A Visualization System for Operational Meteorological Use

Miriam Lux, Thomas Frühauf
Fraunhofer Institute for Computer Graphics
Rundeturmstr.6, 64283 Darmstadt, Germany
mlux@igd.fhg.de

ABSTRACT

RASSIN is a system for professional meteorological visualization that has been developed by the Fraunhofer-IGD together with the German National Weather Service DWD. The software is being used in DWD's research department. Its scope is the visualization of numerical simulation data and measured data. This results in a very precise and highly interactive weather forecasting. Furthermore, the system serves the meteorologists in the evaluation of new forecast models. RASSIN follows a consistant strategy to perform calculation and visualization only on the original, irregular, dynamic grid of the meteorological data. The integration of a new concept for time-critical visualization allows scientific animations. The realized HCI-concept is tailored for users from meteorological sciences.

1 INTRODUCTION

In meteorology a correct interpretation of massive amounts of measured and simulated data is not possible without the techniques of scientific visualization.

For the visualization of meteorological data many systems are available, including turnkey systems and application builders. General purpose visualization systems like AVS, and IRIS Explorer are of limited use because they do not offer special meteorological techniques. Furthermore, the user interface cannot be sufficiently tailored to the special needs of meteorologists.

Several of the turnkey systems were designed for the presentation of meteorological data to a lay audience. Examples are TriVis [Schro93]], Triton i7 [Kavou96], and AccuWeather [Accu96, Zwirn95], to mention just a few. These systems are not applicable to scientific visualization and do not permit a scientific analysis of meteorological data.

Scientific visualization of meteorological data is possible with systems like McIDAS [Hibba89], MeteoVis [Weiha95], EarthWatch [Earth96], and Vis5D [Hibba89]. RASSIN, which is described in this paper, belongs to this group. Its scope is the visualization of numerical simulation data for weather forecasting. The concurrent goals are very precise *and* highly interactive visualization.

Our approach in RASSIN has the following key apects: a consistant calculation and visualization only on the original irregular, dynamic grid of the meteorological data, the integration of a new concept for time-critical visualization by scientific animation, and an HCI-concept, which is tailored to users from meteorological sciences.

To analyse meteorological simulation data efficiently, they should be visualized on their original, irregular grid. Interfaces to existing data bases have to be implemented in order to import the output of numerical simulations as well as observation data. Because dynamics is a key feature in meteorology, methods and tools for a scientific animation must be implemented [Encar93, Lux95]. The data must be prepared and handeled flexibly by the system so it remains interactive despite the huge amounts. For a correct visualization, data must be synchronized in time and in geographical space. All visualization techniques that are required by the meteorologists must be implemented. Last but not least, such a system must be easy-to-use and easy-to-learn for any user [Foley90].

RASSIN is implemented on X11 with OSF/Motif and on IGD's Vis-a-Vis rendering system [Frueh92]. Vis-a-Vis is based on OpenGL for fast image generation. RASSIN has been successfully installed on Silicon Graphics hardware varying from an Indy or Indigo^2 Maximum-Impact to a large Onyx Infinite Reality system.

2 METEOROLOGICAL DATA

For the calculation of weather simulations numerical forecast models are required. The German National Weather Service (DWD) is working with several different forecast models. The data from their European, their German and their new mesoscale local model can be imported and visualized by RASSIN. These models

Fraunhofer-IGD, Rundeturmstr.6., 64283 Darmstadt, Germany

have different extents and resolutions. The European model has 181*129*20 grid points with a resolution of 0.5 degree (approx. 50 km). The German model has 109*109*20 points with 0.125 degree (approx. 15 km) [Prome84]. The local model over Germany has 300*300*20 grid points with approximatly 2,5 km resolution [Doms95]. Simulations are carried out every two hours, forecasts are computed up to 78 hours in advance.

The models have an irregular and dynamic grid structure. Every grid point of the lowest layer describes exactly one geographical point on the earth, given in coordinates of longitude and latitude. Note that only the horizontal location of the 3D grid points is static. The vertical position is varible, depending on the computed local pressure! The meteorological data are computed for every 3D grid point. These points can carry scalar or vector information of many different variables [Prome81], e.g., pressure, temperature, wind direction and speed, specific cloud water content and many more [Damra92].

To accelerate the speed of the numerical simulation the computation is performed in a transformed grid. This grid has the convenient property of being Chapterequally spaced in horizontal orientation (see Chapter 3.1). The output of the numerical simulation is stored in this transformed grid using the so called GRIB database [EM91]. For every meteorological variable the values with a reference to the model grid are memorized.

In addition to the output of numerical simulations, observed data can be imported into RASSIN, as well. This kind of data is stored in the so called MAP database [Pogod97]. For every location – more then 8000 on land, sea and in the atmosphere – several meteorological variables are recorded. In seperate files the position for each station is given in longitude and latitude coordinates. Of course, these positions are not identical to nodes of the simulation grid. To visualize the measured data, other transformation are needed (see Chapter 3.3).

3 VISUALIZATION

3.1 Simulation Grid - Basic Transformations

As mentioned above, the grids of the simulations we are visualizing are characterized by their dynamic vertical coordinates. Furthermore, the data is stored in the transformed horizontal-regular grid. To guarantee a mathematically and physically exact visualization of the meteorological data, the real-world coordinates in the horizontal and vertical directions must be derived from the simulation output.

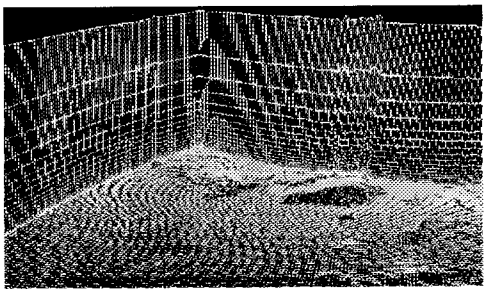

Figure 1: Simulation Grid

In the vertical direction all described models have 20 layers. The coordinates of the layers can either represent height values (z-system) or pressure values (p-system). Both states depend on the pressure values on the ground (PS) and on model-specific vertical coordinate parameters (AK, BK). In meteorological practice the visualization of simulated data is usually requested either at the horizontal grid slices or between two horizontal slices. To calculate the pressure for each horizontal slice, the following equation must be solved [Eddma93]:

$$P(i, j, k) = AK(k) + BK(k) \times PS(i, j) \quad (1)$$

$$P(i, j, k) = \frac{AK(k) + AK(k+1)}{2} + \frac{BK(k) + BK(k+1)}{2} \times PS(i, j) \quad (2)$$

The values for the z-system depend on the differences of the temperature and the pressure at every layer:

$$Z(i, j, k+1) = Z(i, j, k) + 184000 \times \left(1 + \left(\frac{1}{273.5} \times \left(\frac{T_2 + T_1}{2}\right)\right) \times \log\left(\frac{P_1}{P_2}\right)\right) \quad (3)$$

The calculation starts at the lowest slice level with terrain as the basic height. Thus, the height values (in meters) for the vertical grid points are determined.

The data from the European model are internally stored in the so called "rotated latitude/longitude grid" [EM91]. This is a horizontal-regular grid where the simulations can be calculated 35% faster than in the irregular grid. To derive the rotated latitude/longitude grid "the north pole is transformed into the pacific". See Fig. 2.

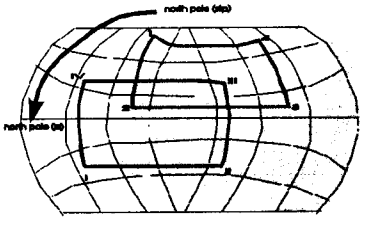

Figure 2: Rotated latitude/longitude grid

To elucidate the difference between geographical coordinates and the rotated latitude/longitude grid the bounding vertices of the two grids are listed in the following table:

	geographical	rotated-spheric	
1	62° N, 36° W	32° S, 57° W	I
2	6° N, 36° W	32° S, 33° E	II
3	6° N, 108° W	32° N, 57° E	III
4	62° N, 108° W	32° N, 57° W	IV

In order to visualize the data, every grid point must be transformed back from the rotated system into the geographical system using equations 4 and 5. Here, the index *rs* denotes the coordinates of the north pole in the rotated latitude/longitude grid and *n* the north pole in the geographical system:

$$\lambda = \operatorname{atan}\left(\frac{\sin\lambda_n \times (\cos\varphi_n \sin\varphi_{rs} - \sin\varphi_n \cos\lambda_{rs} \cos\varphi_{rs}) - \cos\lambda_n \sin\lambda_{rs} \cos\varphi_{rs}}{\cos\lambda_n \times (\cos\varphi_n \sin\varphi_{rs} - \sin\varphi_n \cos\lambda_{rs} \cos\varphi_{rs}) + \sin\lambda_n \sin\lambda_{rs} \cos\varphi_{rs}}\right) \quad (4)$$

$$\varphi = \operatorname{asin}(\cos\varphi_n \cos\lambda_{rs} \cos\varphi_{rs} + \sin\varphi_n \sin\varphi_{rs}) \quad (5)$$

Usually, the meteorological model grids are displayed not in the geographical system but in a stereographic projection. This can be performed with the following calculation, where λ and φ are a point's coordinates while pol_x denotes the north pole's geographical coordinates. [Prome76 , Aftah94]:

$$P_x = pol_x - \tan\frac{90° - \varphi}{2} \times \sin(10° - \lambda) \times c \quad (6)$$

The y-coordinate is calculated similary. Here, *c* is a constant term which is required for the projection [Aftah94]. With the described calculations for the vertical and horizontal coordinates a mathematically and meteorologically exact visualization can be performed.

3.2 Visualization Techniques

Several visualization techniques have been implemented for an efficient analysis of the meteorological data. After discussion with our partners at the German Weather Service, we selected the visualization techniques and implemented them with respect to the users' needs. Each of our algorithms generates polygonal and/or textual graphical primitives which are passed over to our Vis-a-Vis rendering system [Frueh92].

Contour lines

Contour lines are used for two different purposes – as context information and as isolines. Context geometry, such as grid lines, country borders, coast lines, and rivers are essential for the user's orientation in a meteorological visualization. While grid lines or grid layers are generated from the simulation results (see Chapter 3.1) geographical context information is read from files. Isolines in scalar simulation results can be computed on grid layers as well as on arbitrarily oriented slices (see below). The lines are generated as linear splines with a variable resolution and can be color-coded.

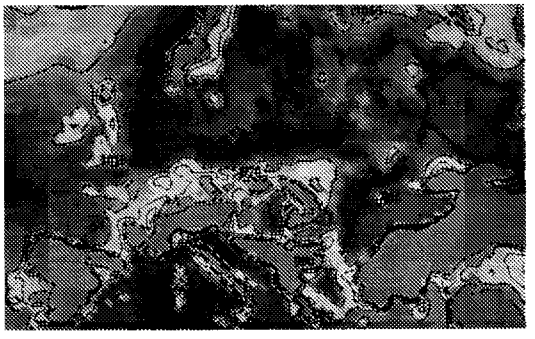

Figure 3: Isolines and pressure

Text

Text strings are used for several purposes. The data level of each isoline can be represented either by an accompomying text string or through use of color-coding. Text is also applied to display the real-world altitude (or the pressure level) of a horizontal slice. Furthermore, textual display of local simulation data is utilized in interactive data probing (see Chapter 4).

Slicing

Slicing in the ijk-layers of the curvilinear simulation grid is performed quickly without computational effort. Additionally, horizontal and vertical slices can be oriented freely in the simulation grid. Arbitrary positioning of slices is not allowed in our system because such visualizations are of little use to forecasters. Slices can be color-coded with respect to a mapping transfer function which is specified by the user. Between the sampling points on the slices Gouraud color interpolation is performed by the graphics hardware. As an alternative, isolines of scalar simulation results (and derived scalars) can be displayed on the slicing planes. During slicing, a real-time update of the visualization is achieved through a variable resolution of sampling points in the slices.

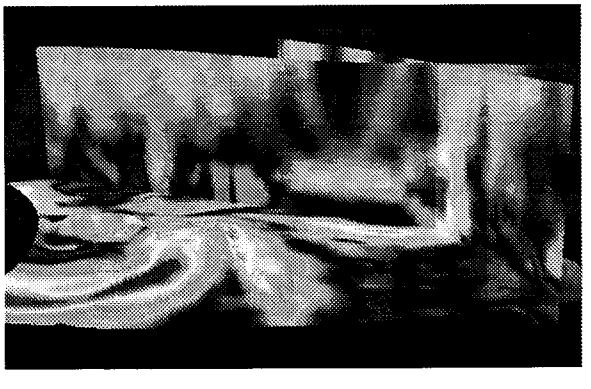

Figure 4: Arbitrary slices

Isosurfaces

Isosurfaces are closed polygonal surfaces representing a certain datalevel in a continuous data volume. In our sys-

tem isosurfaces within the discrete simulation data are approximated using the Marching Cubes algorithm [Loren87]. The actual computation is performed in the rotated latitude/longitude grid. Surface triangles are then mapped to the geographical coordinate system for rendering. Graphical smoothing is available through gouraud shading. Pseudo-color can be used to express the current isovalue-level of the surfaces.

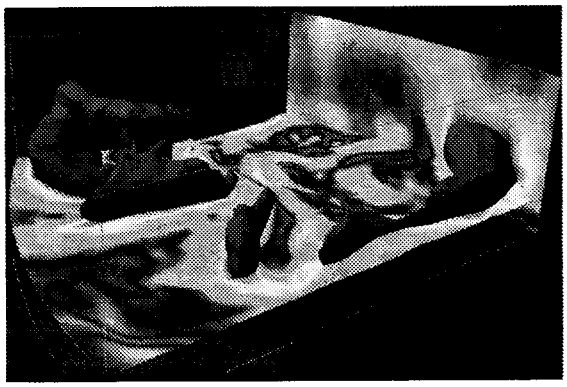

Figure 5: Isosurfaces of wind velocity

Icons and symbols

Vector icons, i.e., arrows can be displayed on grid slices. The arrows may be color-coded and their density can be chosen by the user to prevent visual clutter (see Fig. 6). Several special icons and symbols which are used in the meteorological community (see Fig. 7) are also available – especially for the visualization of observation data.

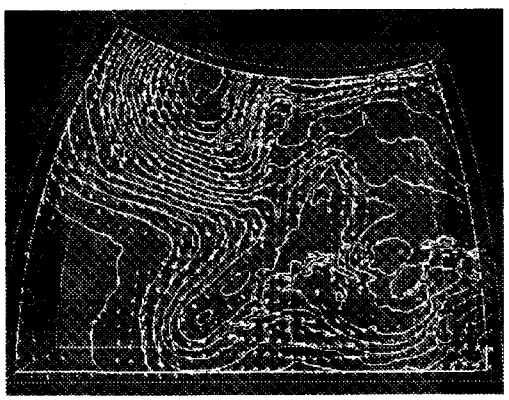

Figure 6: Wind arrows

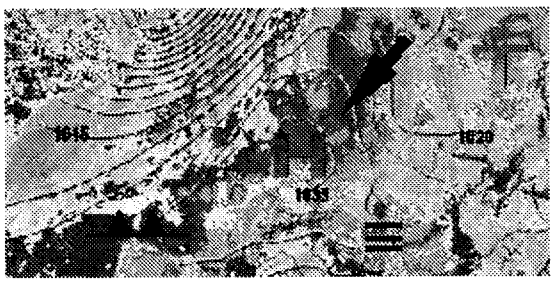

Figure 7: Meteorological symbols

3.3 Integration of Observation Data

The integration of simulated data with measured data is one of the challenging research issues in scientific visualization [Hesse94, Pagen95a, Pagen95b]. Comparative visualization is used to investigate the accuracy of the simulation model and to enhance the simulation algorithms. In meteorological observation, data is gathered hourly for many locations all around the world and is stored in the MAP database as described in Chapter 2. However, in most cases simulation time and observation time are not identical. Furthermore, the geographical distribution of the observation stations is not on a grid but scattered.

Currently in our system, observation data can be visualized only using icons, symbols, and text (see Fig. 8). Thus, a comparison of simulated and observed data is only possible at the locations of the observation stations. If simulation time and observation time are not identical, one or the other can be interpolated. Even in this simple case we have to be aware of possible errors induced by the visualization, e.g., through color interpolation if pseudo-color is used. If we want to compare the data at the numerical level, we transform the geographical coordinates of the observation location to the rotated latitude/longitude coordinate system. In this coordinate system we derive the local simulated data through bi-linear interpolation from the data at the surrounding grid nodes.

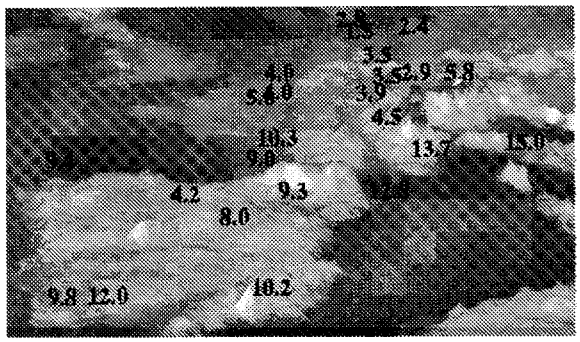

Figure 8: Integrated measured data at selected locations

In the future we intend to transform the scattered observation data with a triangulation algorithm to an unstructured triangle-grid. On this basis a suiting interpolation algorithms will be choosen. Then, we will be able to compare simulated and observed meteorological data also using isolines at the bottom layer of the computational grid. A careful analysis of the interpolation-induced error will be carried out before we release such an analysis tool to meteorologists.

4 NAVIGATION AND INTERACTION

Real-time visualization with a high degree of interaction is the key to an efficient analysis of complex simulation data [Brodl92, McCor87]. On one side, powerful graphics hardware and fast visualization algorithms are required. On the other side, an efficient implementation for user navigation and interaction in 3D space is essential.

Navigation

We implemented a software solution for user navigation in 3D data space with the standard mouse. Since our users' aim is not a fly-through such as in VR applications but an interactive analysis of 3D simulation data we realized a scene-in-the-hand navigation as it is used in desktop engineering applications such as CAD/CAE systems.

Rotation of the scene is realized using a virtual trackball. Zooming and panning are also possible via direct interaction in the 3D graphics window. Application-specific standard viewpoints as well as viewpoint reset are provided through buttons in the 2D GUI .

Interaction with data

Interaction for conFiguration and control of the visualization system is provided through the Motif-based GUI (see Chapter 7). Rotation of the dataset is realized via a virtual trackball. Zooming and panning are also implemented through direct mouse-input in the 3D graphics window. User interaction with the data itself is a powerful approach for a faster analysis of complex simulation data [Spera90]. Slicing and cropping are implemented in our system – as requested by our partners at the DWD – through input in the Motif-GUI. However, we are currently implementing a mechanism for steering probes in 3D data space using direct input in the 3D graphics window.

In principle, point probing can be realized with a 3D manipulator as well. 2D point probing is not as flexible as 3D point probing but more user-friendly. Our current implementation of 2D point probing is restricted to that data which is currently visualized with pseudo-colors on slices. When the mouse is clicked, one frame is rendered purely with ambient light into the back-buffer. Then the pixel color is read, the mapping transfer function of the pseudo-color visualization is inverted, and the local data value is extracted and displayed in numerical form. A more accurate and more flexible point probing algorithm on basis of the calculation of ray/object intersection points will be implemented in the future.

5 TIME-CRITICAL VISUALIZATION

In meteorology dynamics are especially important. In particular, most weather situations can only be analyzed correctly if their dynamics are considered. In most currently available visualization systems the animation features appear to be an add-on. They are limited to playing image sequences or visualizing several datasets consecutively over time, instead of providing precise mechanisms for time control. To visualize the dynamics in meteorology this animation concept is insufficient. A scientific animation is needed. Some research has already been carried out in this field, e.g., by [Polth94, Wijk94, Bryso96].

The implemented animation modul of RASSIN [Aftah95, Lux95] gives the user complete control over how the time-dependend data is visualized. During the scientific animation, the user can continue to fully interact with the system. By decoupling the image display rate from the image generation times, an especially precise analysis of the dynamics is realized. This decoupling is possible because the rendering times are estimated.

The internal data management concept and visualization techniques treat time equally as the fourth dimension of the data. The handling of time must happen concurrently and analogously to the geometric transformations in the three-dimensional space [Asthe91]. Each object has a 4th coordinate which describes its time of existence. The handling of this 4th coordinate allows translation, scaling, shearing, projection and rotation in time. These are the so called time-transformations [Lux95]. The selected viewport in the scene also has a fourth dimension, which is the time period to be mapped on a viewing time. Thus zooming and clipping in the time axis can be performed.

For allowing time control in scientific animation the difference between the real physical time (t_{real}) and the data time (t_{data}) must be distinguished. This is achieved with the definition of the animation-scale-factor (s_a), which describes the relation from real time interval to data time interval [Wijk94].

$$s_a = \frac{\Delta t_{data}}{\Delta t_{real}} \tag{7}$$

Rendering geometrical scenes into frames on the screen always consumes a certain minimum amount of computation time. Frames cannot be shown faster than they are produced. Furthermore, constant rendering times are not achievable due to changing scene complexities or varying processor loads. This requires an a-priori rendering time estimate.

Usually, most scientific animation functions simply trigger the rendering of scenes regularly and get the image display rate depend on the homogeneity of rendering times. To achieve a constant image display rate, the separation of rendering and displaying is necessary. Images must be rendered in advance, long enough before they are to be displayed. By adjusting the time between rendering-completion and image-display, the lack of homogeneity of rendering times can be absorbed. But when there is a long delay between rendering and displaying of frame (i), the system cannot allow interactions by the user. Interactions are only possible between the display of frame (i) and the rendering of the next frame (i+1) (see Fig. 9). So an estimate of the rendering time is nessesary. Furthermore, a decision on how much of the remaining time will be used for interaction and how much for waiting (absorption safety) is needed. The estimation is performed by causing the animation module to look back on the last measured times. The estimation is controlled after preparing and rendering of a frame. The decision of how to distribute the remaining time, is left to the user. He must choose a position between maximum interaction time and maximum display accuracy safety.

The implementer-defined time between two frames (t_{frame}), can be divided into a) the time for allowed interaction for the next frame ($t_{interact}$), b) the time for preparing the next frame within the visualization system ($t_{prepare}$), c) the time for rendering (t_{rend}), and d) the time for waiting until the frame will be displayed (t_{bw}). Figure 9 shows these timings:

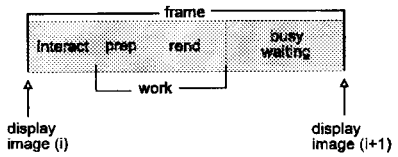

Figure 9: Time between two frames

For the creation of frame (i), the following procedure must be performed: first, the time for preparing and rendering is estimated (this is the minimum amount of computation time). If the total estimated time (t_{work}) is greater than t_{frame}, a number of frames must be skipped. The number

$$\Delta datastep = \left\lceil \frac{t_{work}}{t_{frame}} \right\rceil \quad (8)$$

is calculated and the frame (i + Δdatasteps) will be the next one created. Of the remaining time (t_{frame} * datastep - t_{work}) an implementer-defined percentage is allowed for user interactions. Next, after allowing interactions for a time of $t_{interact}$, the preparation of frame (i) is performed by the visualization system and the time actually consumed is measured. If the time now left until the demanded display of frame (i) is shorter than the estimated rendering time (t_{Erend}):

$$t_{interact} + t_{prep} + t_{Erend} > t_{frame} \quad (9)$$

the creation of frame (i) is immediately aborted and the creation of frame(i+j) (where j is calculated by a new estimate) is started. The user interactions performed up to now are stored in the event-queue (i.e., the interaction concept of Motif), so that they are considered in the creation of the next frame.

In the other case (enough time left), the rendering is performed by the visualization system. After the rendering, the time actually consumed is checked. If it is too early to display the next image:

$$t_{interact} + t_{prep} + t_{rend} \leq t_{frame} \quad (10)$$

the animation will wait until the exact moment (t_{bw}). Therefore, the timings are like in Figure 9. Otherwise, if after the rendering the time t_{frame} is overstepped:

$$t_{interact} + t_{prep} + t_{rend} > t_{frame} \quad (11)$$

the image will be shown immediately and the implementer will be informed about this tardiness. Then, the estimation for the next frame can begin.

With the determination of the percentage of interaction time the implementer has the choice of deciding between the display accurary and amount of interaction. This process of a-priori time estimation is based upon the assumption that constant values of t_{frame} correspond to a constant time between the single datasets. With slight modification, this works for data scattered along the time axis as well. Here, only the calculation of datasteps must be adapted to variable t_{frame} values [Lux95].

If time steps are missing in the data or if they are scattered along the time axis, interpolation in time is performed to produce a virtually regular dataset in time. This is also useful when a higher resolution in time is desired than provided by the data in order to get smoother animations. The interpolation of data in time may be linear or calculated by using a higher order function. The calculation can either be during the animation, which results in increasing $t_{prepare}$ values, or prior to the animation, resulting in an increased memory consumption of the visualization system.

6 SYSTEM ARCHITECTURE

Our visualization system has a modular software architecture. It is divided into sub-modules so that the system is flexible and expandable.

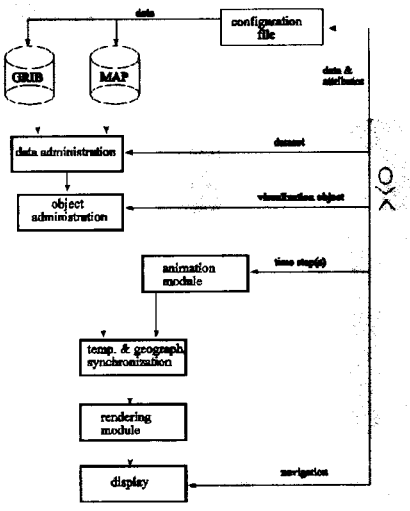

Figure 10: System architecture

The system works with a conFiguration file where meteorological data and their attributes are defined. Thus, the system appears from the start as a meteorological-turnkey system. No further configuration by the user is required and the exploration of the data can commence immediately.

The specified data can be imported from the GRIB- or MAP-database, i.e. simulated or measured data (see Chapter 2). The loaded data is processed and stored in the administration module. The internal data management concept contains two general list [Aftah94]. In the first data list (controlled by the data module) the values for each meteorological variable are stored with a pointer to the referenced model coordinates. Each variable contains a pointer to the object list in which the rendered visualization objects are stored (controlled by the object module).

The user can specify any desired visualization technique and visualization objects. With the integrated animation module the user can define a single time stamp or a scientific animation to be visualized. The chosen visualization object and the time will be filtered in the geographical and temporary synchronization. Thus, only the objects that are convenient to the geographic- and time-contexts pass the filter and are admitted to the rendering

module. After rendering the objects, they are stored and managed by the object administration module as described above. In the 3D scene the user can navigate and interact with the visualized objects.

During run-time, new visualization techniques and objects can be chosen and different time steps or a scientific animation can be performed. Furthermore, any number of datasets with additional meteorological variables or new time steps can be imported on demand. Rendered frames can be sent to a printer, e.g., for presentations. Single pictures or animated sequences can be stored as yuv-files or sent to a digital video recorder.

7 HCI CONCEPT

RASSIN offers a thoroughly designed human computer interaction concept which is tailored to users from meteorological sciences. The concept is based on the five principles for ergonomic dialogue design [DIN88]. Therefore, the typical meteorological work flow must be investigated [VDI90]. This work can be divided into two parts: in the daily operation the meteorologist produces the weather forecast. The output of the numerical simulations must be quickly evaluated. In his/her scientific work the correctness of new forecast models must be proved [Kaise95]. With this background the following five principles were realized:

Adequate to the task: The user interface has two different levels. The first level contains all interaction elements needed to generate the weather forecast. In the second level additional elements and tools are available for an analysis of the data in more detail.

Self description: The names of the elements and the labels of the buttons are self-explanatory or adapted to the meteorological technical language. The user information shows the actual geographical and time context of the visualized data.

Controllable: Configuration files allow definable default settings of all parameters which are relevant to the visualization at system start up. During run-time, any number of datasets can be imported on demand. The elements are inserted and deleted dynamically through the user interface. Thus, only the elements which are currently necessary are shown. A high level of interaction is also achieved through the use of keyboard shortcuts. Furthermore, application-specific standard viewpoints as well as a viewpoint reset are provided through buttons.

Expectation conformity: The procedure to visualize all objects is similar, independent of the dataset and the object itself. Thus the users-computer dialogue is consistant throughout.

Error robustness: Any incorrect user input is intercepted with constructive error messages.

In addition to these five principles, a general color concept for the GUI has been implemented. This concept supports work using simultaneous, multiple meteorological variables by linking color coded visualization objects with variable names and user-interface elements (see Fig. 11).

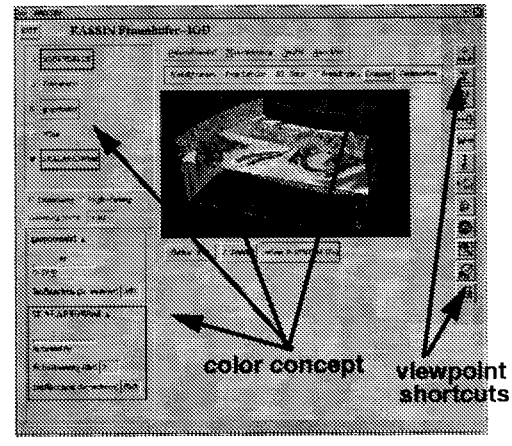

Figure 11: The user interface of RASSIN

8 CONCLUSIONS AND FUTURE WORK

RASSIN is a visualization system for the operational meteorological use. It was developed at Fraunhofer-IGD for the German National Weather Service DWD. The software serves for actual forecasting as well as for visualization in the context of simulation-code development.
RASSIN allows an interactive investigation of numerical, measured, and/or observed meteorological data which are read from different databases. The time-dependent data may be 2D, like (ground-)pressure, or 3D, like temperature or wind. The visualization of simulation data is performed using the original irregular grid, which is – because of the correlation of actual pressure and temperature values – dynamically. The geographical and temporarily context of the meteorological data is considered at every visualization step.

To analyze and investigate the dynamics of the data a dedicated concept for time-critical visualization was implemented. The human-computer-interaction has been designed for an intuitive use of the system by meteorologists.

In future, on one side, we will extend the currently implemented visualization features by traditionally-meteorological visualization techniques. On the other side, advanced graphics features, like hardware-textures, will be exploited to further improve the performance of the system.

ACKNOWLEDGMENTS

We have benefit from a long and fruitful relationship with our project partners at DWD under the direction of H.-J. Koppert.

We thank all colleagues and students who have contributed or who are currently contributing to this project: U. Kaiser, R. Schall, F. Schröder. Special thanks go to Hermin Aftahi who designed and implemented the first version of the RASSIN software and who named the software after her daughter.

We also thank Mike Sokolewicz for the tedious task of proof reading the paper.

REFERENCES

[Accu96] Accu Weather, Inc.; Werbe-Broschuere: Paris 1996

[Aftah94] Aftahi, H.: "Visualisierung meteorologischer Daten"; Diplomarbeit (FH), Fraunhofer-IGD Darmstadt 1994

[Aftah95] Aftahi, H.; Lux, M.; Schröder, F.: "Gaining Insight into Dynamics - Extensive Control of Time in Interactive Scientific Visualization"; Fraunhofer-IGD, 1995

[Asthe91] Astheimer, P.; Felger, W.; Göbel, M.: "Time in Scientific Visualization"; Fraunhofer-IGD, 1991

[Brodl92] Brodlie, K; Carpenter, L.; Earnshaw, R.; Gallop, J.; Hubbold, R.; Mumford, A.; Osland, C.; Quarendon, P.: *Scientific Visualization: Techniques and Applications.* Springer Verlag, Berlin, 1992.

[Bryso96] Bryson, S.; Johan, S.: "Time Management, Simultaneity and Time-Critical Computation in Interactive Unsteady Visualization Environements"; IEEE Computer Graphics & Applications, 1996

[Damra92]Damrath, U.; Majewski,D.; Steppler,J: "Atmosphäre im Computer - Möglichkeiten und Grenzen der numerischen Wettervorhersage"; Zeitschrift c't, Heft 12, Dezember 1992

[DIN88] DIN-Norm 66234 (Bildschirmarbeitsplätze) Teil 8: "Grundsätze ergonomischer Dialoggestaltung"; Beuth Verlag, 1988

[Doms95] Doms, G.; Schättler, U.: "Bereitstellung der Anfangs- und Randdaten für das LM", 2. Arbeitspapier zur Lm-Entwicklung; Deutscher Wetterdienst, 1995

[EM91] "Daten des operationellen EUROPA-Modells (EM) auf der Cray Y-MP und der MFB", Deutscher Wetterdienst, 1991

[Earth96] Earth Watch Communications Inc.; "The world is Anything But Flat"; Werbe-Broschuere, Paris 1996

[Eddma93]Eddmann, W.; Majewski, D.; "Die Datenbank des Europa-Modells des DWD"; Deutscher Wetterdienst, 1993

[Encar93] Encarnação, J. et al.: "Advanced research and development topics in animation and scientific visualization"; in "Animation and Scientific Visualization", Academic Press Ltd., 1993

[Foley90] Foley, J.; van Dam, A.; Feiner, S.; Hughes, J: "Computer Graphics - Principles and Practice"; Adison Wesley, 1990

[Frueh92] Frühauf, M.; Haas, S.: "Die Vis-a-Vis Renderingschnittstelle Version 2.3";Fraunhofer-IGD, Darmstadt 1992

[Hesse94] Hesselink, L.; Post, F.; van Wijk, J.: "Research Issues in Vector and Tensor Field Visualization"; in Rosenblum, L. et. al. (eds.): *Scientific Visualization: Advances and Challenges.* Academic Press Ltd., pp. 488-495, 1994.

[Hibba89] Hibbard, W.; Santek, D.; "Visualizing Large Data Sets in the Earth Sciences"; IEEE Computer Graphics & Applications, 1989

[Kaise95]Kaiser, U.: "Optimierung der Mensch-Maschine-Kommunikation bei interaktiven Visualisierungssystemen in meteorologischer Anwendungen"; Diplomarbeit (FH Darmstadt), Fraunhofer-IGD, Darmstadt, 1995

[Kavou96]Kavouras Inc.; "on the Front-The newsletter for Kavouras Customers and Weather Aficionados"; Vol.6 1996

[Loren87] Lorenson, W.; H. Cline: "Marching Cubes: A High Resolution 3D Surface Construction Algorithm", *ACM Computer Graphics*, 21(4):163-169, 1987.

[Lux95] Lux, M.: "Bedeutung der Zeit in der wissenschaftlich-technischen Visualisierung und ihre Berücksichtigung in meteorologischen Anwendungen"; Diplomarbeit, Fraunhofer-IGD, Darmstadt, 1995

[McCor87]McCormick, B.H.; et al.: "Visualization in Scientific Computing"; ACM Computer Graphics (Proceedings of SIGGRAPH '87), 21(6); 1987

[Pagen95a] Pagendarm, G.; Post, F.: "Comparative Visualization - Approaches and Examples"; in Göbel, M.; Müller, H.; Urban, B. (eds.) Visualization in Scientific Computing. Springer Verlag, Wien, pp. 95-108, 1995

[Pagen95b] Pagendarm, G.; Walter, B.: "Competent, Compact, Comparative Visualization of a Vortical Flow Field", IEEE Transactions on Visualization and Computer Graphics, 1(2):142-159, 1995

[Pogod97] Pogoda, M.; "Die Programmierschnittstelle der MAP-Datenspeicherung (MAPDB 97.1-API)" Deutscher Wetterdienst, 1997

[Polth94] Polthier, Konrad; Rumpf, Martin; "A Concept for Time-Dependent Processes"; 1994

[Prome76] Promet - meteorologische Fortbildung - (Hrsg. DWD); "Das Barokline Modell", 1976

[Prome81] Promet - meteorologische Fortbildung - (Hrsg. DWD); "Meso-scale-Modell", 1981

[Prome84] Promet - meteorologische Fortbildung - (Hrsg. DWD); "Das Europäische Zentrum für Mittelfristige Wettervorhersage (EZMMW)", 1984

[Schro93] Schröder, F.; "Visualizing Meteorological Data for a Lay Audience"; IEEE Computer Graphics and Applications, 1993

[Spera90] Speray, D.; Kennon, S.: "Volume Probes - Interactive Data Exploration on Arbitrary Grids". Computer Graphics, 24(5):5-12, 1990.

[VDI90] VDI-Richtlinie 5005: "Software-Ergonomie in der Bürokommunikation", Verein deutscher Ingenieure, Beuth Verlag, 1990

[Weiha95] Weihai,C.; Zeshang,T.; "MeteoVis-Visualizing Multi-variables Interactively in 3D Meteorological Data Sets"; Tsinghua University, Peking, 1995

[Wijk94] Wijk, J. van; "Time control in interactiv scientific animation"; In Proceedings: Fifth Eurographics Workshop on Visualization in Scientific Computing, Rostock, 1994

[Zwirn95] Zwirn, J.; "Everything Goes-The Goes Program Newsletter"; Space Systems/Loral, Vol.4, April 1995