side in columns, while in the latest version of Outlook (2007), it is also possible to get an overlay view (see figure 2.11(a) and 2.11(d)). In this view all selected calendars are combined into one view. The focused calendar is displayed in solid blocks, while the others are transparent. The boxes are only as big as the available space permits. Sometimes the information is truncated and only visible if the user point at it, to get its details on demand.

2.3.4 Rainlendar

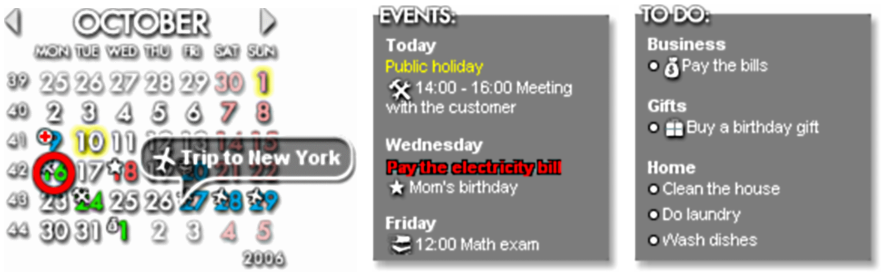
Rainlendar [26] is another desktop calendar, which is not as comprehensive as others but it is highly customizable and easy to use. This widget calendar does not need a lot of space on the desktop, as it shows only short overviews. As a default setting the user gets a monthly overview (see figure 2.12(a)), but also more months can be defined to be visible. In the monthly overview one sees small symbols and colored numbers, which define the category of an event. For example, a star symbolizes a birthday and a plane a trip, to name a few. Upcoming events (see figure 2.12(b)) and to-do's (see figure 2.12(c)) are shown separately in lists, also with the help of symbols and colors for an easier distinction. The creation of tasks is a standard function, but it is also possible to show tasks directly in the calendar.



(a) (b)

(c) (d)

Figure 2.11: Outlook calendar views: (a) daily overlay, (b) weeks side by side, (c) months side by side with middle details, (d) monthly overlay with more details [23].



(a) (b) (c)

Figure 2.12: Rainlendar calendar views: (a) overview, (b) event list, (c) todo list [26].

2.3.5 Summary

Many software calendars are similarly built; they differ only in specific elements. All of them show fixed-size views with limited overview techniques. They use different views to visualize different periods of time. Sometimes it is possible to categorize an event (e.g., work, sports and so on). Colors are often used to distinguish between different calendars. Google does not offer a categorization of events, and, furthermore, it is not possible to define a to-do list. Outlook offers a feature to show several calendars simultaneously for comparison. Tasks are only displayed in external views. Symbols and colors are used in Rainlendar, for fast recognition of similar activities.

2.4 Discussion

Calendar data are time-oriented data and there are many possibilities to visualize them. According to the above sections, displaying large data sets often makes use of information visualization techniques, such as focus+context approaches. Long periods of time can be displayed, with the restriction of analyzing the calendar data for daily or seasonally patterns. The representation of calendar data depends on the application area. An extension of the visualization space from 2D to 3D can be used to compare several data sets simultaneously. The color is used to distinguish between each category of the data set. The 3D space needs advanced interaction techniques to manipulate and explore the data, for example, it is not easy to retrieve detailed information.

Common software applications currently do not make use of sophisticated interaction and visualization techniques. The standard views, for example, do not make use of semantic aspects of any kind (see section 2.2.1). Therefore, the information in the boxes (i.e., showing a daily schedule) is often truncated. Only through additional interaction techniques, the information can be retrieved. Furthermore, tasks are not included into one's schedule, as it is not possible to treat uncertainties.

Calendars are used to manage the data, although the analysis of calendar data is omitted. Relationships and other dependencies are not only crucial for planning systems. There are often activities, which depend on each other. Therefore, it could be interesting to retrieve such information visually. Furthermore, the analysis of calendar data is also important to see patterns or trends. An example is, to see the workload distribution of a month. This can be supported by statistical graph representations of calendar data.

The intention of this thesis is to treat calendar data in a scientific manner, by the support of visualization and interaction techniques. The main issues, about visualization of calendar data in this thesis, are summed up as follows:

1. compact view presenting long periods of time
2. representation of workloads
3. efficient usage of color
4. comparison and analysis of calendar data for a group of persons
5. display of relationships and other dependencies
6. treatment of uncertainties
7. analyze features to see patterns and/or trends
8. include statistics in graphs and/or correlations in a calendar
9. provide interaction techniques for manipulation and exploration

In existing systems some of these elements are partially or fully available, while others are not or in a different context. Table 2.1 summarizes the investigated applications and visualization techniques based on the above listed points.

	long periods	colors	workloads	groups	relationships	uncertainties	patterns/trends	statistics
DateLens	○	○	○	-	+	-	-	-
Perspective Wall	+	+	-	+	-	-	○	-
Spiral Calendar	-	○	-	-	-	-	-	-
Time Lattice	-	+	+	+	○	-	○	-
Time-Series on Spirals	+	○	○	○	-	-	+	-
Cluster	+	+	○	○	-	-	+	+
Planning Lines	+	-	+	-	+	+	○	-
Event Tunnel	+	+	-	+	+	-	+	-
Google, Calgoo	○	○	-	+	-	-	-	-
Outlook	○	+	○	+	-	-	-	-
Rainlendar	○	+	-	+	-	-	-	-

(-) not supported, (○) partially supported, (+) fully supported

Table 2.1: Comparison of the investigated applications and visualization techniques to the main issues of this thesis. All of them offer interaction techniques to manipulate and explore their data.

In the following chapters all of these issues are dealt with. The focus in this thesis lies on the visualization and exploration of calendar data, especially the workloads. Therefore, another representation for visualizing workloads over periods of time is presented first.

Chapter 3

Heightfield Visualization of Calendar Data

The first important step in visualization is to find an appropriate visual mapping of the data, that can provide not only an insight, but also to prepare them for exploration. The main focus is to analyze calendar data, especially the workload distribution. Therefore, an intuitive visualization of the abstract data is described first. The workload for a time slot is treated as a scalar value and is displayed as a height value on a grid. This results in a 3D heightfield showing all workloads for a given period of time. Color is applied to distinguish the workload representation of each time slot in detail. Therefore, another attribute of the calendar data is used. Since the heightfield presents the calendar data in an abstract manner, the retrieval of detail information, such as the activities behind a workload value, is described afterwards. At the end of this chapter, the visualization of groups and dependencies, a technique to focus, for example, on recurring activities, is mentioned.

3.1 Workloads as Height Values

As already mentioned in section 2, there are various possibilities to visualize and analyze calendar data. The core of this thesis is to analyze calendar data by visualizing the workloads of time intervals. In general, calendar data are one dimensional univariate data, though for each activity a vector of data, up to a descriptive information, is associated. The vector is often treated as one abstract object. To visualize the whole data itself is often impossible and makes only sense in few cases. At first, only the duration of an activity is taken into account. To understand how a workload is defined, the duration has to be understood first.

A well formed activity is defined by a start time and an end time. Sometimes, the end time is given by a duration. Duration is the difference between end time and start time. A set of activities can be defined as: $A = \{a \mid a \text{ is an activity with a } duration > 0\}$

fulfilling the equation 3.1. t_{start} and t_{end} are the start/end times of an activity, while a time slot Δt is defined by the interval $[t_1, t_2]$. The duration of an activity in a time slot is therefore bounded by the interval of that time slot.

$$duration(a) = \begin{cases} (t_{end} - t_{start}) & \text{if } t_1 \leq t_{start} \text{ and } t_{end} \leq t_2 \\ (t_{end} - t_1) & \text{if } t_{start} < t_1 \text{ and } t_{end} \leq t_2 \\ (t_2 - t_{start}) & \text{if } t_1 \leq t_{start} \text{ and } t_2 < t_{end} \\ (t_2 - t_1) & \text{if } t_{start} < t_1 \text{ and } t_2 < t_{end} \end{cases} \tag{3.1}$$

The next step is to find all activities for one time slot. All activities are clustered into time slots. The union of all durations is weighted by the size of the time slot. The result is the workload of the time slot (see equation 3.2). Instead of the union of durations other operations can be used. However, the union is able to treat the durations of overlapping activities as one big duration. Otherwise, the durations of overlapping activities would distort the workload result.

$$workload_{\Delta t} = \frac{\bigcup_{i=1}^{n} duration(a_i)}{\Delta t} \tag{3.2}$$

The result of this clustering is an array of intervals and associated workloads. A default visualization of such data are bar charts [29] or time plots [22], but the limit of screen space is reached rapidly for large data sets. Some of the visualization techniques presented in chapter 2 can eliminate these problems.

One way to accommodate large 2D representations is to expand the visualization into 3D. For example, time can be mapped on more than one axis. This results in an efficient utilization of screen space. A solution is presented by Wijk and Selow [34] (see figure 2.6(a)). Each axis can represent a time dimension. For example, weekdays, weeks and hours are mapped onto x, y, z. The interval of one time axis is represented through another time axis, but with a different resolution. The step width between two weeks are always seven days and each day has twenty four hours. The scaling and the visualization of the data can vary, hence there are no restrictions to fixed-size views.

Inspired by Wijk and Selow [34], Mackinlay et al. [21] and others, we visualize the clustered data as a heightfield. A good data representation is a regular grid [25] with equidistant spacing and implicit proximity relationship. One cell corresponds to one time slot (and vice versa) and at first only one height value is stored, i.e., the workload of that time slot in percent as a scalar value. The grid defines the time dimensions and corresponds to the xy-plane, while the workloads extend into the z dimension.

In information visualization, multi-dimensional data is often represented by glyphs (i.e., graphical objects) [38]. They explain different data by their visual properties, such as size, color and others. As we are in 3D space, it is obvious to use a 3D object, which can be seen from all point of views. A cube is simple to visualize and it fits very

well on the grid. If a workload is assigned to a cell, a cuboid is displayed on that cell and the height represents the workload in percent. The maximum height value equals to 100 percent and is defined as the workload limit. If the temporal cells are uniform, the cuboid has a quadratic base (i.e., a square cuboid). To not always specify the geometry of the cuboid, block is used in this thesis as a synonym for cuboid. The size of the base is a little bit smaller than the cell size, so that the blocks can be easily recognized. The boundaries of a block are emphasized by line thickness. The human eye is capable to detect edges very fast [40]. The heightfield is viewed by a parallel projection [14]. Therefore, the objects do not change size and the workloads (i.e., the heights of the blocks) are comparable.

Figure 3.1 shows an example of the implemented prototype. The x-axis defines the first time dimension and represents days while the y-axis represents weeks. With this heightfield representation it is possible to see the workloads of each day and to recognize not so heavily loaded days.

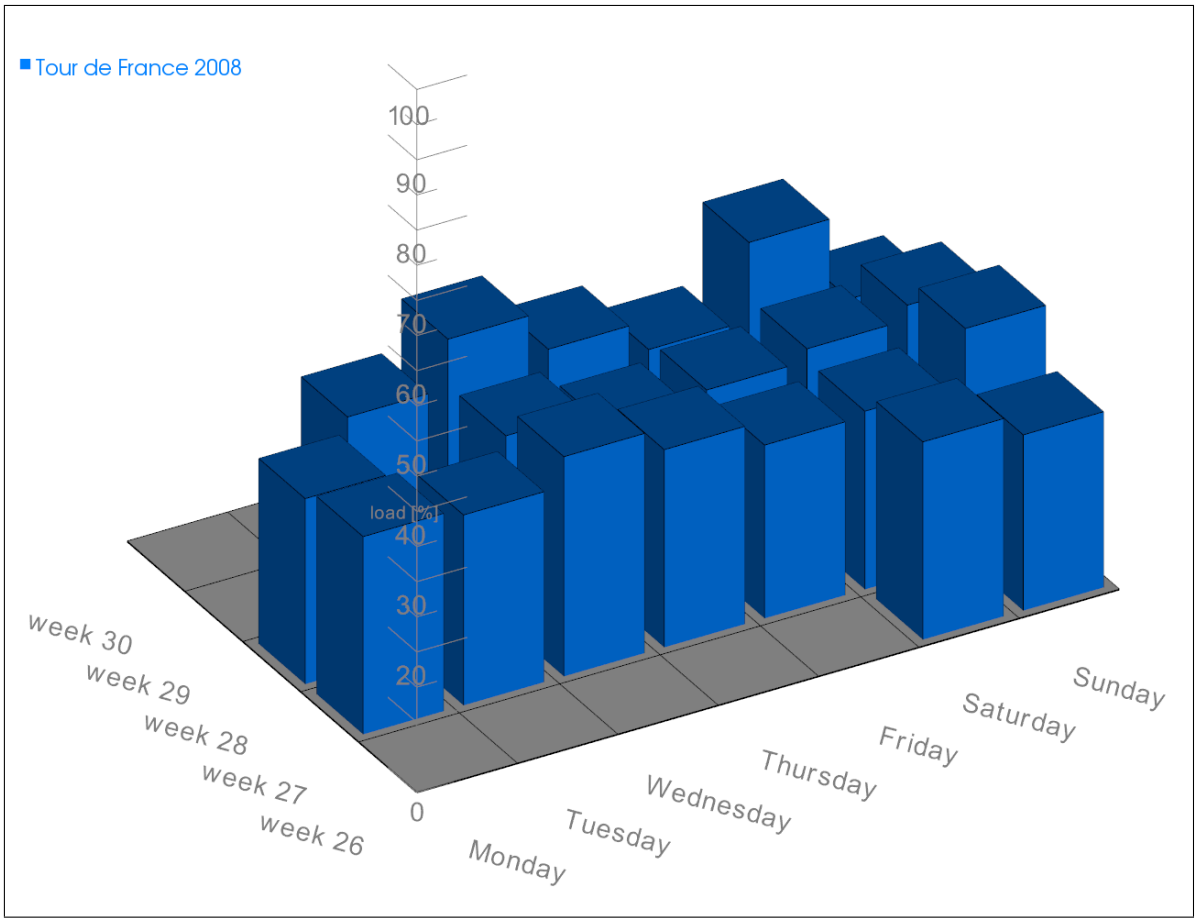


Figure 3.1: Heightfield visualization showing workloads on almost each day in an arbitrary 3D view.

3.2 Color Usage

Calendar data often include many attributes. In addition to temporal characteristics of a data set, the descriptive properties are also important. Moreover a data set can be attached, for example, to a user. Initially the user mapping is not as interesting as the categorization of the workload for one person. This offers two advantages:

- the type of an activity is defined
- activities of the same type are grouped

The visualization of the overall workload per time slot was the first step to be considered. An improvement in graphical exploration of the heightfield data is provided by the visualization of categories. The workload is coded by the size of the glyph. To distinguish between the categories, each of them is assigned to a unique color, either manually or randomly. The correlation between categories and colors is always shown. Coloring is a very important aspect in visualization; the science of colors is described in [40]. The implemented prototype uses the simple RGB color model with alpha-transparency.

There are many possibilities to group calendar data. In this thesis, colored blocks represent groups of activities for a specific time slot. Each group represents another category. To retrieve the different groups, all activities of each time slot are further subdivided by their categories. This allows for example, to easily distinguish between work and family activities. Without clustering of workloads into their categories only one big block would be shown. A workload block is split into parts, representing the categories. The categories are sorted alphabetically. Therefore, the colored parts are drawn in the same order as the categories appear. This helps to identify the included categories of a specific time slot and to compare them with each other. Figure 3.2 shows a 3D heightfield calendar representing the workloads per category.

This feature is also available when more calendars are simultaneously used, i.e., in collaborative systems. The human eye is very well suited to detect minimal variations in color intensity. The Mach band effect [12] is just an example. To distinguish between users and their categories, enough color nuances should be used.

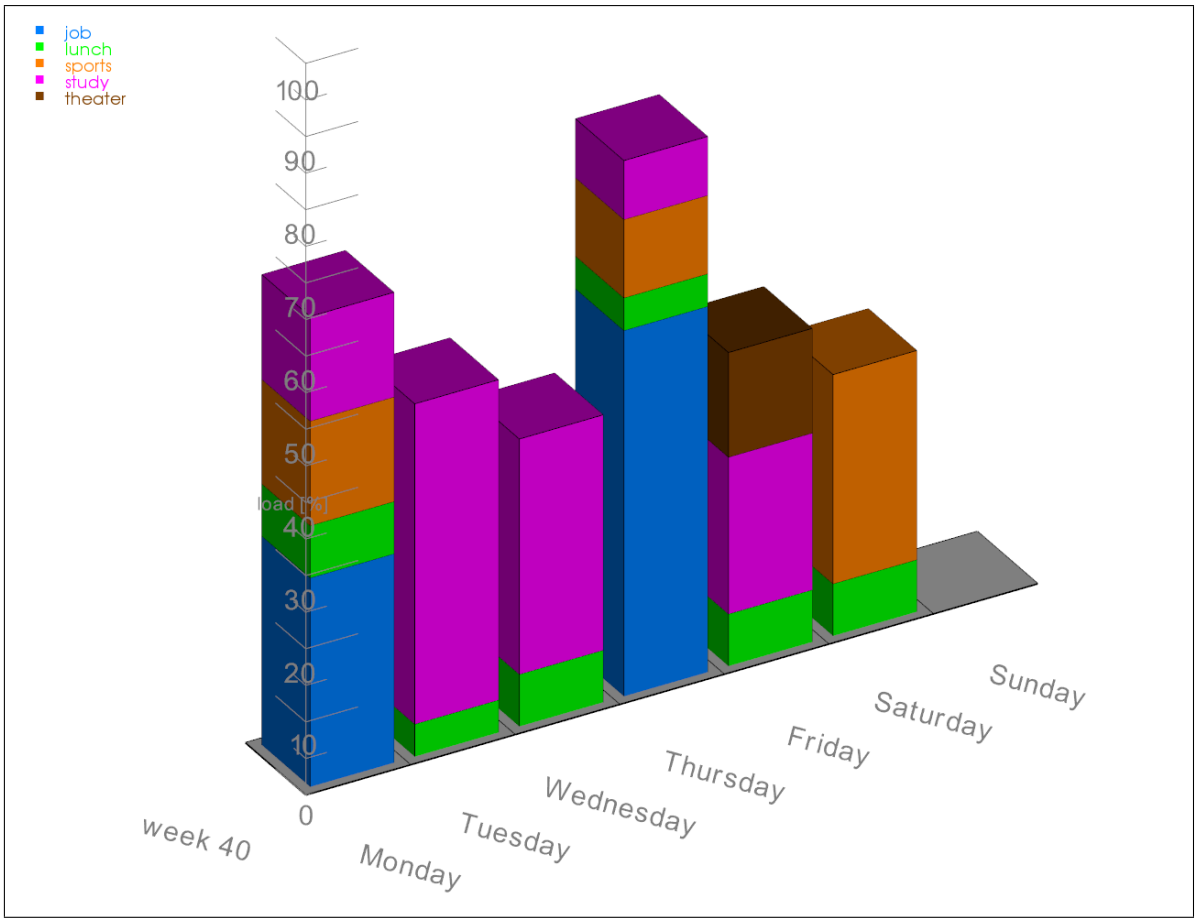


Figure 3.2: Heightfield visualization showing the usage of color to distinguish categories and their workload ratio.

3.3 Detail Information

In many cases, one representation of the complete data set is not enough to show all the information at once. The heightfield is used to present the workloads per time slots over a longer period of time. The color is used to differentiate between the categories. The detail information on which activities are assembled under a specific block is shown on-demand [27] by interacting with the heightfield. The interaction is explained in detail in section 4.1. There are two detail information views available.

The first detail view (see figure 3.3) compactly presents all activities of a time slot in a chronological order in a well arranged overlay [33] overview. For this purpose, the temporal structure and a short summary of each activity are shown at once. Each line in that overlay has some textual information about an activity of a selected block. The time slot information and the overall workload are also displayed. This kind of meta information representation can be shown at any time and from any position by selecting a workload block with a mouse pointer.

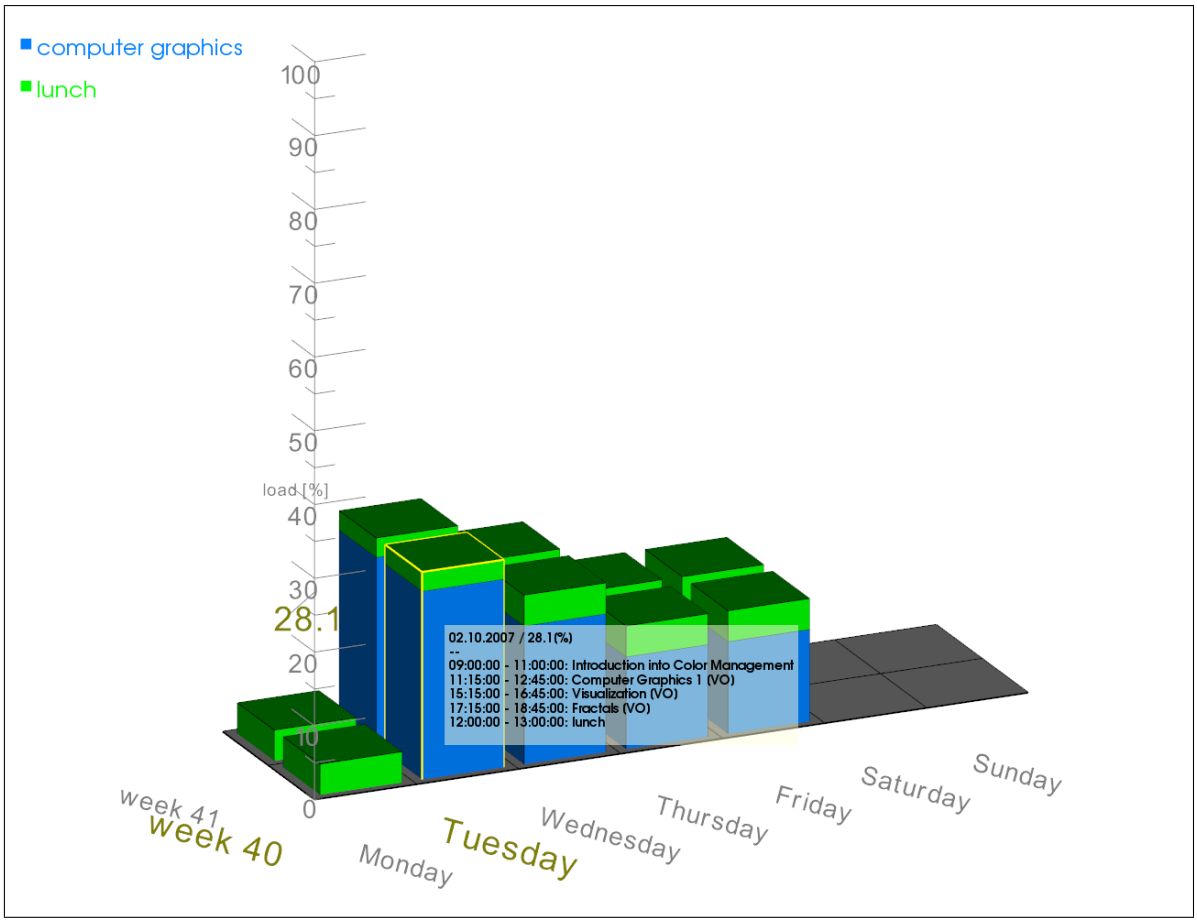


Figure 3.3: Heightfield visualization for a specific time slot showing an overlay overview of all activities.

When the short information is not sufficient, the second detail view can be activated. At this step, all workload blocks of one time slot are hidden and the activities appear. Each activity is then presented by a block on a time scale. To position the blocks, the unit of the workload dimension (z-axis) changes only for the selected interval to show the scheduling. A time scale of the represented time slot is visualized at the position of the selected cell. The visualization of the detail blocks around the time scale offers some advantages in addition to the first detail view (see figure 3.4). The labels of the time scale are visible through the half-transparent representation of the block. In this way, the start and end time of a block are recognizable. More descriptive information of an activity can be shown in an overlay. Furthermore, free slots in an interval of a time slot can be detected interactively, and overlapping activities are also visible if transparency is used. The comparison of overlapping blocks is limited and not a focus of this thesis. The comparison of different time slots (e.g., two successive days) is also possible. Therefore, the time scale uses the axial billboarding technique [3]. It always points to the viewer, if the heightfield object is rotated around the z-axis (e.g., the workload dimension).

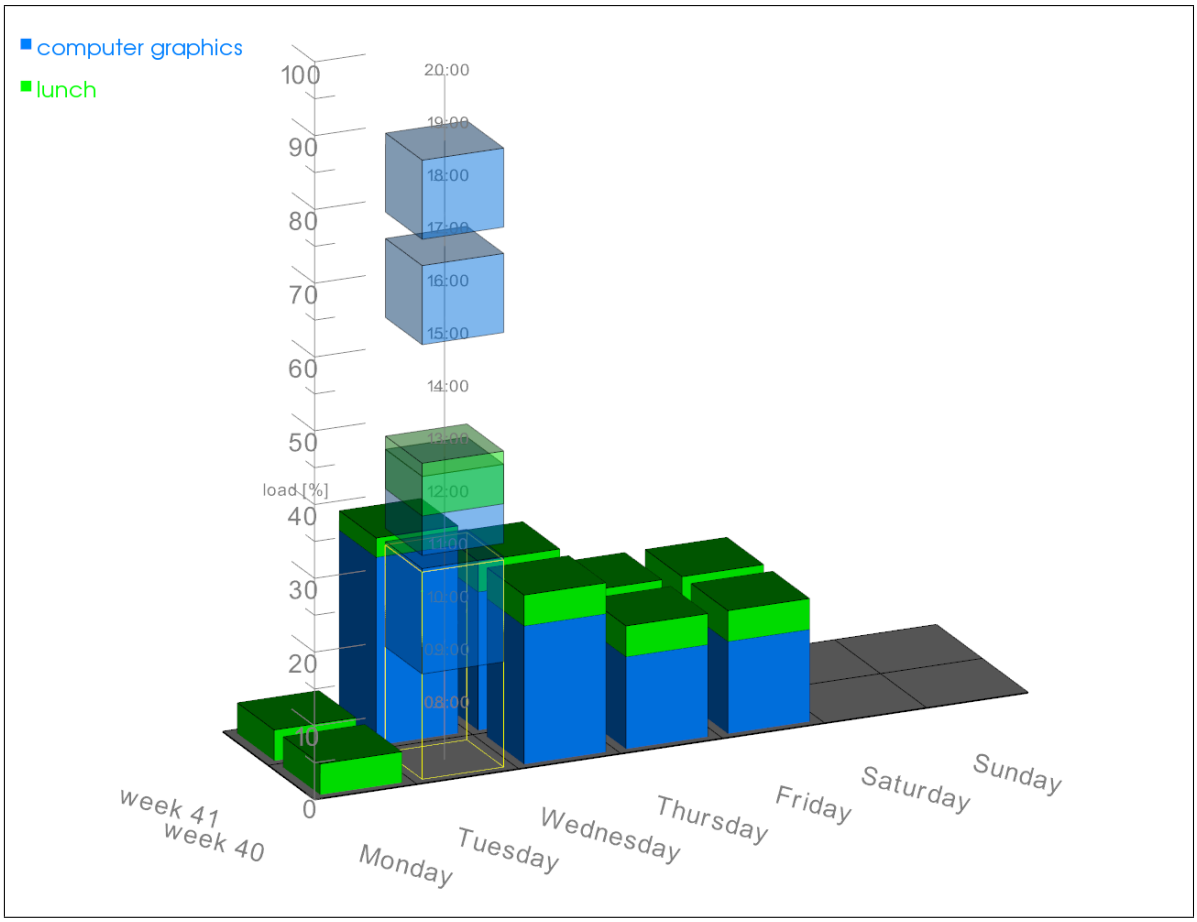


Figure 3.4: Heightfield visualization for a specific time slot showing a detail view with time scale.

3.4 Dependencies and Relationships

Another challenge of this thesis was to find an appropriate visualization for activities of different time slots that belong together. Calendar data can have dependencies, for example, an activity does not start till another one has finished, or an activity is repeated at regular intervals, a so-called recurring date. The color is already used to emphasize the categories of a period. Another feature for highlighting groups has to be found.

According to the examples PlanningLines [2] and Event Tunnel [30], the activities, which belong together, are displayed by connections. The main focus is on analysis of the calendar data, dealing with the workloads in a heightfield, hence this feature has to be applied on the height values. A straight line connection between two workload blocks could be occluded by other blocks. Therefore, the connection cannot be linear, it has to be bent in the direction of the workload dimension (i.e., the z-axis). This results in drawing arcs to connect two different, but grouped, blocks from different time slots (see figure 3.5). Neuman et. al. [24] presented a technique for exploring recurring events in 2D visualization also through arcs.

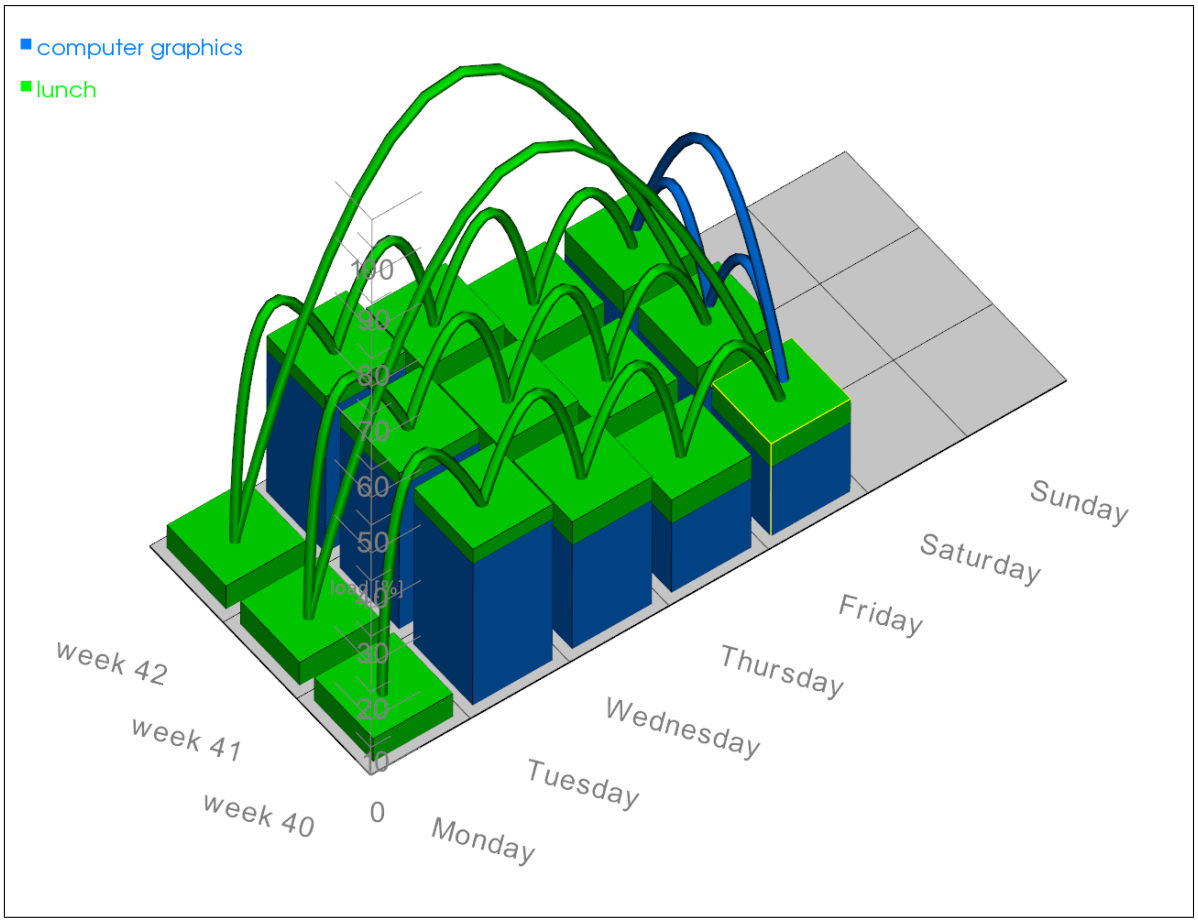


Figure 3.5: Heightfield visualization showing recurrences in form of arcs.

An arc is described by a parametric spline [14], which here for simplicity is defined by three points. These points are the start and end point on each block and one point in the middle between the first two. The middle point should be higher than the others to generate an arc and to overtop a bigger block between the connected two. This type of connection is almost always visible, but it is difficult to distinguish between the affected blocks in front, side and top view. The sequence of dependent activities is explicitly given by the time, thus a direction glyph (e.g., an arrow) is not needed.

Each arc is displayed in the same color as the color of the category, as not all activities of a workload are grouped. The occlusion of similar arcs, which refer to different categories, has to be minimized. A random generator is used to perturb the start and end points a little bit.

The visualization of groups through arcs is dynamical and appears on-demand. This feature is used to show groups, dependencies, or recurring activities. The blocks belonging together build up a chain. It does not matter which element is selected, the complete path within the observed period of time is displayed. To retrieve the information on activities involved in a chain, the arc must be selected and the corresponding blocks of the time slots are presented in a detail view.

Chapter 4

Interaction on the Calendar Heightfield

In chapter 3 the heightfield visualization of calendar data was presented. In this chapter the interaction with the heightfield and its data is shown. For an effective exploration and analysis of large data sets, appropriate interaction techniques are as important as visualization techniques. The interaction with the heightfield should be simple and efficient. Though, 3D space offers more degrees of freedom, it can quickly lead to confusions.

The heightfield can be translated in and rotated about any direction in 3D space. These transformations are realized by the movement of the camera around the object, including zoom. The individual movements use the input devices, mouse and keyboard, in combination. The mouse is also used for selection. First, highlighting an object in the heightfield is explained. This helps to retrieve useful information on clustered workloads and their activities. In 3D space often objects are partially or fully occluded by others. This may not matter for certain viewpoints, but sometimes it is important to have an unobstructed view on an interesting block. Therefore, another technique is presented. Additionally, a method called fuzzy scheduling is introduced. This method helps to schedule asynchronous and fuzzy activities, such as tasks. All these interaction techniques and their on-demand features are processed immediately.

4.1 Highlighting

Highlighting is one of the most well known brushing [37] techniques for interactive visualization. It can be used to select and emphasize items or to get some on-demand information out of the data. Highlighting is used to navigate through the heightfield and to retrieve some information by pointing at and selecting the data (i.e., a workload block). The retrieved information is hidden again by selecting the same item once

again.

There are two visual highlighting methods which help to navigate in the heightfield. On the one hand, the labels of the axes and, on the other hand, the interesting workload block is highlighted. This is initially done if the user points at a block (see figure 4.1). Even though the interaction takes place in 3D space, the movement in the heightfield is based on a 2D grid. Therefore, it is sufficient to determine the position of the picked grid cell and to visualize it. The grid lines are drawn not to loose the orientation. The position in time and the workload value in percent of the block are shown on the labels of the axes. This is done by changing the font size and the font color. Showing the information above the picked block cannot be done as this overlaps with the detail information.

A block in the heightfield is highlighted by a colored wireframe, which encloses it. The advantage of this method is that the color of the block is not changed. This means, that the visual mapping of the category to color does not get lost. The idea to scale up the selected block was rejected, since the workload would be interpreted wrongly or the neighboring blocks would be occluded.

Highlighting is a pre-stage for other interaction techniques, including the detail views (section 3.3), the display of groups and dependencies (section 3.4), and the importance view, which is explained in the next section. All interactions can be performed by pointing and selecting an object with a mouse. Some of the mouse events trigger highlighting actions. Highlighting gets fired by a mouse over event while hovering over the heightfield. If the mouse points at the same workload block for some time while it is not moving, another event is fired. This timer based event displays an overlay overview of detail information. The more detailed information view, with time scale, is activated if the interesting block is selected by pressing a mouse button while pointing at it.

The selection of multiple blocks at a time is also possible and can help to compare them with each other. Blocks having attribute values in common can be grouped dynamically (section 3.4). This results in filtering calendar data manually, which is similar to categorical clustering.

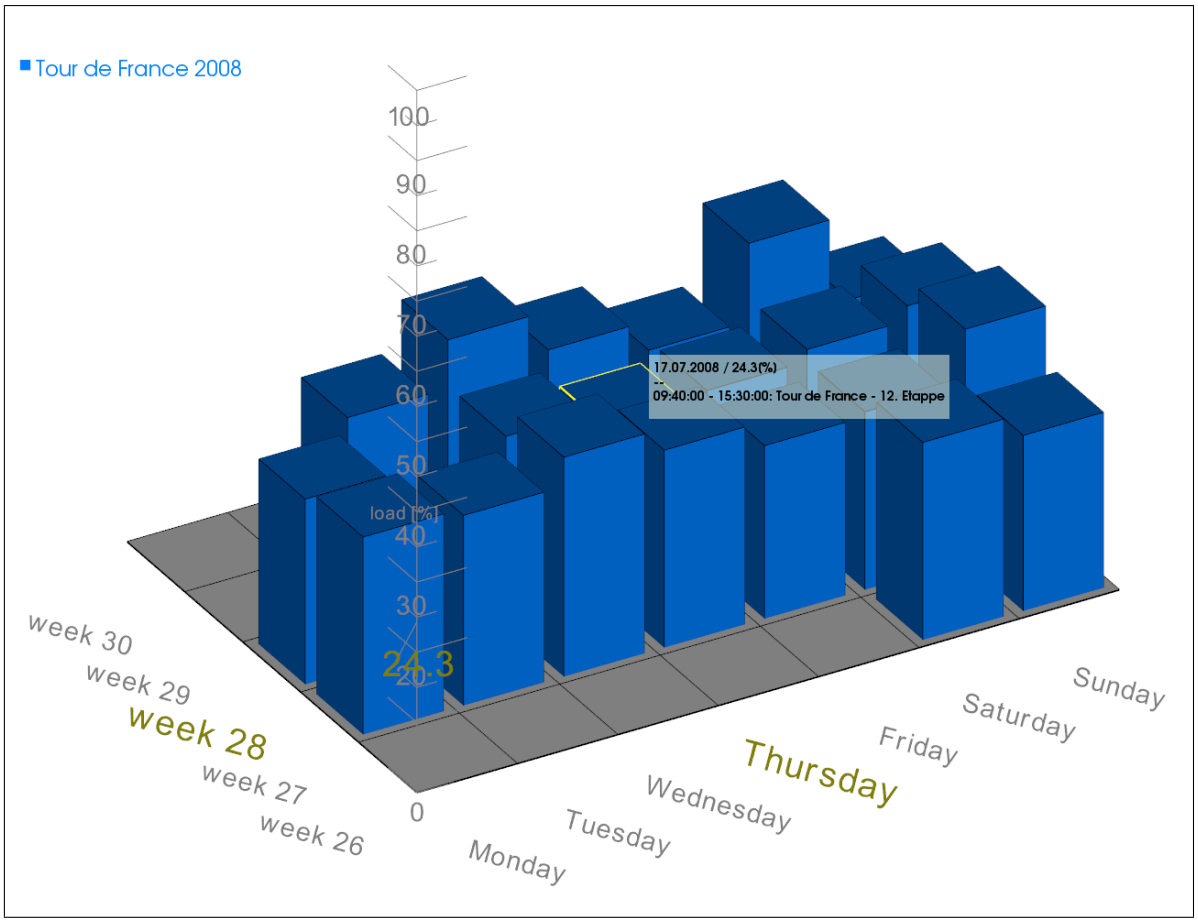


Figure 4.1: Heightfield visualization of highlighted block and labels showing information in an overlay overview.

4.2 Importance Visualization

To view an interesting block in the heightfield is sometimes not as easy as assumed at first, because interesting objects are often occluded by other objects. Occlusion is not a heightfield specific problem, and there are interesting visualization techniques that handle it, such as importance-driven visualization and other methods [36] [35]. The motivation of importance visualization in the calendar heightfield is to support the viewing of interesting blocks, independently of the viewpoint and if other blocks occlude the interesting one partially or fully.

To understand how importance visualization works, occlusion culling is explained first. An occlusion culling algorithm [3] determines the objects hidden by other objects in a scene and tries to cull them away. This can be done, for example, in a front to back manner, starting by the viewpoint. In contrast to occlusion culling methods, here, the importance visualization technique traverses the scene in a reverse manner, starting by the interesting block, to determine the occluders in front of it.

The prototype for this thesis has a simple reverse occlusion culling algorithm implemented. For simplicity, the problem can be reduced by one dimension to the 2D regular grid of the heightfield. Figure 4.2 illustrates this situation. The view to the interesting block (the red colored square, defined by the coordinates (x, y)) should not be occluded by other workload blocks. The culling area is defined by the boundaries of the important block in the direction of the camera position. The shape of the area (defined by the viewpoint and two vertices of the block) can be interpreted, for example, as a triangle or a rectangle. Actually, the prototype uses a rectangular shaped area, as in figure 4.2. The area is specified by two parallel dashed lines. The affected blocks are shaded in gray.

The algorithm 1 $GetOccluders(x, y)$ gives a straight forward method to find the occluders in front of the important block, which are lying partially or completely in the area of interest. Intersection tests [3] are used to determine if a block is an occluder. The main part of the algorithm are the two $while$ loops, to traverse the rows and columns of the grid. If all columns of one row have been checked (i.e., $boundary_y$ has been reached), the next row gets traversed, by changing the $boundary_y$ and walking direction to their opposites (see lines 16ff in algorithm 1). The code could be optimized by sharpening the $while$ conditions.

If a block or a region of interest gets highlighted as being important, all occluders change their appearance from solid into wireframe as long as the block is in focus. This feature depends on the camera position and is fully interactive. If the camera is moved in 3D space and a block is highlighted to be important, there is always a free view on that block guaranteed.

Culling techniques are acceleration algorithms, since they do not send the objects to the rendering pipeline [12]. However, the occluders in front of the selected workload block change their appearance interactively from solid into wireframe mode, so that the context is not entirely discarded. This feature is used in combination with highlighting, and it can be also used for groups of objects. This means that every object in the group is an important block.

Figure 4.3 shows the advantage of this interaction and visualization technique. Occluded objects can be made visible, while the context does not get lost.

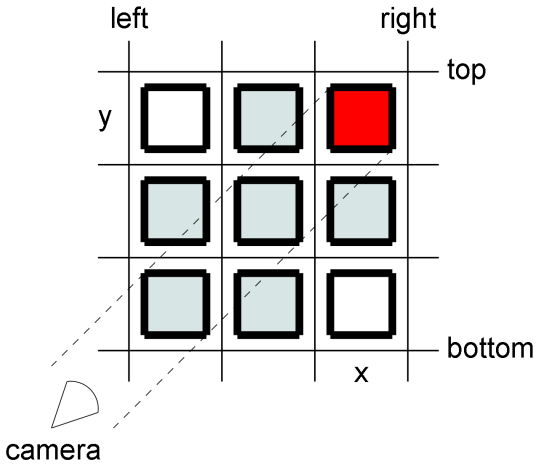


Figure 4.2: Importance visualization schema: Determine the occluding blocks by checking for intersections with the rectangular shaped area, specified by two parallel dashed lines.

Algorithm 1 $GetOccluders(x, y)$

Require: boundaries b_{left}, b_{right}, b_{top}, b_{bottom}; $b_{left} \leq x \leq b_{right}$; $b_{bottom} \leq y \leq b_{top}$

$O = empty$
$p =$ position of the important block, determined by (x, y)
$c =$ position of the camera
$\vec{v} = c - p$
$col = x$
$row = y$
$boundary_x = (\vec{v}_x$ is pointing left) ? $b_{left} : b_{right}$
$boundary_y = (\vec{v}_y$ is pointing down) ? $b_{bottom} : b_{top}$
while $row \neq boundary_y$ **do**
  while $col \neq boundary_x$ **do**
    $col = col + \vec{v}_x$
    if $obj_{(row,col)}$ is an occluder **then**
      $Add(obj_{(row,col)}, O)$
    end if
  end while
  $boundary_x = (boundary_x$ is $b_{left})$? $b_{right} : b_{left}$
  $\vec{v}_x = \vec{v}_x * (-1)$
  $row = row + \vec{v}_y$
end while
return O

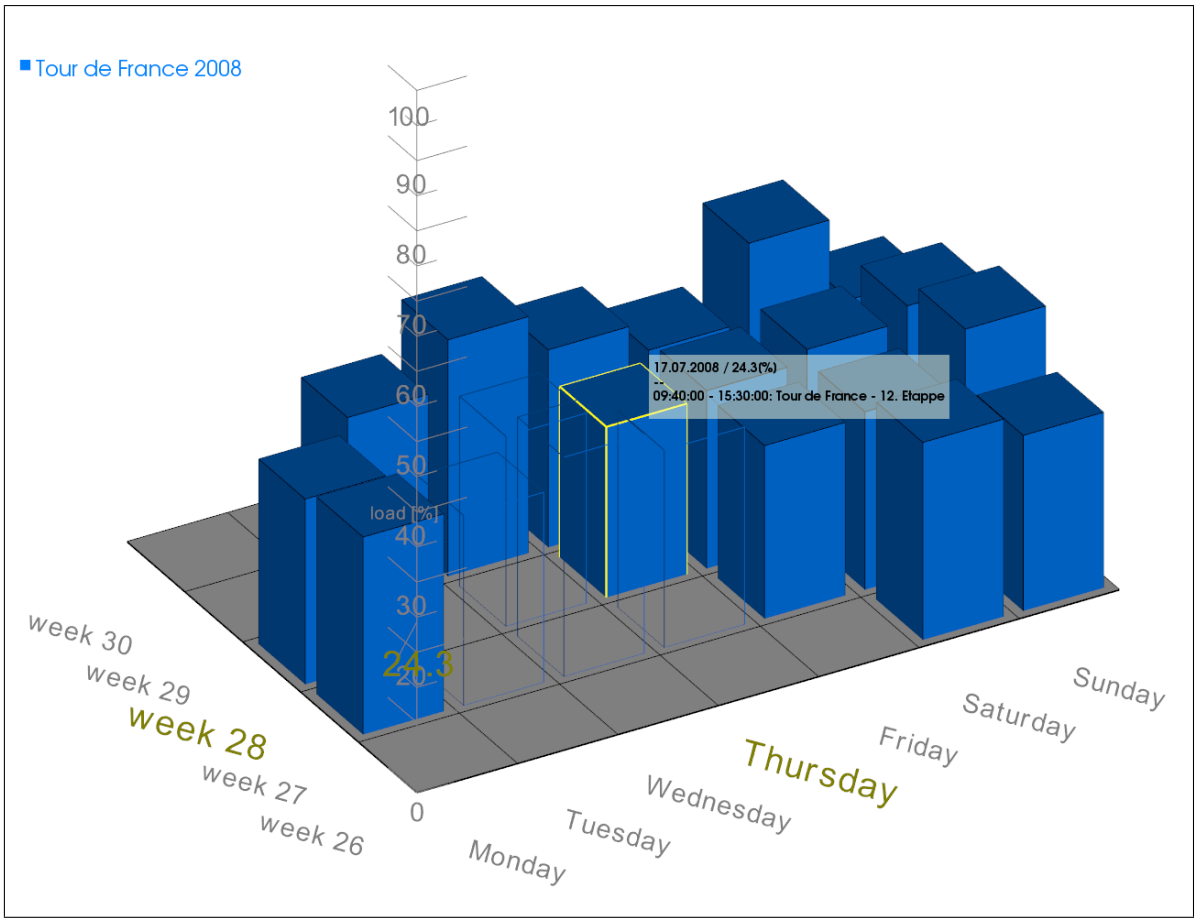


Figure 4.3: Heightfield visualization showing an interesting block with non-occluding objects in front of it.

4.3 Fuzzy Scheduling

In default software applications for calendar data (section 2.3), the activities shown in a schedule are almost always well defined by its start and end time. Although, tasks also can be specified by start and end times, they usually are not integrated into one's schedule, because of possible missing time attributes. A challenge is to use the calendar heightfield for scheduling all types of activities, whether the time attributes are specified completely or not. Scheduling by on-demand workload balancing and allocation of tasks with missing time attributes are further goals of this thesis. To understand how asynchronous scheduling works, the definition of fuzzy tasks and allocating them into the calendar heightfield is introduced first.

4.3.1 Fuzzy Task

First of all the term fuzzy task has to be explained. The main difference between tasks and events is the uncertainty of available time attributes for tasks. If there is at least

one missing time attribute, then such a task is said to be fuzzy. Tasks must not be fuzzy, but often they are inaccurately specified by their start and end times. For example, tasks are frequently defined by a deadline, while the start time is missing. Since a task can be specified as well as an activity, by defining a start and an end time, there are three possible types of fuzziness:

- fuzzy start: only the end time is specified
- fuzzy end: only the start time is specified
- fuzzy start and end: neither start nor end time is specified

In default calendar applications it is not possible to treat such temporal uncertainties, and so, tasks are normally not integrated into the same calendar view of all other appointment and event entries. The motivation is to find a way to schedule events and tasks in the same calendar view. A possible solution is to define at least one more time attribute: an estimated effort in time units to complete the task, for example, until a specified deadline. Though it is feasible in some project planning systems [2], it is sometimes difficult to estimate an effort for known or unknown activities. A lot of experience is required.

The implemented prototype can handle fuzzy activities, if an estimated effort is specified next to a start or end time. The missing time attribute can be calculated and the fuzzy task can be inserted into the schedule. Tasks, that do not have a start nor an end time, are also able to be fitted into the schedule, if an estimated effort is specified. Since, the start time is currently not known, the task is not in process and it can be started earliest by now. Such fuzzy tasks are assumed in the calendar heightfield to be fuzzy on end time.

If a task is fuzzy, the type of fuzziness can be received by the time attributes. With this information and specifying an estimated effort for the task to complete it, the task can be integrated into the calendar heightfield. Tasks are similar positioned in the heightfield like other activities. If a task is completely specified by its time attributes, it is treated as an event. To determine the position(s) for a fuzzy task in the heightfield, several attributes are taken into account. The possible degrees of freedom, specified by the number of attributes, for a task are:

- fuzzy type
- estimated effort
- cluster size
- priority

There are two new attributes: the cluster size and the priority. Maybe a task needs several days to be completed, the task has to be split up on these several days. For example learning for a test can be split up into sections and distributed over days.

Cluster size defines the smallest time unit a task can be partitioned into, and priority is used to order several tasks from the very to the less important. Both attributes can be defined by the user. The default assumption in the prototype for cluster size is one hour and all tasks have the same priority. A definition for cluster size lower than one hour is possible and maybe in some cases useful. However, the fragmentation of a task increases in inverse proportion to cluster size. The fragmentation indicates the scattering of parts of a task across the calendar.

By specifying these attributes, positioning fuzzy tasks into the calendar heightfield can be achieved. Figure 4.4 shows a solution to insert a fuzzy-end activity. A task a_1 is inserted in the second time slot t_2, since the first time slot t_1 currently has reached the workload limit. The cluster size is set to 1 for simplicity. The task a_1 does not fit completely into time slot t_2 and it is split into two parts, a_{11} and a_{12}. The second part a_{12} is distributed to the next available time slot, which is not fully loaded, t_3. The part a_{12} exceeds t_3, so it is split up again. This process is done as long as all parts are entirely inserted.

The algorithm 2 $InsertFuzzyDate(a)$ provides a solution to find not fully loaded time slots to insert a fuzzy task entirely. This is done for all fuzzy activities in the set A_{fd} and the fuzzy activities are sorted by priority in descending order. The workload limit can be specified. Q is a set of workloads of the heightfield.

First the starting position of a is determined. The propagation direction for distributing the parts of a is also retrieved by the type of fuzziness. The negative direction is only used for fuzzy-start tasks, for example, a deadline task. Therefore, the process starts at the given end time position (i.e., a time in the future) and searches in the reverse direction of time, bounded by now. It is not possible entering fuzzy tasks partially or fully in the past, this is verified every step in the $while$ condition. As long as the rest of estimated effort is greater than cluster size, a cluster size part is split up and inserted into a time slot, if the workload of that time slot will not exceed the limit. If the time slot in the calendar heightfield was not empty, the existing block has to be raised by the workload amount, otherwise a new block has to be inserted. This is done by $UpdateBlockGeometry(pos, load)$ in line 16 and 23.

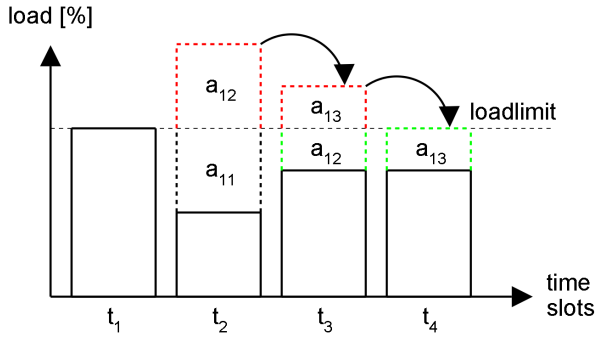


Figure 4.4: Insert fuzzy task schema: Insert a fuzzy-end task into a schedule. The red dashed area (i.e., rest of estimated effort) exceeds the workload limit and is distributed on not fully loaded time slots until the entire activity is inserted.

Algorithm 2 $InsertFuzzyDate(a, loadlimit)$

Require: $a \neq 0$, $0 < loadlimit \leq 100$ percent, $q \in Q$

1: /* starting position of a */
2: $pos =$ today
3: **if** type of a is fuzzy start **then**
4: $pos =$ end time
5: **else if** type of a is fuzzy end **then**
6: $pos =$ start time
7: **end if**
8: /* propagation direction for distributing the parts of a */
9: $dir =$ (type of a is fuzzy start) ? (-1) : (1)
10: $x =$ estimated effort of a
11: $c =$ cluster size of a
12: **while** $x > 0$ and pos is valid **do**
13: **if** $c <= (x - c)$ **then**
14: **if** $(load(q_{pos}) + load(c)) < loadlimit$ **then**
15: $load(q_{pos}) = load(q_{pos}) + load(c)$
16: $UpdateBlockGeometry(pos, load(c))$
17: $x = x - c$
18: $continue$
19: **end if**
20: **else**
21: **if** $(load(q_{pos}) + load(x)) < loadlimit$ **then**
22: $load(q_{pos}) = load(q_{pos}) + load(x)$
23: $UpdateBlockGeometry(pos, load(x))$
24: $x = 0$
25: **end if**
26: **end if**
27: $pos = pos + dir * size(timeslot)$
28: **end while**

4.3.2 Scheduling of Fuzzy Tasks

In the previous section the term fuzzy task and how such tasks can be inserted into the calendar heightfield are described. The process of allocating fuzzy tasks depends on several attributes. One of the most important attribute is the workload limit. The motivation is to change the workload limit globally for the entire calendar heightfield and modify the distribution of workloads on-demand, by balancing the fuzzy tasks across the time slots.

This is done very similar to the allocation of fuzzy tasks, starting with a set Q of workloads including A_{fd}. Each time slot has its own specific workload. If the workload limit drops below a specific workload of a time slot including fuzzy tasks, then one or more fuzzy tasks are split up and the parts are distributed in the neighborhood. How many tasks are involved of this process depend on the new workload limit and the attributes defined in the previous section 4.3.1. For example, the priority of the fuzzy tasks specifies the order in which they are treated.

The simple, straightforward, but effective algorithm 2 (in section 4.3.1) has to be adapted. The result is given in algorithm 3 $ScheduleTodoList(Q, loadlimit)$. All blocks of the calendar heightfield have to be tested against the workload limit. If the workload limit decreases, the fuzzy tasks distribute partially or fully across the heightfield. This process is also called ''flood-scheduling'', because fuzzy tasks flood from higher to lower workload blocks. The reverse direction (i.e., ''combine-scheduling'') is also possible, consequently the split parts of tasks are combined together again. The algorithm, here, shows only the flooding method.

Each fuzzy task in a block has to be checked, if the magnitude of $offset$ is greater than the cluster size or the current estimated effort, to start the scheduling process for that fuzzy activity. $Offset$ is specified by the difference between the current workload limit and the workload of this block. The sign of $offset$ determines the scheduling direction. If $offset$ is less than 0, then the flooding process is used. The workloads and the block geometries are updated if another not fully loaded time slot has been found. The old block shrinks and the new one raises.

The scheduling process is done on-demand by user interaction. To follow the scheduling process, the geometry of a changing block has to be updated immediately (i.e., in real-time), otherwise the recognition of the distribution will be disturbed. If there would be many fuzzy tasks moving in between the calendar heightfield while scheduling them, a user would be confused. Therefore, the technique to visualize group connections (section 3.4) is used again (see figure 4.5(f)). This technique show arcs, that group the fuzzy tasks which belong together. It is possible to detect where parts of a fuzzy activity have moved. In a first design step another color has been used for such tracking, but this results in a conflict to the default color usage (section 3.2).

Algorithm 3 $ScheduleTodoList(Q, loadlimit)$

Require: $Q \neq$ empty, $0 < loadlimit \leq 100$ percent

for all $q_i \in Q$ **do**
  if $load(q_i) < loadlimit$ or q_i has no fuzzy tasks **then**
    $continue$
  end if
  $offset = loadlimit - load(q_i)$
  if $offset < 0$ **then**
    for all $a_i \in q_i$ and $offset < 0$ **do**
      $c = min($cluster size of a_i, current effort of $a_i)$
      if $abs(offset) < load(c)$ **then**
        $continue$
      end if
      $pos = GetPositionOf(q_i)$
      $next = pos$
      $found = false$
      while $next$ is valid and not $found$ **do**
        if $(load(q_{next}) + load(c)) < loadlimit$ **then**
          $found = true$
          $offset = offset + load(c)$

          $load(q_{next}) = load(q_{next}) + load(c)$
          $UpdateBlockGeometry(next, load(c))$

          $load(a_i) = load(a_i) - load(c)$
          $UpdateBlockGeometry(pos, -load(c))$
        end if
        $next = next + GetPropagationDirectionOf(a_i) * size(timeslot)$
      end while
    end for
  end if
end for

A fully interactive scheduling of fuzzy tasks is implemented in the prototype. Figure 4.5 shows a sequence of three fuzzy types flooded, until the maximally distribution has been reached. To control the scheduling process, the user moves a slider on a bar on the right side of the window (see figure 4.5(a)), by interacting with the mouse. A transparent plane in the workload dimension (z-axis) is linked by the position of the slider and is also moved up and down. While the slider always shows the current position in percent, the plane is smoothly integrated in the heightfield to represent the current workload limit. The plane is drawn transparently to see the workloads below. By starting the fuzzy scheduling process, the initial position of the plane starts above the maximum workload of all workload blocks in the calendar heightfield. The user pulls the plane down, if the plane intersects blocks, their fuzzy activities are distributed over the available time slots (see figure 4.5(b), 4.5(c), 4.5(d)). The intersection test

is determined by an offset between two scalar values, i.e., the workload of the block against the position of the slider. Each affected block is raised or shrunken by the offset, which is added or subtracted. Since the workloads of the blocks change, the geometry representation also has to be adapted. This process is done, as long as the requirements for fuzzy tasks (see section 4.3.1) are fulfilled and the maximally possible distribution has not been reached (see figure 4.5(e)). The maximally possible distribution depends on the data set and it is reached, if no fuzzy activity can be flooded anymore.

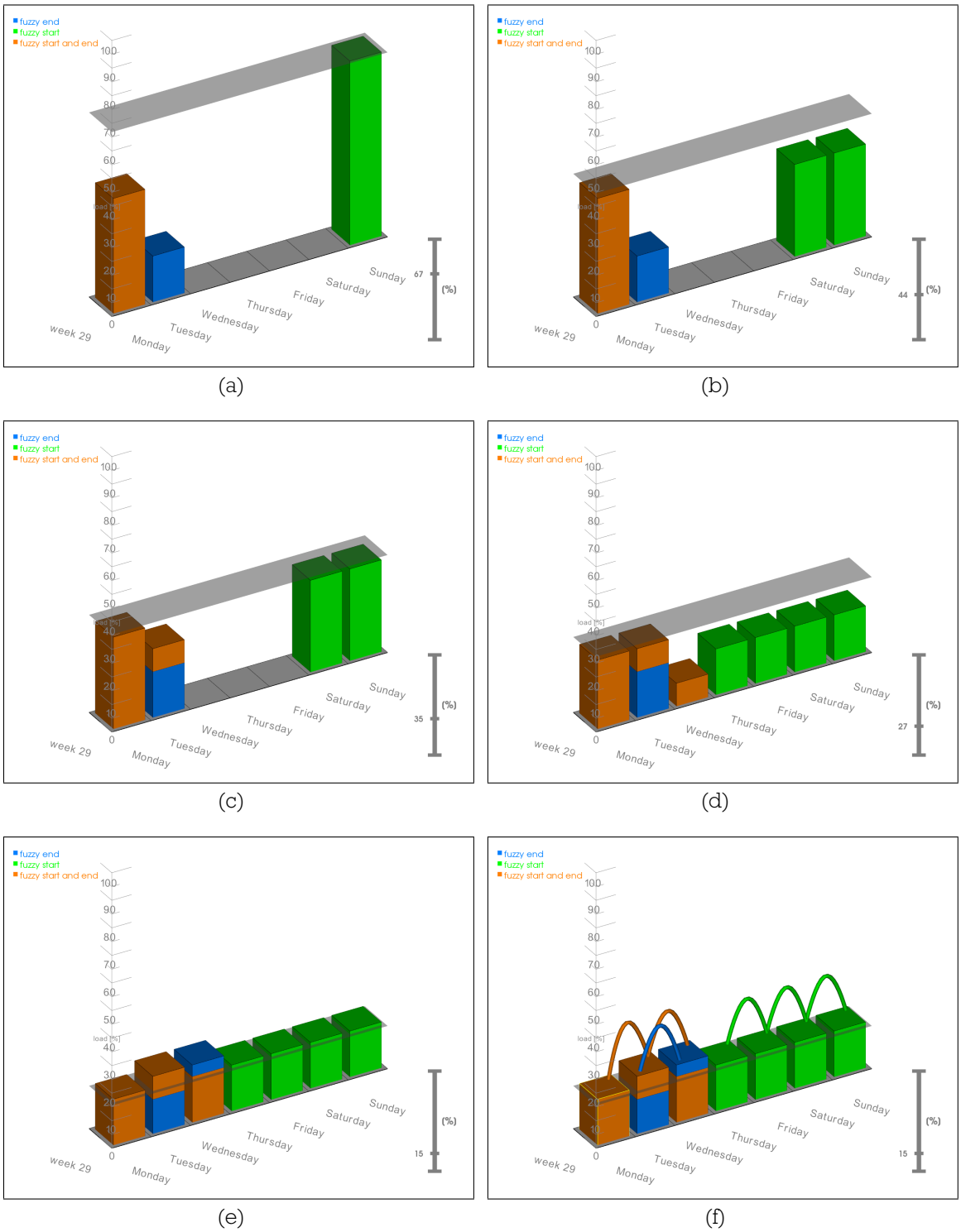


Figure 4.5: Heightfield visualization with fuzzy scheduling: (a) shows 3 types of fuzzy tasks, the maximum workload starts by 67%, (b) flooded by the transparent boundary plane to 44%, (c) 35%, (d) 27%, (e) 16%, the minimum has been reached, (f) shows arc connections to visualize the fuzzy tasks which belong together.

Chapter 5

Advanced Statistic Views

Another important sub-task in visualization is the search for visual patterns. There are many possibilities to accomplish this, for example by trying to find groups in data [19], while searching for patterns or trends. In this chapter, a possibility to help the user finding patterns and trends by visualizing advanced statistic views, is presented. The mathematical science of statistics is often used to summarize, analyze or describe a collection of data. This process is supported by visualization and interaction techniques.

There are many examples in information visualization to show statistical interpretation of the underlying data, such as scatterplots, histograms or other types of charts [13]. The structure of the calendar heightfield resembles a 3D bar chart. Showing the workload distribution over a period of time, clustered into time slots, can be interpreted as a statistical graphical analysis. Calendar data can be described through statistics. For example, it is possible to show how often an employee was ill or how the capacity utilization in the past two weeks was. There are several possibilities to interpret calendar data in statistical manner, it depends first of all on the data itself. Advanced statistic views are used to summarize and describe the calendar heightfield data by descriptive statistics and present the results in semantic views. Descriptive statistics [6] are used, to see, for example, the minima, maxima and means of workloads. The results, showing line graphs, are then visualized in semantic views, which are explained subsequently in the next section. Finally, two possible advanced statistical views to explore the calendar data are described.

5.1 Semantic Views

The calendar heightfield with its workloads can be viewed from almost every viewpoint in 3D space, the interpretation of the visualized data is always the same. To change the representation or show the user another meaning of the data, in case of changing the viewpoint, semantic views are involved. It is called semantic view, be-

cause it focuses on the meaning of the data in a specific viewpoint. In such a view special characteristics of the data can be emphasized. Besides data specific values or attributes, other information, such as descriptive notes or statistical interpretation, can also be superimposed. The semantic views are displayed only under specific viewpoints in the calendar heightfield view not to loose the context.

The semantic views are used within the calendar heightfield to represent various statistical results. Since the calendar heightfield is a 3D object, three semantic views are useful. In each direction of the Cartesian coordinate system, a statistical representation of the workload data is projected onto a plane. The view on a calendar heightfield makes only sense, if the workloads are always visible. Therefore, the semantic views will only fade in front, side and top view. The front view is defined by the xy-plane, the side view by yz-plane and the top view by xy-plane. As described in section 3.1, the z-axis represents the workload dimension, while the x- and the y-axis are the temporal dimensions.

The statistical views are immediately refreshed, if the calendar heightfield data has changed, for example, after a fuzzy scheduling process (section 4.3). The visualization of the semantic views is an interactive process. The semantic view slowly fades in, if the direction of the viewpoint becomes parallel to the normal vector of the projected view plane. It changes from invisible to transparent to nearly opaque, the more one is interested in that view. Such a semantic view act as a transparent mirror, to see the calendar heightfield at any time, in order not to loose the context.

Figure 5.1 shows, where the semantic views are placed and the directions along which the heightfield data is processed. This results in two trend views and one pattern view. In the figure they are specified as $view_1$, $view_2$ and $view_3$. The statistical representation and the interpretation of these views are explained in the following sections.

5.2 Trend View

The semantic front and side views are so-called trend views. They show how the distribution of workloads trends over a specific time dimension. In these views, descriptive statistics is built from the calendar data in each particular view. Many statistical evaluations are possible, often the simplest ones are very expressive. In a bar chart it is possible to show several characteristics of descriptive statistics, the center, spread and, for example, the skewness of the data. While viewing the calendar heightfield along a time dimension, the first statistical evaluation through a bar chart is given (see figure 5.2(a)). It shows the maximum workloads of the time slots for the viewing time dimension. This view makes it possible to determine the maximum workloads interactively. However, free or not fully loaded time slots can not be detected, as the

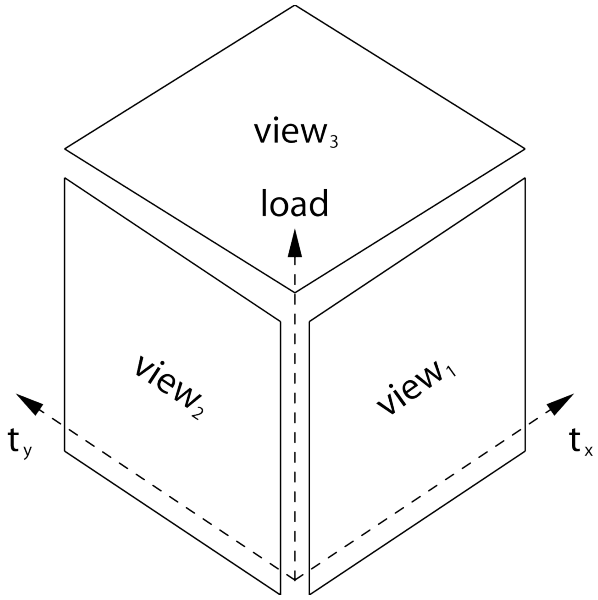


Figure 5.1: Schema of semantic views for the calendar heightfield visualization.

calendar heightfield is projected in a front or side view. To get more involved about the distribution of the workloads along a time dimension, the following statistical values are calculated from the workloads of the calendar heightfield:

- range
- arithmetic mean
- standard deviation

The range of the workloads over a period of time is defined by an interval between the smallest and the greatest observation. The arithmetic mean is a measure for the average of the data set, it can be seen as an expected value. The standard deviation is a measure for the statistical dispersion of the data. The projection of the calendar heightfield along a time dimension results in a bar chart, as already mentioned. The intention is to calculate expressive statistical values out of the calendar data along a time dimension. The time slots along the time dimension have to be traversed. Therefore, two common ways exist to accumulate the workloads of these time slots. Either the number of observations or the total number of time slots is taken into account. The latter method is essential for determining entirely free time slots. Furthermore, the accumulation of the workloads can be done for each block including all categories or more detailed for each category.

The characteristics, range, arithmetic mean and standard deviation, are calculated for each time dimension (i.e. x- and y-axis) and for all time slots along this dimension. The results are then presented as line graphs in a semantic view (see figures 5.2(b), 5.2(c) and 5.2(d)). Since the workloads can be distinguished by their categories, the statistical representation can also treat the calendar data per category. Each line

graph on the plane corresponds to one specific category, visualized by the corresponding color. If the calendar data has only one category, then the statistical values are positioned in the middle of each time slot. To show statistical values for several categories, a small offset between each two categories is used to separate them, otherwise they would overlap and the information is lost. The arithmetic means of each time slot are connected by a solid line and the points are marked by hollow circles. A series of points connected by a continuous line is described as a polyline or curve. The range and the standard deviation are shown by thin dashed line representations. The outer points of both are depicted by triangles. The range is presented through a line, which connects the minimum and maximum workload of each time slot. The standard deviation shows the region of dispersion of the data around the arithmetic mean through a rectangle.

This exploratory method is used to analyze the workload trends in different time dimensions by studying the statistical graphical representations. The x-axis, for example, is divided into daily time slots, while the y-axis shows the weeks. Therefore, the front view shows a daily statistic (see figure 5.2(b)). Each weekday is summed up for several weeks and can be compared against each other. Analogously, the side view represents a weekly statistic (see figures 5.2(c), 5.2(d)), showing a weekly workload polyline.

For a better understanding, a simple data set with one category is used in figure 5.2. The left figure shows the statistical representation independently of the category, while the right one differentiates. To switch between these two representations a specified key on the keyboard has to be pressed. The left figure shows two arithmetic mean curves. The polyline (abbr. in the figure as mean2, symbolized with a square) representing the number of observations and their standard deviation (abbr. in the figure as sd, symbolized with a triangle) is shown in orange. The other polyline (abbr. in the figure as mean1, symbolized with a circle) represents the total number of time slots in black. The right figure takes the category into account and shows the arithmetic mean for the total number of time slots. The range is displayed instead of the standard deviation. Since there is only one category, the arithmetic mean curves of both pictures are equal. The range is similar to the standard deviation. The graphical statistics are recognizable and provide a simple but efficient method to analyze the data.

Figure 5.2: Visualization of statistical views: (a) shows all maximum workloads of the time slots along the week dimension in a bar chart, (b) shows a trend view for weekdays, including all categories, the orange arithmetic mean curve represents the number of observations and their standard deviations, the black curve represents the total number of time slots, (c) shows a trend view for weeks, including all categories, (d) shows a trend view for weeks, the blue curve represents the total number of time slots and the workload ranges for only one category.

5.3 Pattern View

Another representation of the data is given by the top or pattern view. First of all, the calendar heightfield is viewed from above along the negative direction of the workload dimension (i.e., the camera points down the negative z-axis). The grid, defining the time dimensions, is shown, while the heightfield structure is lost. The intention is to find a new compact representation of the heightfield structure in the xy-plane, to support analyzing the workload distribution. In contrast to trend views, this semantic view treats each block of each time slot, to show a statistical interpretation. This view shows for each time slot two values:

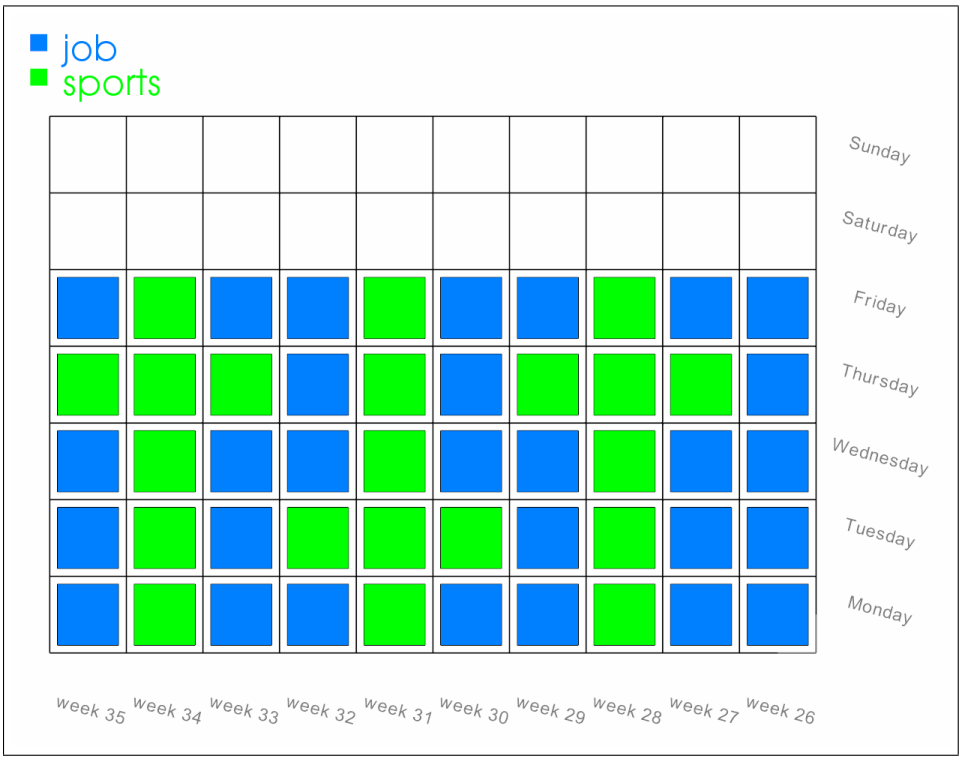
- the overall workload
- the categorical distribution

Both values are treated as attributes of a quadrangle for each time slot in the grid view. Each quadrangle represents the overall workload of that time slot by its size and the categorical distribution of that time slot is encoded in color. The size of a quadrangle in the grid view is proportional to the amount of overall workload of a time slot. The maximum possible size is given by the cell dimension of the grid view and corresponds per default to 100 percent workload. Correlations of similar sized quadrangles and workload distributions are detectable. Furthermore, fully loaded time slots are recognizable very fast, since they cover the underlying block entirely. As mentioned in section 3.1, the maximum base of a block is always smaller than the cell size. The maximum size of a quadrangle is therefore also defined, the predefined margin to the cell is the same.

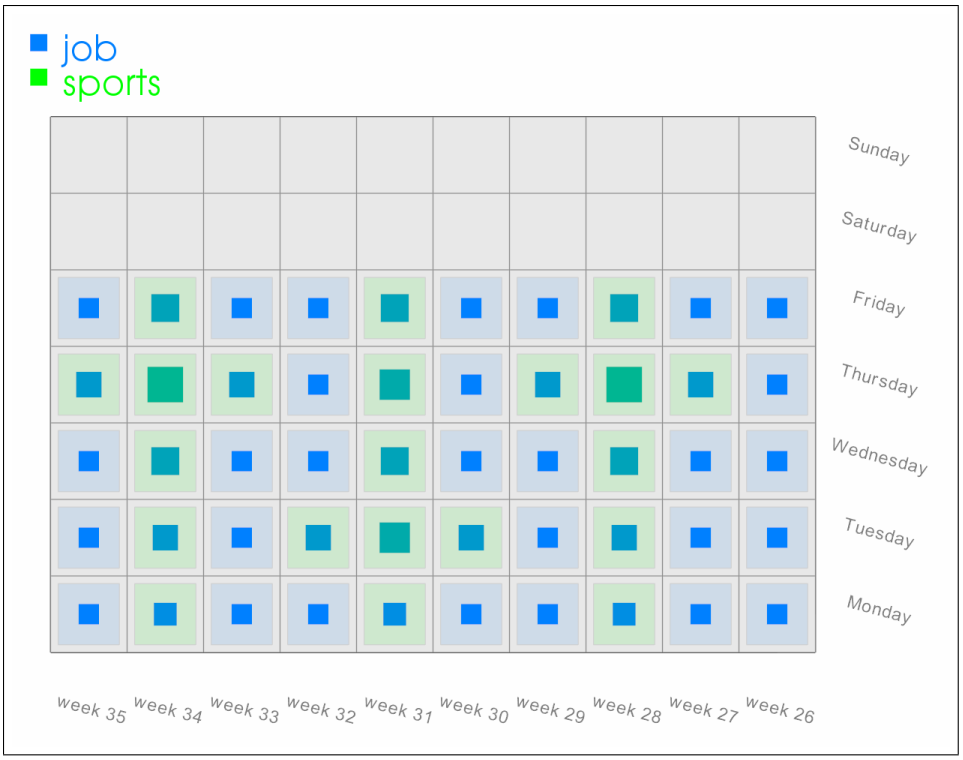
If a time slot in the calendar heightfield has more than one category, each category has its own workload part and colored representation. To show the influence of each categorical workload part based on the entire workload of a time slot, each unique color is weighted by its workload part. This results in color blending as in equation 5.1. The new summed color is a mixture of all other category colors and indicates the including categories. The category color with the greatest workload part will dominate the summed color. With a little practice in RGB color model it is possible to extract the primary colors and associate the categories. Consequently correlations of similar colored quadrangles and categorical distributions are also detectable.

$$color_{sum} = \sum_{i=1}^{n} color_i \frac{load_i}{\sum_{j=1}^{n} load_j} \tag{5.1}$$

Calendar data often have recurring activities or events. For example, the daily cycles of a working people do not differ greatly. This view is predestined to visualize patterns and to help finding visual structures by the size and the color of distributed quadrangles. A simply real life example is given in figure 5.3. It shows the workload distribution of job and sports. Figure 5.3(a) shows a top view, without the pattern view, while figure 5.3(b) displays the colored quadrangles in a pattern view. This should illustrate how patterns can be detected. Both pictures show which days are loaded and which are free. It is impossible to say something about the workloads for each day in figure 5.3(a), but the size of the quadrangles in figure 5.3(b) helps to imagine. Job is definitely less than half as big and is uniformly distributed. Several workloads, also including sports, are greater than half as big. Every middle of a cross certainly has a workload greater than 50 percent. Furthermore it is interesting, that the sports workloads are increasing until the middle of the cross is reached.



(a)



(b)

Figure 5.3: Visualization of top and pattern view: (a) shows a top view of the calendar heightfield without statistical representation, (b) shows a pattern view, the size of a quadrangle represents the underlying workload and color blending indicates the included categories.

Chapter 6

Implementation

To demonstrate the visualization and interaction techniques of the theory, a prototype was implemented. To concentrate on the realization of the design study the development cycles were kept relatively small and a visualization framework was used. The programming language C++ in combination with Microsoft Visual Studio 2005 was used. The rendering pipeline is realized through the Visualization Toolkit [18]. Import and export of calendar data is done by the libical library [7], which is an open source implementation of the iCalendar protocol. Modifications to both libraries were performed. Each library is shortly introduced. An overview about the architecture of the prototype is also given.

6.1 Visualization Toolkit

The Visualization Toolkit, in short VTK, is an open source framework developed by the Kitware company. It is freely available and used by thousands of researchers and developers around the world. It offers a huge C++ class library and supports different platforms. VTK is an object-oriented software system for 3D computer graphics, such as scientific visualization, image processing and volume rendering.

6.2 iCalendar

The iCalendar file format was used to import and organize the calendar data in this prototype. It is standardized and supports interchange between different calendar applications. The full specification of iCalendar is defined in a request for comments (rfc) document [11].

Currently, the prototype supports only two types of calendar activities: events and tasks. They both have several descriptive attributes in common, such as categories, priority and summary. The category is important for building groups of similar activities, such as family, work and other. Therefore, calendars of two or more people can

be easily distinguished. The summary gives only a short meta information about the calendar activity. Both types have date and time related information. The time inputs for an event are usually defined by a scheduled amount of time, for example by a start and an end time of an activity. Sometimes a duration is given instead of an explicit end time. A task can also have time inputs, but normally these are missing and the activity is said to be fuzzy.

The iCalendar specification is standardized according to rfc, although it offers the possibility to define non-standard attributes. Tasks are extended by the possibility to specify an estimated effort and a cluster size. Both attributes represent integer values in time units. The estimate effort specifies how long a task will take until it is completely finished and the cluster size defines the smallest time unit a task can be partitioned into.

6.3 Architecture

Figure 6.1 shows the architecture of the implemented prototype, and how the procedure of importing calendar data until visualizing the calendar heightfield is working. Each stage of the procedure consists of several parts with several classes. First the calendar data is acquired. Some of the data have to be prepared, such as fuzzy tasks. In the third step the calendar data is visually mapped. Finally, the calendar heightfield is shown and the exploration can start. The next sections concentrates on how the parts fit together and where they are applied. Source code or exact interface definitions are not explained.

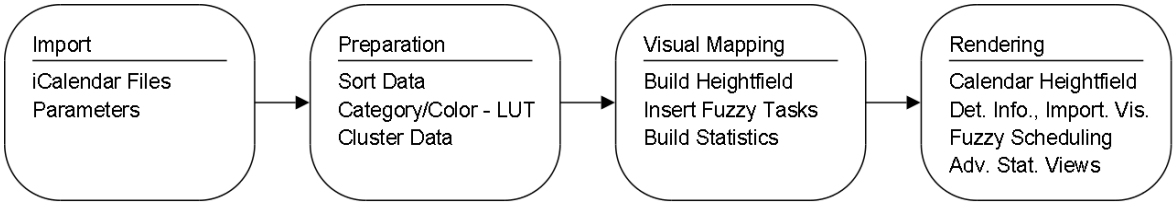


Figure 6.1: Implementation architecture schema: shows four stages, each consists of several parts.

6.3.1 Import

First of all the prototype needs some data to work with, therefore, iCalendar files can be imported. Furthermore some parameters can be specified, such as the observed period of time and the interval for a time slot. All calendar activities within the period of time are stored in a list. The additional attributes, such as estimated effort and

cluster size for a fuzzy task, are preset by the prototype, if fuzzy tasks are available at all.

6.3.2 Preparation

The activities in the list, such as events, are ascending sorted by start time. During sorting all activities, the available categories are picked out. The categories are then assigned to colors. This mapping is stored in a lookup table.

The sorted activities are clustered into time slots. Therefore, all activities are traversed once again. The fuzzy tasks are picked out and stored separately. The observed period of time is partitioned by the interval for a time slot. A time attribute normally consists of a date and a time part. Depending on the interval size either only the date part of an activity is taken into account or the complete time attribute. All activities are mapped, based on their time attributes, to time slots over the period of time. If an activity has a duration greater than the interval of a time slot or the activity is recurring, then references to this activity are stored in the other time slots. The gained data structure is a map of time values to a list with assigned activities.

6.3.3 Visual Mapping

In this stage the calendar data is processed to create the calendar heightfield. The period of time and the interval for a time slot define the grid of the heightfield. Each time value of the map structure corresponds to a cell on the grid. The map structure is traversed to calculate the workloads (see section 3.1) for each time slot. If the calendar data includes more than one category, then the workloads are calculated in detail for each category. On the one hand, the workloads are stored in a list for further processing, such as in fuzzy scheduling. The index of the list corresponds to each cell in the grid and therefore to each time slot value. On the other hand, a workload defines the height of a block on the grid cell. A block is built up of polygons. Each block is colored by the corresponding category. Analogously, the block is divided by the number of categories in a time slot and each part represents the corresponding category through height and color.

The calendar heightfield is created and all workloads are known. The fuzzy tasks can be inserted into the calendar heightfield. Therefore, the algorithm in section 4.3.1 is used. Furthermore, the statistical representations of the calendar heightfield are prepared.

6.3.4 Rendering

The last stage is responsible for rendering the data, especially the calendar data. The labels of the axes are also drawn. The interaction and visualization techniques are tightly coupled.

When interacting with the calendar heightfield, detail information about the workloads can be retrieved. Therefore, the picked block and the grid cell are determined. The information about the time slot is queried from the map structure and shown in an overlay.

The importance view is updated every frame, if the viewpoint changes and depends just on the blocks and the grid of the heightfield. Therefore, the algorithm in section 4.2 is used.

Fuzzy scheduling returns feedback about the current distribution of workloads and updates the shrinked or raised blocks. The distribution of fuzzy tasks is done by the algorithm in section 4.3.2. Therefore, the list with workloads is traversed to receive the loaded time slots including fuzzy tasks. Each workload value is compared to the workload limit. If a block should change its height, then $UpdateBlockGeometry(pos, load)$ is used. Only four vertices have to be updated, if the height of a block with one category should be changed. For more than one category, it is necessary to know the category of the distributing fuzzy task and possibly more than four vertices have to be updated.

The visualization of advanced statistic views is coupled to the viewpoint of the camera. It is possible to use keyboard shortcuts to switch exactly in a front, side or top view. The statistics are only updated, if the workloads change their distribution, for example after fuzzy scheduling.

Chapter 7

Results

The following chapter focuses on the presented visualization and interaction techniques on the basis of real world data. The examples in the figures of the theory, in the previous sections, are consciously chosen, to emphasize on the techniques and not on the data. Here, the same strategy is applied. Certainly, only one data set, a student's schedule over a period of 18 weeks, is chosen. This is done, to show that the used techniques do not depend on specific data sets. The implemented prototype is used to present visualization results.

The results are structured in a logical sequence. First, the examination of the heightfield visualization of calendar data is performed. In this case, the demonstration of interaction techniques is also important. Then, the fuzzy scheduling is determined. The analysis of the data for patterns or trends is also presented. Finally, all obtained results are discussed and compared to other applications.

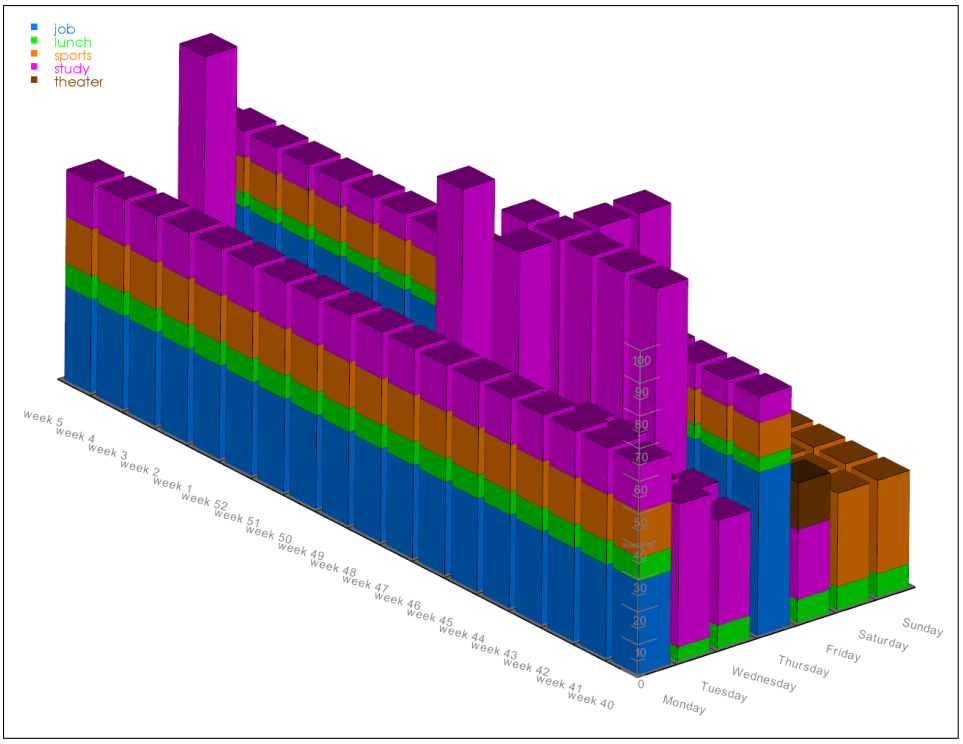
7.1 Visualization and Interaction

The heightfield visualization of calendar data is well suited to present the distribution of workloads over periods of time. This is visualized in the figures 7.1 and 7.2. The workload distribution of the student's schedule example is comparatively constant over the period of time, but there are some hidden features that can be explored with the heightfield visualization.

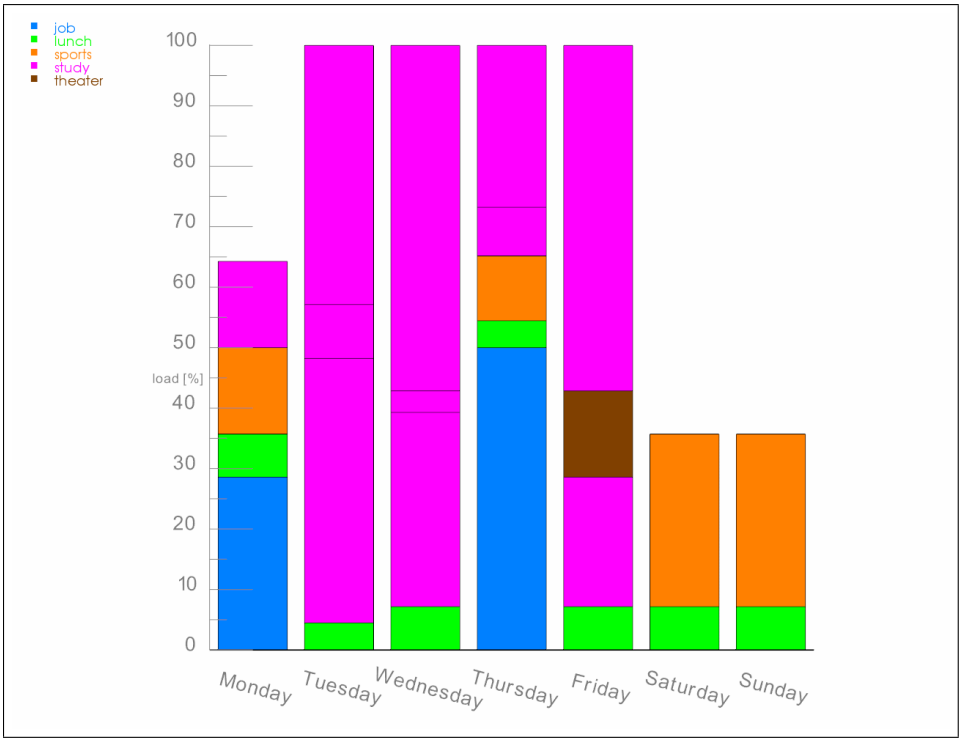
Figure 7.1(a) shows the schedule in a default 3D view between the 10th of October 2007 and the 3rd of February 2008. The timespan for visualization is selectable by definition of the start and end time, hence there are no restrictions in fixed size views. Another free parameter is the time interval for one time slot. The interval starts from 08:00 in the morning and ends at 22:00 in the evening. Only activities, which fully or partially overlap this interval, are taken into account. The block size to represent a 100 percent workload is currently fixed, therefore the range of the interval effects the representation of the workload blocks.

A distinct color represents a category of events. The colored blocks give a feedback on the distribution of each category over the period of time. For example, category ''job'' (colored light blue in figure 7.1) takes place every Monday and every Thursday, while category ''theater'' (in brown) occurs on Friday. The 3D view (see figure 7.1(a)) gives an overview of the workload distribution and some structures can be recognized. For example, is ''theater'' a recurring event similar to ''job''? To show this information, interaction techniques are used in order to change the viewpoint or to retrieve some detail information. ''Theater'' is the last category (in the sorted list of categories), therefore it is the top most workload block. The fastest way to find all other recurrences of ''theater'', if there are any, is to look at the calendar heightfield in top view (see figure 7.1(b)). The result is, that ''theater'' occurs every third week on Friday. To explore hidden or occluded data, the techniques in section 3.3, 3.4 and 4.2 have to be used to retrieve detail information or find dependencies to other activities while viewing an interesting block.

The front, side and top view (see figures 7.1(b), 7.2(a) and 7.2(b)) of the calendar heightfield visualization can be used for initial analysis. The front and side views show a cumulative bar chart, representing the time slots along a time dimension. Therefore, the front view shows the maximum workloads for each weekday, while the side view displays the maximum workloads for each week. Free time slots (i.e., the entire day is free) can be found very fast in the top view. There is much more information in depth for all of these views, which are described later.

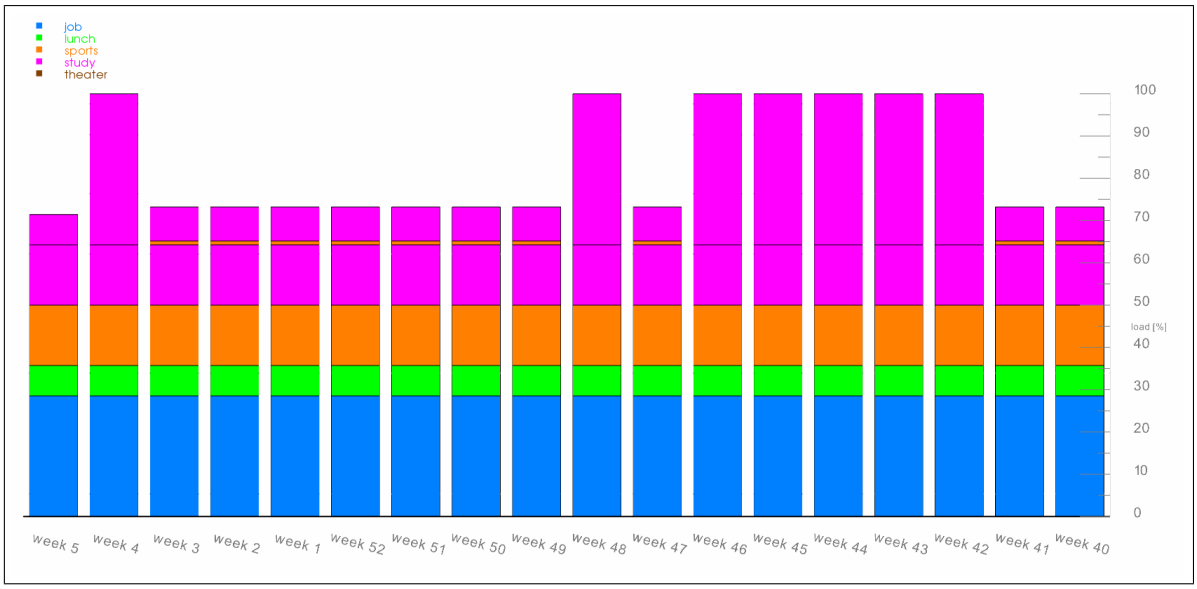


(a)

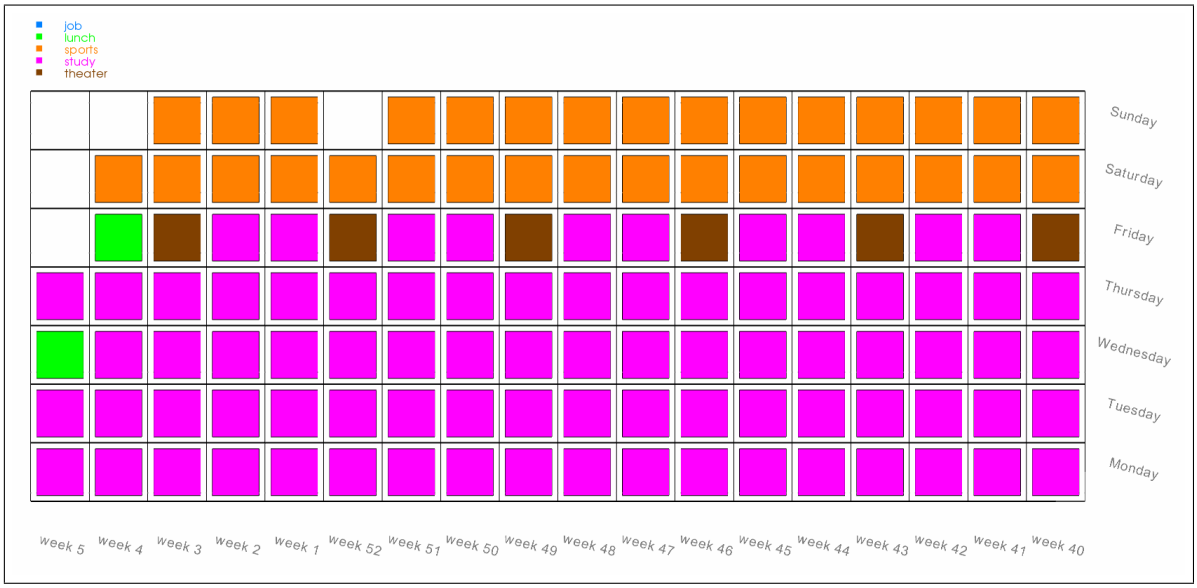


(b)

Figure 7.1: Heightfield visualization views showing a student's schedule between the 10th of October 2007 and the 3rd of February 2008: (a) shows the workload distribution in a default 3D view, (b) shows the maximum workloads for each weekday over this period of time in a bar chart.



(a)



(b)

Figure 7.2: Heightfield visualization views of the student's schedule: (a) the side view shows the maximum workloads for each week in a bar chart, (b) the top view of the calendar heightfield shows some patterns.

7.2 Fuzzy Scheduling

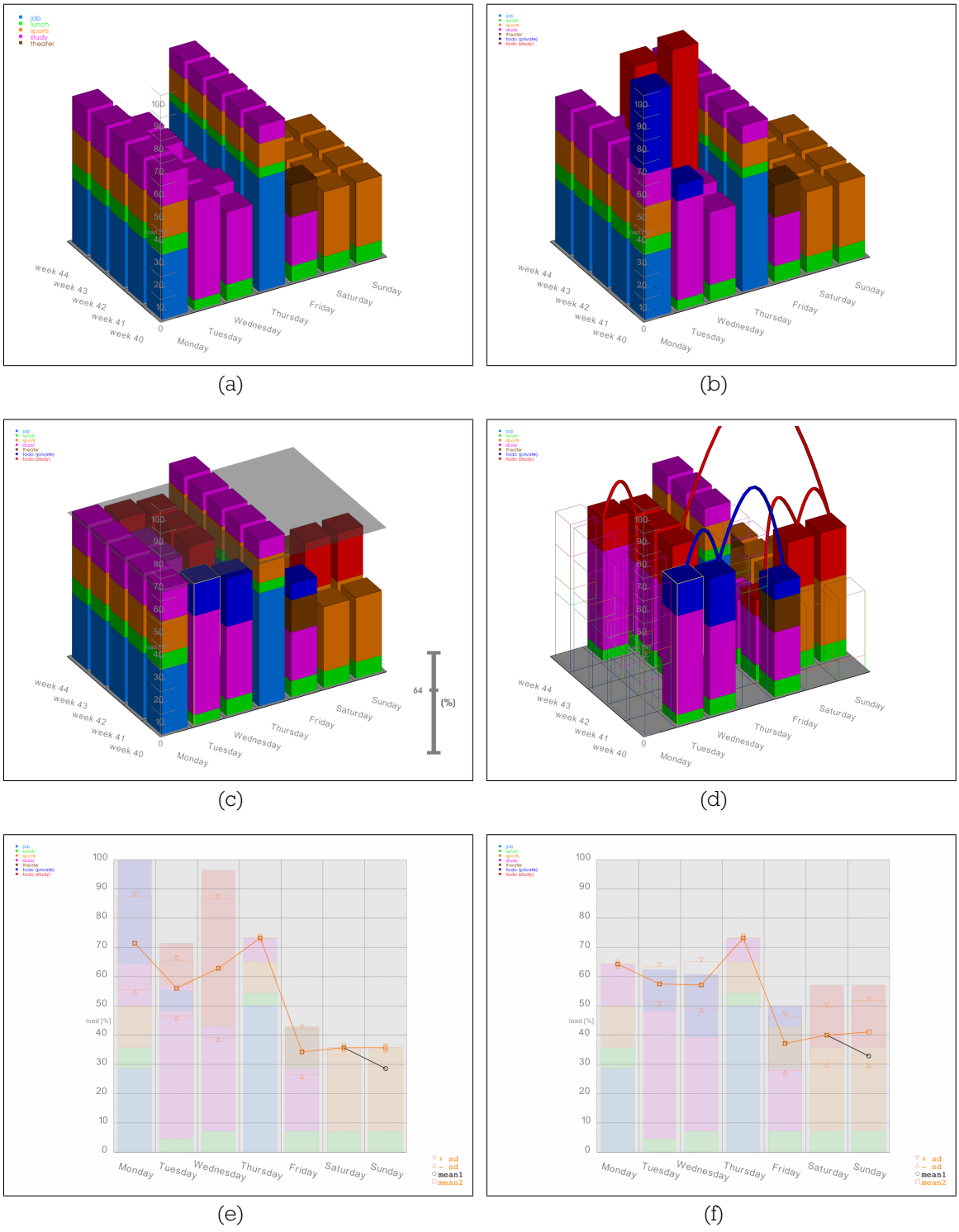
In section 4.3 it was mentioned how fuzzy tasks are inserted into one's calendar and how they are distributed if the overall workload changes. The following storyboard (see figure 7.3) presents the complete process of allocating fuzzy activities, distributing them, showing the new positions and how the workload distribution is affected. A small section of the student's schedule example has been used. First of all, the following activities have to be allocated.

- three fuzzy start tasks (e.g., delivery of homework) with ten hours, three hours and five hours estimated effort and one hour cluster size
- one fuzzy start and end task (e.g., cleaning the apartment) with six hours estimated effort and half an hour cluster size

For this process, the algorithm in section 4.3.1 is applied. Figure 7.3(a) shows the calendar heightfield in a default 3D view. In figure 7.3(b) the fuzzy tasks are inserted with their degrees of freedom. The tasks have their own categories and thus different colors to distinguish them better from other events. There are three blue blocks representing the fuzzy start tasks and one brown block for the fuzzy start and end task. If a fuzzy task needs more space than is available, it must be moved or split. For example, the brown task is divided onto two time slots. The first time slot is filled up fully, while the second time slot is filled up by the rest of the estimated effort. There are fully loaded days for the time interval between 08:00 and 22:00.

With fuzzy scheduling it is possible to distribute the fuzzy activities. The process is described in detail in section 4.3.2. The implemented prototype uses a slicing plane to detect the loaded time slots by comparing the workloads of each time slot with the workload limit specified by the position of the plane. The plane is controlled by a slider bar. By moving the slider, the fuzzy tasks are distributed over the calendar heightfield. The result is shown in figure 7.3(c). The fuzzy tasks have been flooded below the 64 percent mark. To recognize where they exactly have been moved, the visualization of groups through arcs (see section 3.4) in combination with importance visualization (see section 4.2) can be used (see figure 7.3(d)). The brown fuzzy start and end task has been moved completely away from Monday to Tuesday and Wednesday. The rest of the estimated effort is gone to Friday, because the workload limit is already below the workload of Thursday. The 'flood-scheduling'' is done as long as the requirements (in section 4.3.1) are fulfilled and the plane is moved.

Fuzzy Scheduling is done interactively, all changes can be seen immediately for fast feedback. After the geometry updates and workload distribution, the statistical representations are also recalculated. The last two figures 7.3(e) and 7.3(f) show the workload distribution before and after the fuzzy scheduling process in advanced statistic views (see section 5). It is possible to see what has happened during the scheduling process. How to analyze the statistical representations of these figures is explained in section 5.2 and 7.3.



(a) (b) (c) (d) (e) (f)

Figure 7.3: Heightfield visualization with fuzzy scheduling of the student's schedule: (a) shows the calendar heightfield without fuzzy tasks, (b) shows the allocated fuzzy tasks (red ones are fuzzy start, the brown one is fuzzy start and end), (c) shows the fuzzy scheduling process with the transparent slicing plane which is controlled by the slide bar, (d) shows the scheduled fuzzy tasks highlighted in a combined view of showing groups through arcs and importance visualization, (e) shows the workload distribution before fuzzy scheduling, (f) shows the workload distribution after fuzzy scheduling.

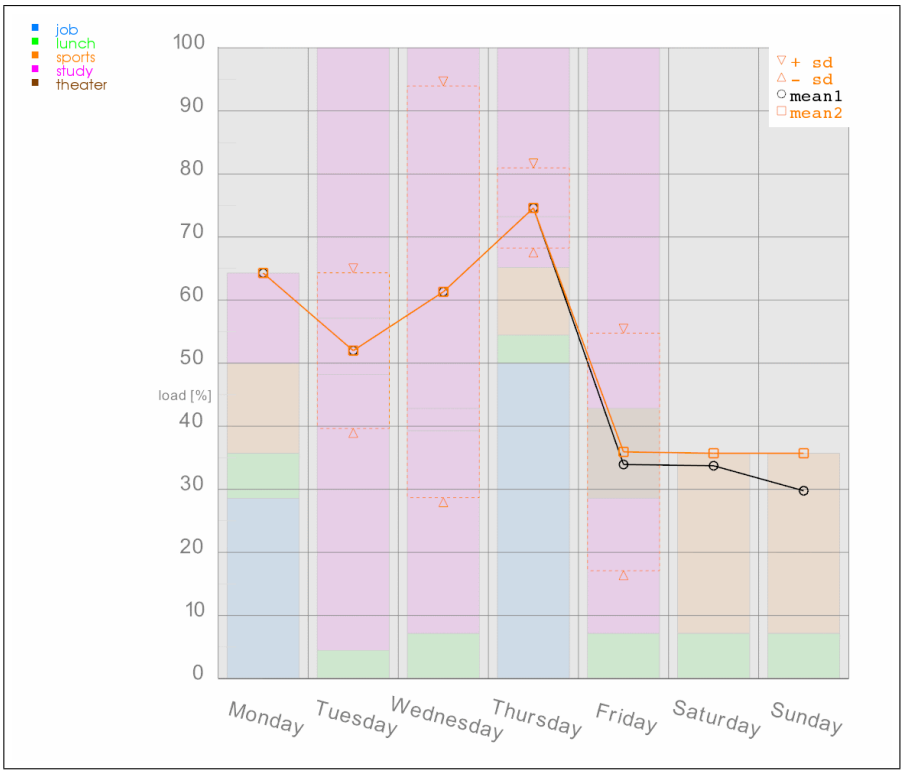
7.3 Patterns and Trends

The heightfield visualization of workloads from calendar data is appropriate to explore the data for patterns and trends. As mentioned in section 5 the projection of the calendar heightfield in front or side view results in a cumulative bar chart, that shows the maximum workloads of each time slot along a time dimension. Here, the main focus lies on analyzing the advanced statistic views of the student's schedule.

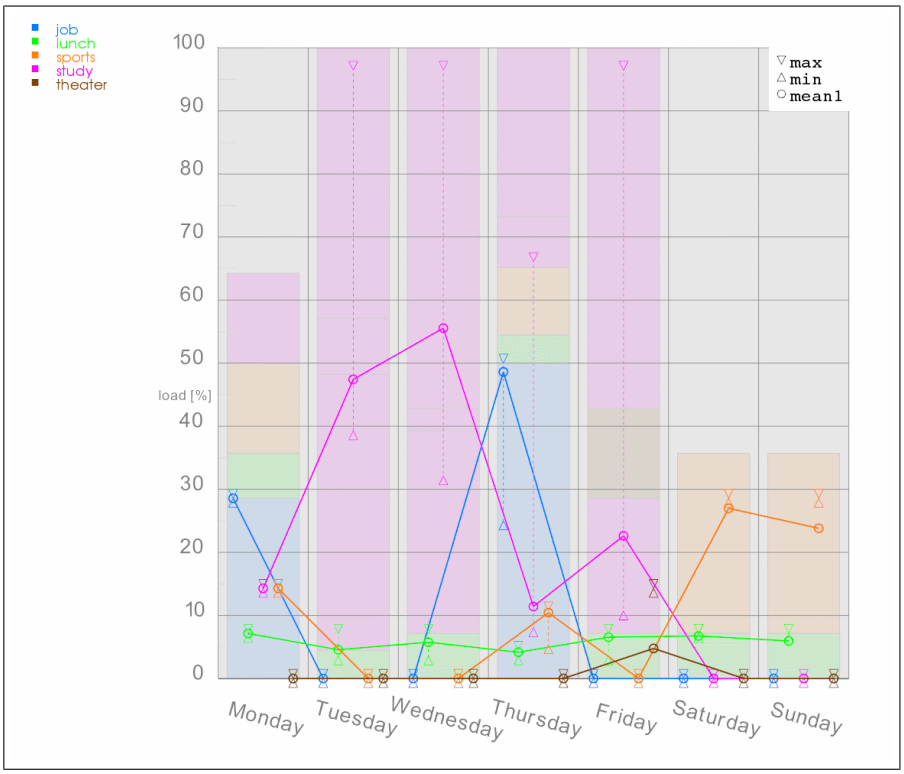
Figure 7.4 shows trend views of the calendar heightfield. As described in section 5.2, there are two ways to accumulate the workloads. The orange polyline (abbr. in the figure as mean2, symbolized with a square) in figure 7.4(a) represents the arithmetic mean of the observed time slots. The standard deviation (abbr. in the figure as sd, symbolized with a triangle) around the arithmetic mean is shown by a dashed box in orange. The black polyline (abbr. in the figure as mean1, symbolized with a circle) depicts the arithmetic mean of the total number of time slots. Both curves use the overall workloads of each time slot for calculation. If the curves lie on top of each other, there are no entirely free time slots. In this example, the black curve has a small offset on Friday, Saturday and Sunday. With a little experience, it is possible to determine how many entirely free time slots are encoded by the size of an offset between these curves. Friday and Saturday each have one and Sunday has two entirely free time slots. The weeks, in which the entirely free time slots occur, are determined later.

The curves of figure 7.4(a) show the averaged trend between each time slot independent of the categories. Figure 7.4(b) shows the arithmetic mean for each category. The color of a curve corresponds to a unique category. The range (interval of minimum and maximum occurrence) of each category is shown as well. It is defined by the dashed line between the upper and lower triangles. The curves are also build up of the total number of time slots. Therefore, it is possible to detect gaps in the series, if the arithmetic mean does not lie within its range. Apart from the analysis of trends and estimation of entirely free time slots, overlapping occurrences for recurring events can be found. For example, the lunch (the green curve) happens every day at the same time. Based on the curve, irregularities appear to the user. There are overlapping activities on Tuesday, Wednesday, Thursday and Sunday.

The side view offers the same possibilities and is used to determine the weekly trends (see figure 7.5). With both views in combination, the front and side view, the analysis is much easier and the exact position of a time slot can be estimated, for example the entirely free time slot on Sunday in the week 52.

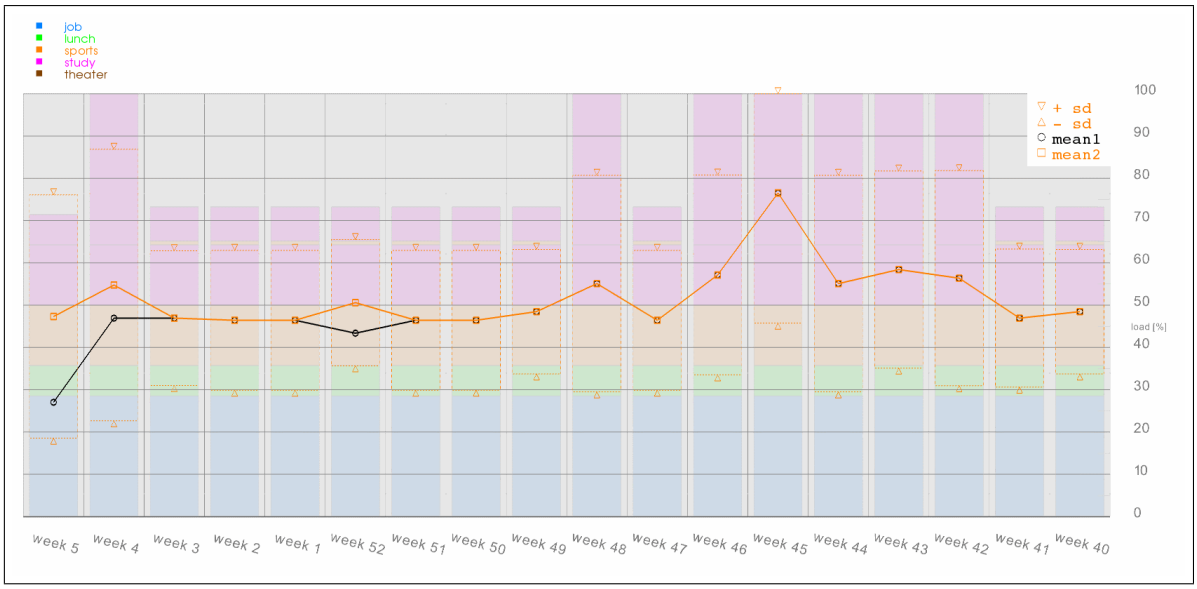


(a)

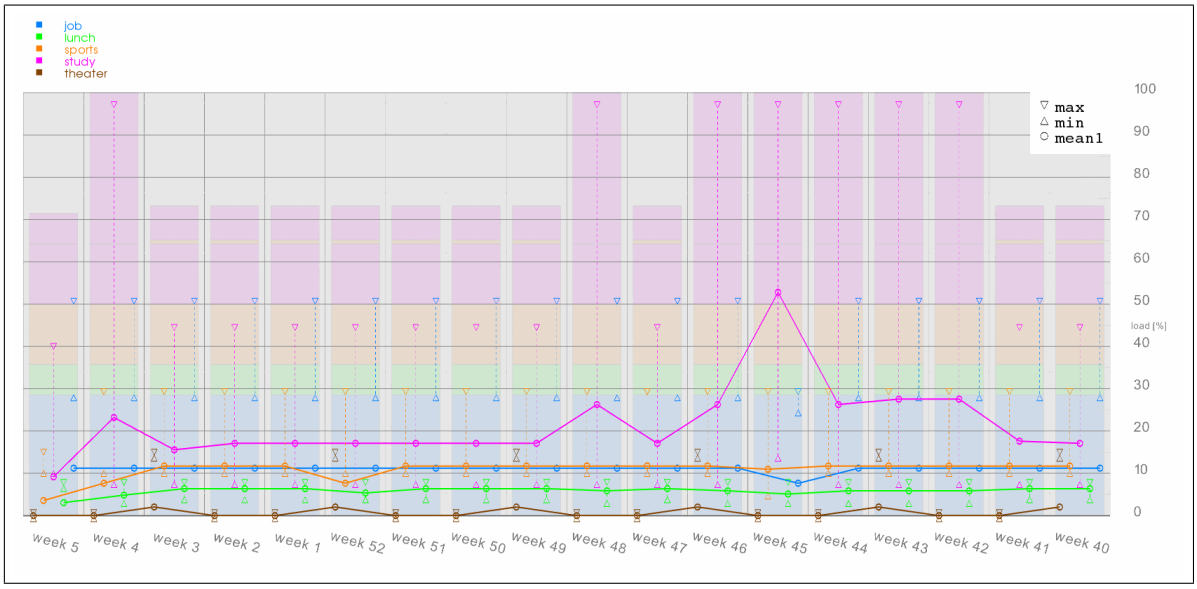


(b)

Figure 7.4: Trend views of the heightfield showing the student's schedule: (a) shows arithmetic mean curves and standard deviations for all weekdays, mean1 includes the total number of time slots along the time dimension, while mean2 represents only the number of observations, (b) shows arithmetic mean curves and ranges of each category for all weekdays.



(a)



(b)

Figure 7.5: Trend views of the heightfield showing the student's schedule: (a) shows arithmetic mean curves and standard deviations for all weeks, mean1 includes the total number of time slots along the time dimension, while mean2 represents only the number of observations, (b) shows arithmetic mean curves and ranges of each category for all weeks.

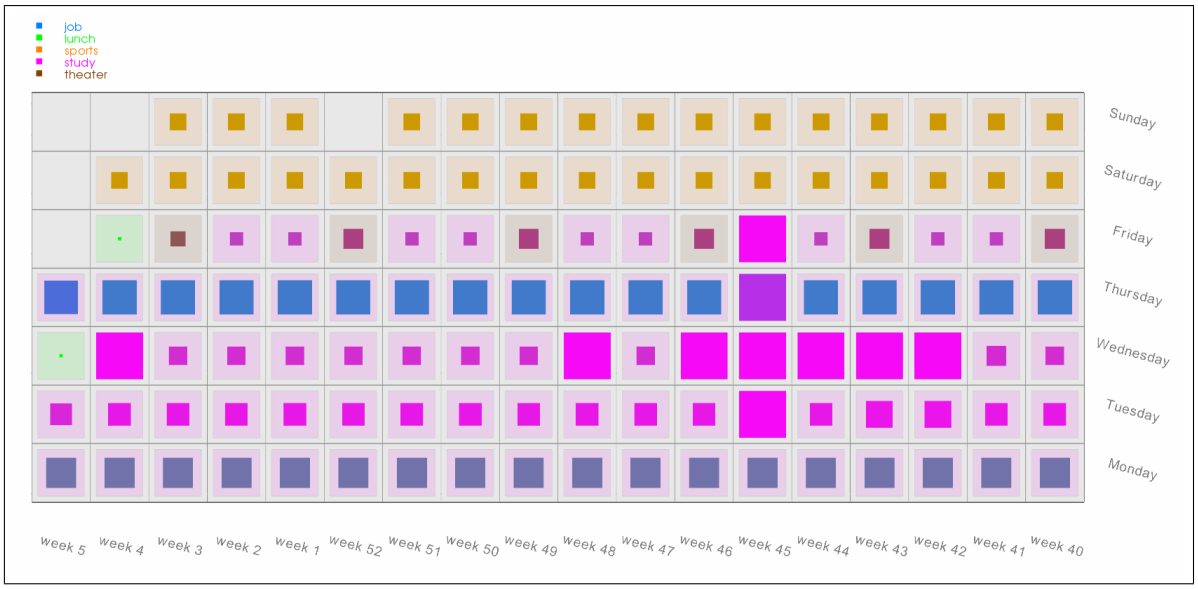


Figure 7.6: Pattern view of the heightfield showing the student's schedule. The quadrangles of each time slot give information about the underlying workload and the involved categories.

The pattern view (see section 5.3) of the student's schedule is shown in figure 7.6. The pattern view is used, for example, to find entirely free time slots at once, such as the Sunday in week 52. The workloads of each time slot are shown by the size of the quadrangles. A fully loaded day will cover the underlying block entirely, but not the cell as there is a predefined margin. The involved categories of a time slot are represented by color blending. Thursday is dominated by the category ''job'', while Tuesday and Wednesday are booked for the category ''study''. Also recurring events can be estimated. See the category ''theater'' (in cyan) that is recurring every three weeks on Friday. Symmetries can be found very fast.

7.4 Discussion

The above presented results show the sophisticated interaction and visualization techniques with calendar data on real world data. The gained results has been performed on a common personal computer (Intel Core 2 Duo E4300 with 2x 1.8GHz including 2MB shared L2-Cache and 1GB of main memory). During testing no delays have been recognized, all processes are done in real-time. Although, performance tests are not focus of this work, the bottleneck in the implementation of the prototype is the rendering part. Searching for workload values in a list and determining the fuzzy tasks is not a challenge for a CPU. But updating all changed blocks and rendering the heightfield affect the computational effort. Presenting periods of time over 5 months and more are also possible, therefore, the 3D visualization has to be extended by further

visualization and interaction techniques. Some ideas are considered in chapter 9.

The calendar heightfield offers the possibility to show calendar data in such a manner, that it can be used for exploration and scheduling at the same time. The focus lies on the visualization of workload distribution, but detail information for comparison can also be shown. The representation of the observed period of time is variable as well as the interval for one time slot. Therefore, the view of the calendar heightfield is not fixed. The heightfield data can be viewed and analyzed from different viewpoints. Recurring activities and entirely free time slots can be easily recognized.

Common calendar applications only offer static views, but different representations of data and periods of time. They make use of search algorithms to find specific data, while the heightfield calendar is able to do this also in an exploratory and visual way. There are appropriate and easy to use interaction techniques involved, to explore and manipulate the calendar heightfield.

The comparison of events and the scheduling of fuzzy activities are significant features. In common applications it is not possible to treat tasks with uncertain time attributes. However, fuzzy scheduling can be used to combine events and tasks into one schedule. This can help to schedule more effectively and faster than with normal applications. Furthermore, the workload distribution can be changed dynamically.

All these features can be used in combination with analyzing patterns and trends. Therefore, statistical interpretations of the calendar data, as a whole or specifically by category, can be shown. Finally, the calendar heightfield can be used to exchange the data with common calendar applications.

Chapter 8

Summary

In this chapter a short summary of this thesis is given.

8.1 Introduction

Information visualization deals with the representation of mostly abstract non-physical data. In this thesis the main focus is on the visualization of calendar data, which is a subset of time-oriented data. The visualization of time-oriented data occurs in many domains and, therefore, different visualization and interaction techniques have been developed.

To visualize large data sets on small screens, focus+context techniques are often used. For example, long periods of time, such as time lines, can be efficiently explored with these techniques. However, daily or seasonal patterns are almost not visible. Therefore, spirals are used to analyze serial and periodic behavior. Large data sets are often abstracted and clustered. On the one hand, relationships and other dependencies can be determined. On the other hand, correlations between the data and the time can be explored. Analysis techniques are used to recognize patterns and trends.

The visualization of calendar data is always a challenging task. Software applications for scheduling one's calendar are very popular. Most of them are similarly built, including the visualization of events and tasks as well as their interaction techniques. They use different views to visualize different periods of time. To-do's are displayed in a separated view. To compare schedules of a group of persons some applications have the possibility to switch to so-called collaboration views. Only few of these applications make use of information visualization techniques to explore calendar data.

8.2 Heightfield Visualization of Calendar Data

One possibility to visualize calendar data over a large period of time in a compact manner, is to split up the time dimension and map it on more than one axis. Each axis represent a time dimension and the interval of one axis is represented through another, but with a different resolution. For example, one axis represent weeks, while the other one shows the weekdays. This method, extended by another dimension, is used to visualize calendar data in 3D space. The first two axis show the time dimensions, while the third one represent workloads. This visualization of calendar data results in form of a heightfield, which makes it possible to see the workload distribution at a glance.

The workload is a scalar value, that represents the overall duration of activities for a specified time interval. To determine all workloads over a period of time, the calendar data is clustered into time slots. Then, all durations of activities inside a time slot are unified and weighted by the time interval of this time slot. This is done for all time slots. Each workload for a specific time slot is represented by the height of a block in the heightfield. To position the blocks in the heightfield, a regular grid is used. This 2D regular grid defines the two time dimensions and is spanned up by the vectors x and y of the Cartesian coordinate system. The xy-plane is partitioned to correspond to a regular grid. Each cell represents a time slot and an assigned height value (i.e., the workload in percent). As a result, the workloads can be compared with each other.

Color is used to get more insight into the calendar data. It is a distinctive feature for different categories of one calendar data or for distinguishing different calendar data sets in collaboration views. To visualize the categories of one data set, the activities of each time slot are further subdivided by their categories. This results in a subdivided workload block, each part represents the workload of one category. Each category is assigned a unique color either manually or randomly. The categories are sorted alphabetically. Therefore, the colored parts are also displayed in the same order.

Per default, the height of each block corresponds to the workload of that time slot. It is possible to switch from a workload view to a detail view per block. The block becomes invisible and a time scale appears in the middle of the cell. Each event is displayed by a block, while the position and the size represent the start and end time of it. The blocks are drawn transparently, to see the time scale in the middle and to recognize overlapping events. It is also possible to compare neighboring time slots in detail view, but the comparison is very limited. The detail view is dynamically displayed and it points always to the viewer. It is visible from almost every viewpoint. For example, the bottom view does not offer an insight into the calendar heightfield.

They workloads in the calendar heightfield are visually grouped by the correlation of categories through colors, although many activities have nothing in common. For

example, to find recurring events or ones which depend on others, it is sufficient to pick only one of them. As a consequence, all blocks which belong together will be connected through arcs. The color of an arc corresponds to the category of this group.

8.3 Interaction on the Calendar Heightfield

Interaction techniques have to be involved to explore a complex data set in 3D space, such as the heightfield visualization of calendar data. Besides the default transformations in 3D space and zooming, other interaction methods are available to interact with the calendar heightfield.

It is possible to highlight and select blocks in the heightfield. Highlighting is the default visualization effect to navigate through the calendar heightfield and to retrieve some information. A block in the heightfield becomes highlighted through a colored wireframe, which encloses it. The corresponding start and end time (i.e., the position in the time interval) and the workload are displayed on the labels of the axes. The visualization of the highlighted block or the display of detail information are triggered by interaction events of the mouse pointer.

An importance-driven visualization technique is used to emphasize a block which is occluded by others. Therefore, an algorithm has been developed, which determines the occluders in front of an interesting block. The occluders are then drawn in wireframe mode, in order not to loose the context. This method depends on the viewpoint and it is immediately updated if the position of the camera has changed.

Calendars are often used to schedule tasks. A task differs from an event in a way, that the time attributes (i.e., start and end time) can be missing. The calendar heightfield can treat such temporal uncertainties of tasks. A task with at least one missing time attribute is said to be fuzzy. The types of fuzziness are described as follows: fuzzy start (i.e., only the end time is specified), fuzzy end (i.e., only the start time is specified) and fuzzy start and end (i.e., both times are missing). To integrate a fuzzy task it is necessary to define another time attribute: an estimated temporal effort. It is often not possible to specify an exact start and end time for a task, but the user often can imagine how long a task will take until it is completed. A lot of experience is needed.

A fuzzy task with an estimated effort can be integrated into the calendar heightfield. Depending on the type of fuzziness, the other time attribute can be calculated. Tasks, that do not have a start nor an end time are assumed to be fuzzy on the end time. As long as such a fuzzy task is not in process, it can be started earliest by now. A fuzzy task is similar positioned in the calendar heightfield as an event. Therefore, an algorithm has been developed. This algorithm determines the first not fully loaded time slot next to the given time attribute. The search direction depends on the type of the fuzzy task. If the start time is given, a position in the future is searched. Analogously

for an end time, a position before the end time has to be found. It is not possible to seek for not fully loaded time slots in the past. If a position has been found, the fuzzy task is inserted into the time slot. If the globally defined workload limit, specified by the user, has been reached, then the fuzzy task is split up into parts. These parts are then distributed on further time slots as long as all parts are inserted. The possibility to split up a fuzzy task is given by another attribute in time units: the cluster size. This attribute is also specified by the user and it defines the smallest time unit a task can be partitioned into. If a fuzzy task should not be split up, then the algorithm tries to find a time slot where the task fits entirely, but it is possible, that such a task is never inserted. To define a special order in which the fuzzy tasks should be inserted, a priority can also be specified by the user.

The allocation of fuzzy tasks is one part of fuzzy scheduling. The second part is a fully interactive scheduling method, to change the distribution of workloads by balancing the fuzzy activities in the heightfield. A ''flood-scheduling'' algorithm has been developed. This algorithm is linked to a plane above the calendar heightfield. If the plane is moved downwards and intersects blocks with fuzzy activities, then these activities are spread over the time slots if possible. The parts of fuzzy tasks flood from higher to lower workload blocks. The flooding direction depends on the fuzzy type. The maximally possible distribution depends on the data set and it is reached, if no fuzzy task can be flooded anymore. To visualize where the parts of a fuzzy task has been distributed, the method to visualize groups or dependencies of activities, by connecting them through arcs, is used.

8.4 Advanced Statistic Views

Searching for visual structures while analyzing the data is an important task in information visualization. With advanced statistic views it is possible to find patterns and trends in calendar data. To represent statistical representations of the calendar data, semantic views are used. Such a view focuses on the meaning of the data in a specific viewpoint, to represent or interpret the underlying data in statistical manner. There are three useful semantic views, positioned in front, side and top view of the calendar heightfield. The views are displayed in the calendar heightfield view and are drawn transparently, not to loose the context.

Trends can be analyzed by viewing the descriptive statistical results of the projected time dimensions. The results for each view are calculated by traversing the calendar data along the projected time dimension. Several parameters, such as the minima, maxima and arithmetic mean are processed for all workloads of each time slot. It is also possible to take the categories for subdivided results into account. The results are visualized as line graphs in the corresponding semantic view. By interpreting the

line graphs, trends of distributed workloads over a specific time dimension can be observed.

The projection of the calendar heightfield in the workload dimension (i.e., top view) results in a pattern view. Each workload of a time slot on the grid is interpreted as a quadrangle. The overall workload of a specific time slot is represented by the size of this quadrangle. A quadrangle which covers the underlying workload block entirely encodes a 100 percent workload. The ratios of the categories of one specific time slot to the overall workload are encoded by the color of the quadrangle. Therefore, color blending is used. The quadrangles on the pattern view are representative for the calendar heightfield data and can be used to detect visual patterns. For example, similar loaded time slots or free time slots can be detected.

8.5 Results

To get an impression of the functionality of the proposed interaction and visualization techniques of calendar data, the implemented prototype has been used to retrieve results. As an example, a data set of a student's schedule has been analyzed.

The basis of heightfield visualization and interaction shows the possibility to represent a large period of time of calendar data as a workload distribution. While interacting with the calendar heightfield, hidden or some interesting features can be found. This process is supported by sophisticated methods for exploration. The color and the size of blocks represent events and tasks of categories with workloads. The front and side view of the calendar heightfield corresponds to a bar chart, showing the maximum workload distributions of all time slots along each time dimension.

Fuzzy Scheduling shows its strengths by the allocation of fuzzy activities into the student's calendar and by scheduling them in dependence of the workload distribution. Therefore some uncertain tasks are integrated into the data set of the student. The ''flood-scheduling'' algorithm is used to shrink the loaded days by flooding the fuzzy tasks. This process is done interactively. The results of the movements and the workload distribution are updated in ''real-time''.

The student's schedule has also been analyzed for patterns and trends. The insight into the data is gained by advanced statistic views. Semantic views are used to show advanced statistic information, such as the overall and categorical workload distribution of each time slot along a time dimension. This supports among others the recognition of patterns, finding entirely free time slots or overlapping occurrences. Symmetries and recurrences can be found very fast in the pattern view.

Chapter 9

Conclusions and Future Work

In this thesis sophisticated visualization and interaction techniques for calendar data are presented. The heightfield visualization is an intuitive metaphor. It is well suited to present workload distribution over periods of time. Many techniques have been developed to support the process of visualization and exploration. Color is used for categorization and for depicting the ratio of each category with respect to the overall workload. Detail information from the time slots can be shown and compared with others, which could be very helpful for scheduling. The connections of groups and dependencies are explained for faster recognition visually. The definition, allocation, and scheduling of fuzzy tasks are fundamental tasks to treat temporal uncertain activities. This could be a popular feature in calendar applications. Advanced analysis features are provided to see patterns or trends in semantic views, which show statistical representations. All of these techniques can be used alone, but in combination they will show their strengths.

In this thesis, the time dimension is always represented to be linear. In order, to analyze periods of time over several years, logarithmic representations of the time dimension are better suited, possibly in combination with fisheye views or other focus+context techniques to show the distortion in time.

To determine the time slots of only one specific category, it would be helpful, if other categories can be hidden on-demand. For interaction, efficient zooming and panning can be used to follow recurring or grouped activities with great distances in between. Another interaction technique can be used to move blocks, including activities, among time slots and their intervals. Furthermore, a selection tool can be implemented, which is deformable and movable in the Cartesian coordinate system to slice the heightfield or to consider only a section of it. Another possibility is to walk along a time dimension from one time slot to the next, for example, to go through the weeks.

Fuzzy Scheduling is an interactive method to change the distribution of workloads on-demand. There are many possibilities for the distribution algorithms, for example, a asymmetrical or Gaussian distribution. The estimated effort for an activity can

be predefined automatically by using an artificial intelligence system for the analysis of workloads in previous activities. There are many more possible parameters to simplify the allocation of fuzzy tasks. For example, it would be helpful, to specify the possible or desired days and/or times where the fuzzy activity should be placed. The boundaries of fuzzy tasks could be more flexible to specify an earliest and a latest start or end time. Another parameter can be used to help scheduling meetings within a group of people.

The semantic views currently show descriptive statistics, however, it is possible to create other statistics. The calendar heightfield could be represented by a statistical isosurface, created through the curves of both trend views. This would look like a real landscape. Fuzzy Scheduling can be used in combination with semantic views to smooth or to rough the workload distribution of several activities within a category along one time dimension.

Finally, when thinking about the presented visualization and interaction techniques, shown by the calendar heightfield, the impression arises that they fulfill all the claims presented in table 2.1 (see section 2.4). However, extensive tests with real world data and real world users have to be performed and evaluated, to find all strengths and weaknesses and to optimize the visualization of calendar data.

Acknowledgments

I would like to thank Matej Mlejnek for his great support from the beginning until the completion of this thesis with ideas and motivation. I further thank Eduard Gröller for his supervision and quick support. Furthermore, I would like to thank my current boss at work, Leopold Wagner, for his understanding, that I was concentrating on my master's thesis. And, many thanks to my girlfriend, for her patience and proofreading.

List of Figures

Bibliography

[1] Wolfgang Aigner, Silvia Miksch, Wolfgang Müller, Heidrun Schumann, and Christian Tominski. Visualizing time-oriented data - a systematic view. *Computers and Graphics*, pages 401–409, 2007.

[2] Wolfgang Aigner, Silvia Miksch, Bettina Thurnher, and Stefan Biffl. Planninglines: Novel glyphs for representing temporal uncertainties and their evaluation. In *INFOVIS '05: Proceedings of the Ninth International Conference on Information Visualization*, pages 457–463, Washington, DC, USA, 2005. IEEE Computer Society.

[3] Tomas Akenine-Möller and Eric Haines. *Real-Time Rendering*. AK Peters, Ltd., Wellesley, Mass., second edition, 2006.

[4] Business Objects an SAP Company. Inxight Software Inc. is part of Business Objects since 2007, `http://www.businessobjects.com`, as of October 2008.

[5] Benjamin B. Bederson, Aaron Clamage, Mary P. Czerwinski, and George G. Robertson. Datelens: A fisheye calendar interface for pdas. *ACM Transactions on Computer-Human Interaction*, pages 90–119, 2004.

[6] Georg Bol. *Deskriptive Statistik*. Oldenburg, München, Wien, 1993.

[7] Eric Busboom, Art Cancro, Omar Kilani, and Dave West. Free association (libical). `http://freeassociation.sourceforge.net`, as of October 2008.

[8] Stuart K. Card, Jock D. Mackinlay, and Ben Shneiderman, editors. *Readings in information visualization: using vision to think*. Morgan Kaufmann Publishers Inc., San Francisco, Calif., USA, 1999.

[9] Stuart K. Card, Peter Pirolli, and Jock D. Mackinlay. The cost-of-knowledge characteristic function: display evaluation for direct-walk dynamic information visualizations. In *CHI '94: Proceedings of the Conference Companion on Human Factors in Computing Systems*, page 216, New York, NY, USA, 1994. ACM.

[10] John V. Carlis and Joseph A. Konstan. Interactive visualization of serial periodic data. In *UIST '98: Proceedings of the 11th annual ACM Symposium on User Interface Software and Technology*, pages 29–38, New York, NY, USA, 1998. ACM.

[11] F. Dawson and D. Stenerson. Internet calendaring and scheduling core object specification (icalendar). http://tools.ietf.org/html/rfc2445, as of November 1998.

[12] James D. Foley, Andries van Dam, Steven K. Feiner, and John F. Hughes. *Computer Graphics Principles and Practice*. Addison-Wesley, Boston, Mass., USA, second edition, 1997. International Edition in C.

[13] Robert L. Harris. *Information Graphics: A Comprehensive Illustrated Reference*. Oxford University Press, second edition, 1999.

[14] Donald Hearn and M. Pauline Baker. *Computer Graphics, C Version*. Prentice Hall, Upper Saddle River, NJ, USA, second edition, 1996.

[15] K. Higgins, D. Miner, C.N. Smith, and D.B. Sullivan. A walk through time. http://physics.nist.gov/GenInt/Time/ancient.html, as of October 2008.

[16] Google Inc. Google calendar. http://www.google.com, as of October 2008.

[17] Inxight Software Inc. Timewall. http://demo.labs.businessobjects.com/VizServer/demos/sample-MovieBoxOffice.html, as of October 2008.

[18] Kitware Inc. Visualization toolkit. The current version of VTK is 5.2.0, which has been released on the 28th of August 2008, http://www.vtk.org, as of October 2008.

[19] Leonard Kaufman and Peter J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley, New York, USA, 1990.

[20] Jock D. Mackinlay, George G. Robertson, and Stuart K. Card. The perspective wall: detail and context smoothly integrated. In *CHI '91: Proceedings of the Conference Companion on Human Factors in Computing Systems*, pages 173–176, New York, NY, USA, 1991. ACM.

[21] Jock D. Mackinlay, George G. Robertson, and Robert DeLine. Developing calendar visualizers for the information visualizer. In *UIST '94: Proceedings of the 7th annual ACM Symposium on User Interface Software and Technology*, pages 109–118, New York, NY, USA, 1994. ACM.

[22] Stefano Mazzocchi. Timeplot. http://simile.mit.edu/timeplot/, as of October 2008.

[23] Microsoft. Microsoft office outlook. http://office.microsoft.com/outlook/, as of October 2008.

[24] Petra Neumann, Stefan Schlechtweg, and Sheelagh Carpendale. Arctrees: Visualizing relations in hierarchical data. In Ken W. Brodlie, David J. Duke, and Ken I. Joy, editors, *EG '05: Proceedings of the 2005 Eurographics / IEEE VGTC Symposium on Visualization*, pages 53–61, Aire-la-Ville, 2005. Eurographics Association.

[25] Gregory M. Nielson, Hans Hagen, and Heinrich Müller. *Scientific Visualization: Overviews, Methodologies, and Techniques*. IEEE Computer Society, Los Alamitos, Calif., USA, 1997.

[26] Kimmo Pekkola. Rainlendar. http://www.rainlendar.net, as of October 2008.

[27] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. *IEEE Symposium on Visual Languages*, pages 336–343, 1996.

[28] Calgoo Software. Calgoo calendar. http://www.calgoo.com, as of October 2008.

[29] Robert Spence. *Information Visualization: Design for Interaction*. Pearson-Prentice Hall, Harlow, second edition, 2006.

[30] Martin Suntinger, Hannes Obweger, Josef Schiefer, and Eduard Gröller. The event tunnel: Interactive visualization of complex event streams for business process pattern analysis. In *PACIFICVIS '08: IEEE VGTC Pacific Visualization Symposium*, pages 111–118. IEEE Pacific, March 2008.

[31] Claus Tøndering. Calendars through the ages. http://www.webexhibits.org/calendars/index.html, as of October 2008.

[32] Edward R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, Conn., USA, second edition, 2002.

[33] Edward R. Tufte. *Envisioning Information*. Graphics Press, Cheshire, Conn., USA, tenth edition, 2005.

[34] Jarke J. van Wijk and Edward R. van Selow. Cluster and calendar based visualization of time series data. In *INFOVIS '99: Proceedings of the IEEE Symposium on Information Visualization*, page 4, Washington, DC, USA, 1999. IEEE Computer Society.

[35] Ivan Viola, Miquel Feixas, Mateu Sbert, and Eduard Gröller. Importance-driven focus of attention. *IEEE Transactions on Visualization and Computer Graphics*, pages 933–940, Oct 2006.

[36] Ivan Viola and Eduard Gröller. Smart visibility in visualization. In L. Neumann, M. Sbert, B. Gooch, and W. Purgathofer, editors, *EG '05: Proceedings of EG Workshop on Computational Aesthetics in Graphics, Visualization and Imaging*, pages 209–216, May 2005.

[37] Robert Voigt. An extended scatterplot matrix and case studies in information visualization. Master's thesis, Magdeburg University of Applied Sciences Germany, Oct 2002.

[38] Colin Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann, San Francisco, Calif., USA, 1999.

[39] Marc Weber, Marc Alexa, and Wolfgang Müller. Visualizing time-series on spirals. In *INFOVIS '01: Proceedings of the IEEE Symposium on Information Visualization*, page 7, Washington, DC, USA, 2001. IEEE Computer Society.

[40] Günther Wyszecki and Walter Stanley Stiles. *Color Science: Concepts and Methods, Quantitative Data and Formulae*. John Wiley, New York, USA, 2000.