INTERACTIVE FOCUS AND CONTEXT DISPLAY OF LARGE RASTER IMAGES

Uwe Rauschenbach[1]
Tino Weinkauf
Heidrun Schumann

University of Rostock,
Computer Science Department
D-18051 Rostock
GERMANY

{urausche,weinkauf,schumann}@informatik.uni-rostock.de
http://wwwicg.informatik.uni-rostock.de/

ABSTRACT

This paper presents the RECTANGULAR FISHEYE VIEW, an interactive focus–and–context presentation technique for large raster images on mobile computers with small displays and limited processing power. Both the viewing of locally available images and the demand–driven display and transmission of remotely–stored images are supported by the technique. The underlying geometry calculations are explained, and the design decisions for supporting rapid interactive feedback are discussed. A scenario is described which demonstrates the performance of the method.

Keywords: Fish Eye View, Raster Images, Interactive Display, Focus–and–Context Techniques, Mobile Computers

1 INTRODUCTION

Graphical presentation of large amounts of information on computer displays has to cope with the problem of limited screen space. The viewer of such a presentation has two conflicting goals in mind: He requests a *high degree of detail* but also wants to have an *overview* over the whole presentation for orientation purposes. Visualization researchers proposed *focus–and–context techniques* to solve this conflict for the display of large layouts on desktop workstations.

Using mobile computers, both processing power and display space are more limited than in a workstation scenario. Additionally, mobile devices often fetch graphical data as *raster images* from a remote server using a low–bandwidth data link. A presentation technique for those mobile settings must be able to display a large raster image in a screen–space–saving manner, trading off the two requirements "detail" and "overview" against each other. Furthermore, its computing requirements should be moderate in order to work on the targeted hardware and to allow rapid interactive control. Last but not least, the scheme should work with a demand–driven transmission method like the one presented in [Rausc99], transmitting only those image data which are needed for display.

After reviewing related work, we will propose the RECTANGULAR FISHEYE VIEW as a new technique meeting these requirements. In contrast to [Rausc99], where we discussed the transmission implications of our technique, this paper focuses on the necessary geometry calculations and the interactivity of the technique.

[1] This work was supported by The German Science Foundation under contract no. Schu-887/3-3.

2 RELATED WORK

In the research field of visualization, *focus–and–context techniques* have been proposed to solve the problem of displaying very large layouts on desktop workstations equipped with powerful processors. These techniques combine a *focus display*, which shows a part of the layout at a high degree of detail, and a *context display*, which presents the whole picture in lower detail to provide the overview. The position of the focus is determined by the current *point of interest* of the user.

There are several possibilities for focus–and–context displays: focus and context can be displayed separate over time (this corresponds to *zooming*) or in two separate presentations side by side. This approach avoids introducing distortion, but the user is required to make the link between focus and context mentally. Another approach is the *fish eye view*, where focus and context are integrated in one presentation. Depending on a measure of *distance* from the focus, the context is distorted such that its space requirements decrease. This approach offers a detailed view near the point of interest and maintains large–scale features of the whole layout at the cost of introducing distortion far away from the point of interest.

Let us now discuss some of the many works in the field of fish eye views related to our technique. An excellent, comprehensive bibliography about non–linear magnification techniques and fish eye views can be found on the Nonlinear Magnification Home Page [Keahe98].

Fish eye techniques have first been used in computer science by Furnas [Furna82] for the presentation of structured data, achieving context reduction by hiding nodes in the structure based on the distance from the focus and an interest measure. Other works (e.g., [Keahe96]) propose methods for non–linear magnification, where the scaling factor (magnification or minification) is defined as a two–dimensional function of the location in the picture. These techniques may use a different scaling factor for every pixel, which introduces smooth, concentric distortions and requires a high computing effort.

Since the focus shows only a part of the whole picture in high detail, the user often wishes to move it to another position to reveal detail there as his point of interest changes. This *interactive exploration* requires recomputation, which needs a powerful processor for fast response using the technique described above. Sarkar et al. [Sarka93] proposed for graphical layouts a method called “rubber sheets” which divides the image into stripes and stretches all pixels of a stripe by the same factor. Thus, computing requirements are decreased.

3 THE RECTANGULAR FISHEYE VIEW

3.1 Requirements

The basic requirements to a focus–and–context display technique for mobile environments have already been stated in section 1. Additionally, flexible tailorability of focus region and context areas is highly desirable. Especially, the user should be supported in exactly specifying the area and the degree of detail of the focus. Since not all parts of the presentation offer the same degree of detail, interaction techniques with short response time have to be provided for revealing hidden details by re–positioning the focus. In order not to destroy basic properties of the image (like angles, proportionality or parallelity), no distortion is acceptable in the focus area. In the context area, distortion is the foundation for saving display space and can not be avoided. Interactive control over the magnitude of this distortion must be provided to support the user in adapting the tradeoff between distortion and space requirements to his current task.

3.2 Description of the technique

We will now propose the RECTANGULAR FISHEYE VIEW as a new focus–and–context technique for the interactive display of large raster images in mobile environments. This technique has been designed to meet the requirements stated above.

To keep the computing effort low and allow the integration with a region–of–interest–based image transmission technique, the image is divided into non–overlapping rectangular parts in each of which the distortion is chosen the same for all pixels. As distortion parameters, we are using downscaling factors which are powers of two in order to speed up computations and to allow the integration with the transmission method.

The figures 1 and 7 illustrate the idea. A focus region R_0 in the center of the image is not downscaled at all. It is surrounded by context belts, which display the remaining image parts in a downscaled version to save screen space. Each belt C_j $(j > 0)$ is defined as a pixel set difference based on the context rectangles R_j (see figure 1) as follows:

$$C_j = R_j \setminus R_{j-1}. \tag{1}$$

The j–th belt is assigned a context downscaling factor $s_j := 2^j$ which controls the distortion of the belt and the amount of screen space saved. Since the degree of interest of the user in image details decreases with increasing distance from the focus, the downscaling factor of a belt is the higher the further that belt is away from the focus. In order to attach each context belt to its neighbour without discontinuities, the belt C_j is split into $8 \cdot j$ partial rectangles. Each of these partial rectangles is assigned one scaling factor for the X– and another one for the Y–direction. Some of the factors are pre–determined by the context belts inside the belt C_j, and the remaining ones are set to

2^j. The grid of partial rectangles is called the downscaling grid. Figure 7 shows such a grid with two context belts, denoting the pair of downscaling factors for each grid rectangle as downscaling pair (s_X, s_Y). Figures 3 to 6 show the result of performing the downscaling.

Tailorability of the focus does not only mean to specify its size and position but also to define the degree of detail needed. This can be done by zooming, which assigns a zoom factor z to the focus. To fit into our scheme, we chose to support zoom factors which are powers of two. In order to maintain the assumption that the focus offers the highest degree of detail, we multiply all downscaling factor by the zoom factor[2].

The context belts can be displayed in different ratios to each other (e.g., the first belt with $s_1 = 2$ may be chosen twice as wide as the second one with $s_2 = 4$). These ratio parameters allow the user to tailor the context belts with respect to distortion and space requirements to his needs. An automatic choice of the ratio parameters using linear optimisation techniques can be used to force the size of the RECTANGULAR FISHEYE VIEW to fit into the available display or window size. Further flexibility can be gained by the opportunity to select which context belts should be visible.

4 GEOMETRY CALCULATIONS

After having described the basic structure of the RECTANGULAR FISHEYE VIEW, this section will explain how to compute the *corner coordinates of the context rectangles*. The computation is based on the *size of the image*, the *position and size of the focus region* and the *context belt ratio and visibility*. Modifying these parameters through direct manipulation (see section 5) supports the tailorability of the view and requires rapid recomputation of the coordinate mapping to update the downscaling grid.

These input parameters are needed:

$$\begin{aligned}
PicWid &\Rightarrow \text{width of original image}\\
PicHgt &\Rightarrow \text{height of original image}\\
nBelt &\Rightarrow \text{number of belts plus focus}\\
Scl_i &\Rightarrow \text{downscaling parameter of belt } i\\
Ratio_i &\Rightarrow \text{ratio parameter of belt } i\\
& \quad\;\; i = 0, \ldots, nBelt - 1\\
Pos_{left,0} &\\
Pos_{top,0} &\\
Pos_{right,0} &\\
Pos_{bot,0} &\Rightarrow \text{coordinates of the focus}
\end{aligned}$$

[2]The zoom factor is interpreted as downscaling factor, too – the higher this factor, the smaller the displayed image.

The array Scl_i contains for the focus the zoom factor z and for each visible context belt the product of its context downscaling s_j and z. Analogously, $Ratio_i$ contains an undefined value for the focus and the ratio parameter for each visible belt.

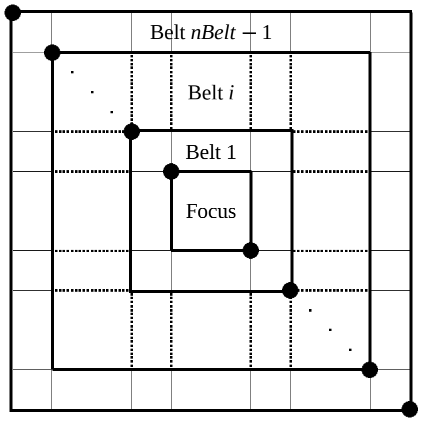

Figure 1: Geometric structure.

To describe the geometric structure of the RECTANGULAR FISHEYE VIEW, only the coordinates $Pos_{r,i}$ of the belt rectangles are needed (cf. figure 1). For reasons of consistency, the belt rectangle with the number zero is formed by the focus, and the belt rectangles with a number greater than zero represent the context rectangles. Thus, the computation has to calculate the following coordinates:

$$\begin{aligned}
Pos_{r,i} &\Rightarrow \text{coordinate of belt } i \text{ in direction r}\\
& \quad\;\; \text{in the original image}\\
& \quad\;\; \text{where } r \in \{left, top, right, bot\}
\end{aligned}$$

The coordinates $Pos_{r,nBelt-1}$ are defined implicitly by the point $(0,0)$ and the size of the original image $(PicWid, PicHgt)$ and correspond to its corner points. The rectangle vertex coordinates of the belts $i = 1, \ldots, nBelt - 2$ remain to be computed. To do that, the space between the focus and the image boundary has to be distributed between the belts $i = 1, \ldots, nBelt - 1$ according to the ratio parameters specified in $Ratio_i$. The width of a belt i in one direction r is computed as follows:

$$\begin{aligned}
Wid_{r,i} &= \left| Pos_{r,nBelt-1} - Pos_{r,0} \right| \\
&\quad \cdot \frac{Ratio_i}{\sum_{k=1}^{nBelt-1} Ratio_k}
\end{aligned} \tag{2}$$

The width of belt i in the four directions is now used to compute the coordinates of its belt rectangle based on the (already computed) coordinates of the next–inner belt $i-1$ as follows:

$$Pos_{r,i} = \begin{cases} Pos_{r,i-1} - Wid_{r,i} & r \in \{left, top\} \\ Pos_{r,i-1} + Wid_{r,i} & r \in \{right, bot\} \end{cases} \tag{3}$$

As mentioned earlier, the partial rectangles in the RECTANGULAR FISHEYE VIEW are created from rectangular regions in the original image by downscaling them in both directions by – possibly different – factors. After having computed equation (3), the problem may arise that the division of the width and height of these resulting rectangles does not deliver an integer result. The consequence would be discontinuity problems at the boundary between two belts arising from skipped pixels. In order to overcome this problem, we introduce a boundary condition for equation (2) which varies $Wid_{r,i}$ for all belt rectangles excluding the focus such that a division without remainder by Scl_i is possible:

$$Wid_{r,i} \equiv 0 \pmod{Scl_i} \tag{4a}$$

$$(Dist_{r,i} - Wid_{r,i}) \equiv 0 \pmod{Scl_{i+1}} \tag{4b}$$

$$Dist_{r,i} \geq Wid_{r,i} \geq 0 \tag{4c}$$

where

$$\begin{aligned} i &= 1, \ldots, nBelt - 1 \\ Dist_{r,i} &= |Pos_{r,nBelt-1} - Pos_{r,i-1}| \end{aligned}$$

This boundary condition assumes that width and height of the focus rectangle are multiples of the zoom factor. Furthermore, the remaining space must be dividable without remainder by the downscaling factor Scl_1 of the first belt. If this condition is not met, the image may have to be padded.

To satisfy the boundary condition (4), the results $Wid_{r,i}$ from equation (2) are iteratively incremented by one until (4) is met. It can be proven that this leads always to a partitioning where width and height of the belt rectangles are multiples of the respective downscaling factors Scl_i. This boundary condition may lead to small deviations from the belt size relations specified in $Ratio_i$, but it ensures that no pixel rows or columns are skipped.

5 INTERACTION ISSUES

5.1 Manipulating the parameters

As stated in section 3.1, the user must be offered control over the parameters of the RECTANGULAR FISHEYE VIEW in order to support rapid interactive exploration of the presented content. This is achieved by various techniques of interactive *direct manipulation*. Especially, control of size and position of the focus must be achievable in a fast and intuitive way, such that the user can explore various parts of the presented image in detail driven by his special interest. To do this, the operations Move, Resize and Jump are provided. This has been combined with a zoom–and–pan–approach (Zoom+Pan), which makes it easy to obtain an overview over the whole image. Further parameterisation to control the context belts (ratio parameters, visibility of belts) is possible for fine–tuning.

We will now discuss the direct manipulation techniques.

Move. This function allows to change the focus position in the original image by small steps. Moving the focus does not influence the space requirements of the RECTANGULAR FISHEYE VIEW since the size of the focus region and the context parameters remain constant. Moving will usually be exploited if the point of interest drifts slowly into the neighbourhood of the focus.

Jump. Using this function, the focus can be set instantaneously to a new location possibly far away from the old one by specifying the centre position of the new focus. This function is usually exploited to set the point of interest into the context area in order to explore some features there in greater detail. The total view size is not changed.

Define. If the user requires a focus region with a new position and a different size, the Define function supports this goal. The function takes the upper left and the lower right corner of the new focus as parameters affects the size of the view.

Resize. The demand for controlling the size of the area of interest can be met using this function, which allows the user to adjust the size of the focus region. Changing the focus size influences the space requirements of the whole RECTANGULAR FISHEYE VIEW.

Zoom+Pan. The RECTANGULAR FISHEYE VIEW saves screen space by supporting focus–and–context, but it is nevertheless not always guaranteed that the display is large enough to show the whole image. By specifying a zoom factor, the total size of the RECTANGULAR FISHEYE VIEW can be adjusted in steps of powers of two. If not the whole image is visible, panning offers – compared to scrolling – a fast alternative to reveal invisible parts.

5.2 Optimisations for fast response

The interactivity of the RECTANGULAR FISHEYE VIEW enables the user to explore and to understand the displayed image. Thus, it is important that each action of the user results in a fast response of the system. Only by interactive response times the user can develop a *feeling* for the distorted presentation.

This is especially important for the techniques Move and Resize, which are only efficient for the user if each small position or size change results in a rapidly updated display to reflect the changes. In order to meet this requirement, screen redrawing is done belt–sequentially in an interruptible manner. First, the new focus region is drawn, followed by the first context belt and so on. Interruptibility ensures that the redraw sequence is aborted before completion if during the display update one of the functions Move or Resize is executed again.

For generating the RECTANGULAR FISHEYE VIEW, large parts of the original image have to be scaled which is computationally intensive. That's why it is advantageous to store the image data redundantly at different scaling levels and to reuse the correctly–scaled version during interaction feedback in order to decrease the response time. The generation of the pre–scaled versions can be done in a preprocess when loading the image, or image parts scaled for display in an earlier interaction step can be reused.

When creating redundancy it is necessary to find a tradeoff between the additional memory requirements and the savings in processing power. Redundancy is very high if a copy of the image scaled according to each possible downscaling pair is stored. For instance, storing all resolution combinations in figure 7 results in 206.25% redundancy. It can be shown that the redundancy diverges for a growing number of context belts. However, the redundancy can be reduced to $33.\overline{3}\%$ independently from the number of belts if only image versions scaled using pairs (s_X, s_X) with $s_X = s_Y$ are stored redundantly and the remaining scaled image parts are generated on request by scaling the according pre–stored version. For the latter, there are two opportunities: scaling up a low resolution image, which results in high computing speed, or scaling down a high resolution image, which results in high quality. During *interaction*, speed is more important than quality. That's why we are using upscaling: If, for example, a partial rectangle with a downscaling of $(2,4)$ is requested, it is generated by scaling up the X direction of the corresponding rectangle pre–stored with downscaling $(4,4)$. During periods *without interaction*, we use downscaling to re–compute the view at higher quality (see below).

If a transmitted image is viewed online during transmission (cf. [Rausc99]) using the RECTANGULAR FISHEYE VIEW, only some image parts are available at a certain scaling level. Furthermore, recovering the image from the wavelet coefficient field of the transmission technique is too slow for interactive response times. Thus, an efficient image representation must be exploited to store scaled versions of those image parts which are available.

We realised a *local image cache* which is illustrated in figure 2. To achieve efficient storage, each scaling level of the image is divided into tiles. Only those tiles

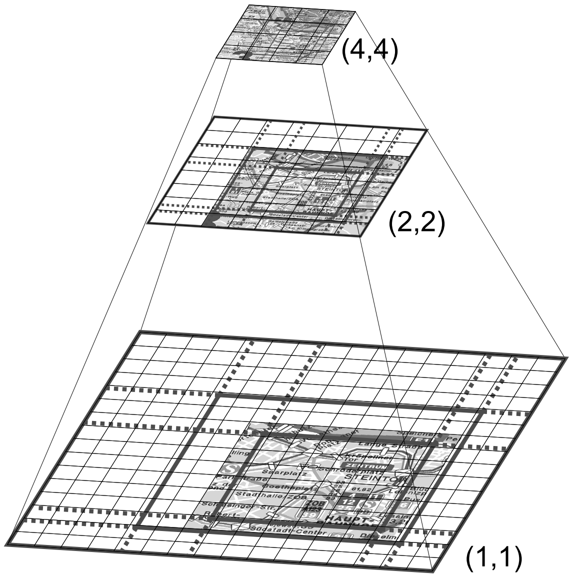

Figure 2: Redundant local image cache.

require storage space which contain image data already transmitted. Since only the focus is available at full resolution, just this image part needs to be stored at level (1,1). Focus and first context belt are available at the next downscaling level and so on. Moving the focus initiates new transmissions which add some more tiles to each level. If a tile which is not available at a certain scaling level is needed for display, data from the corresponding tile at the next lower level are upscaled and used instead. This ensures fast display refresh and efficient storage by exploiting the sparse nature of the transmitted image data.

If the user does not interact with the view, a so called quality view is computed and replaces the rapidly computed, but lower quality representation used during interaction. This quality view is generated by downscaling the full resolution version of the image (if a locally–available image is displayed) or by direct recovery from the wavelet coefficient buffer (if the transmission technique described in [Rausc99] is used).

As a result of exploiting the local image cache with $33.\overline{3}\%$ redundancy, even the computationally intensive techniques move and resize can be executed with interactive response behaviour.

6 EXAMPLE

To illustrate the possible amount of bandwidth and screen space saved, we will now describe a scenario where a scanned image of the Rostock public transport map (see figure 7) is transmitted via GSM and viewed online during transmission using the RECTANGULAR FISHEYE VIEW. The bandwidth of the simulated transmission channel is 7200 bits per second, which approximates the usable bandwidth when running HTTP via a GSM mobile phone. For more detail on the transmission method, please refer to [Rausc99].

Figure 7 shows the downscaling grid overlaid on the original 1024x1024 pixel image. The focus size is chosen to be 256x256 pixels, and there are two belts $belt_1$ and $belt_2$ with downscaling $Scl_1 = 2$ and $Scl_2 = 4$. Both belts are assigned the same ratio parameter value $Ratio_1 = Ratio_2 = 1$. That means, both belts will have the same width in the resulting fisheye view in the figures 3 to 6. Considerable savings in display space can be achieved using this configuration – the RECTANGULAR FISHEYE VIEW requires only 25% of the screen area needed for the original image.

By combining the viewing method with the transmission scheme proposed in [Rausc99], substantial bandwidth savings can be achieved, too – in theory, the same amount as for the screen space. In practice, the results slightly deviate from that since the compressibility of the image varies locally.

Prior to starting the image transmission, the initial position of the focus has to be determined. This may be done by using some context information, e.g., the position of the viewer or a point of interest. Here, it is assumed that the user plans to arrive in Rostock by train. That's why the focus is positioned into the main station area. Figure 3 shows the transmitted image after 27 seconds. Although only a small fraction of the image data has been transmitted at this early stage, the information in the focus region is recognisable. Compared to the 27 seconds transmission time for the RECTANGULAR FISHEYE VIEW, transmitting the whole image at the same degree of detail would have taken 115 seconds. Thus, the RECTANGULAR FISHEYE VIEW requires only 23.4% the bandwidth at this early stage. Spending these 27 seconds transmission time on the whole image would have lead to an incomprehensible result. Figure 8 illustrates that by comparing the quality of RECTANGULAR FISHEYE VIEW and whole image after 27 seconds.

At this point, the user moves the focus down and to the left. The viewing software immediately changes the layout based on the data already received; the left and the lower part of the new focus region are reconstructed from the available data at lower resolution (see figure 4). A request specifying the new grid is sent to the server, instructing it to change the transmission settings. Given the new configuration, only those data which have not yet been transmitted are now sent as differential refinement information. Six seconds after moving the focus (plus latency for sending the request to the server), the new focus region is available at the same degree of detail as the old one (see figure 5). Further progressive refinement is carried out automatically, and 88 seconds after starting the transmission, the readability of the whole fish eye view (see figure 6) has reached a stage which can not be further improved by transmitting more data (although the compression artefacts could still be reduced). Note that transmitting the whole image at the quality of the focus in figure 6 would have taken 321 seconds – the RECTANGULAR FISHEYE VIEW needs 27% of that time resp. bandwidth.

7 CONCLUSION

In this paper, we have presented the RECTANGULAR FISHEYE VIEW as a new technique for the interactive focus–and–context display of large raster images. Special attention has been paid to low computing requirements, low display space consumption and the opportunity to integrate redundancy–free image data transmission. This makes the technique especially suitable to be used in mobile environments.

These goals are achieved by overlaying a downscaling grid of rectangles on the image and to apply the same downscaling factor combination for X and Y direction to all pixels in the individual rectangles. Depending on the exact configuration of the grid, the screen space and transmission bandwidth savings can be substantial as described in section 6.

The interactivity of the technique offers a high degree of control to the user. This is necessary since the display technique saves space by using distortion. Rapid direct–manipulative interaction allows the user to explore the displayed image and to trade off space requirements and distortion. To support interactive response times, we have introduced a redundant storage mechanism for sparse image data representations.

REFERENCES

[Furna82] G.W. Furnas. The fisheye view: a new look at structured files. Technical memorandum 82-11221-22, Bell Labs, 18 October 1982.

[Keahe96] T.A. Keahey and E.L. Robertson. Techniques for non-linear magnification transformations. In *Proc. IEEE Symposium on Information Visualization, IEEE Visualization*, pages 38–45, October 1996.

[Keahe98] T.A. Keahey. The nonlinear magnification homepage. http://www.cs.indiana.edu/hyplan/tkeahey/research/nlm/nlm.html, 26 October 1998. Version 1.5.

[Rausc99] U. Rauschenbach. The rectangular fish eye view as an efficient method for the transmission and display of large images. In *Proc. IEEE International Conference on Image Processing (ICIP)*, Kobe, Japan, 25–28 October 1999.

[Sarka93] M. Sarkar, S.S. Snibbe, O. Tversky, and S.P.. Reiss. Stretching the rubber sheet. In *Proc. ACM Symposium on User Interface Software and Technology*, 1993.

ACKNOWLEDGEMENT

The authors wish to thank the Warnow Regional Public Transport Board and the Nitschke Advertising Studios for kindly permitting to use their map image data.

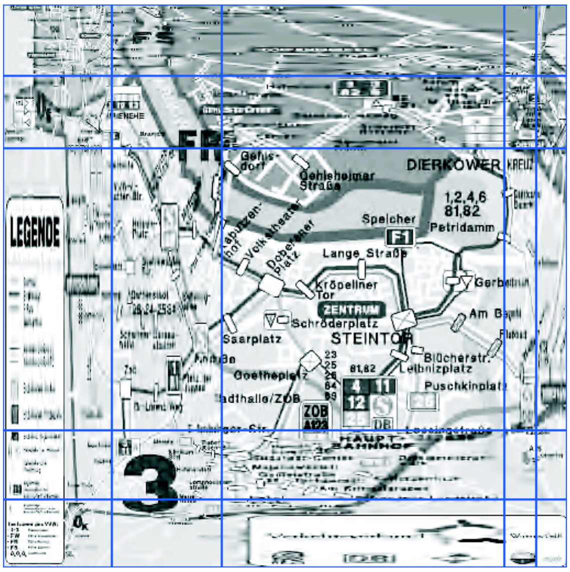

Figure 3: Initial RECTANGULAR FISHEYE VIEW, 512 x 512 pixels, 24562 bytes transmitted.

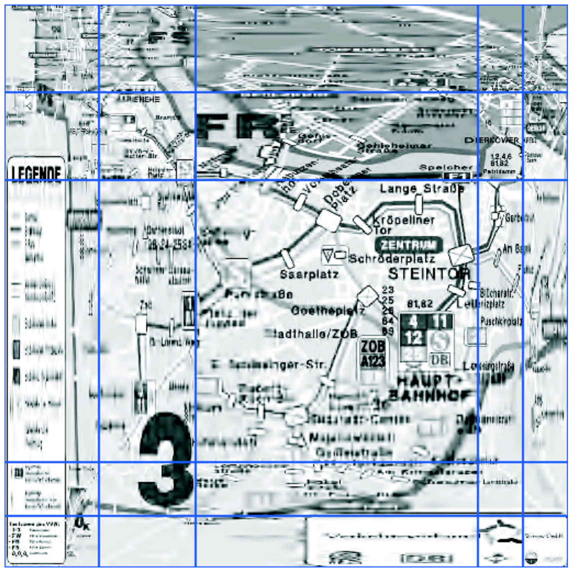

Figure 4: RECTANGULAR FISHEYE VIEW after moving the focus, 24562 bytes transmitted.

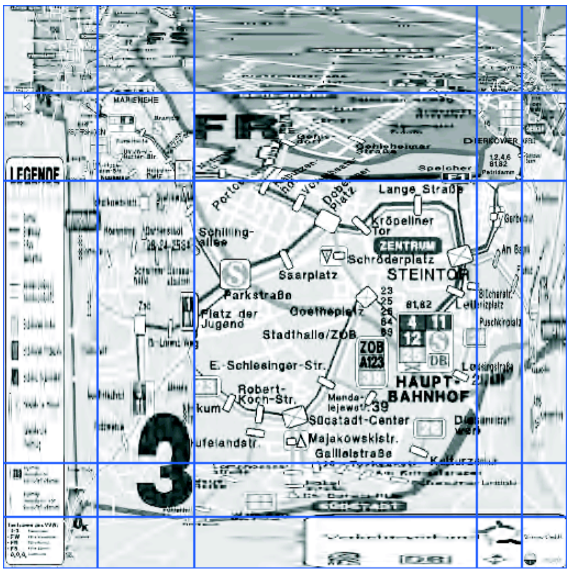

Figure 5: RECTANGULAR FISHEYE VIEW with the new focus, 30091 bytes transmitted.

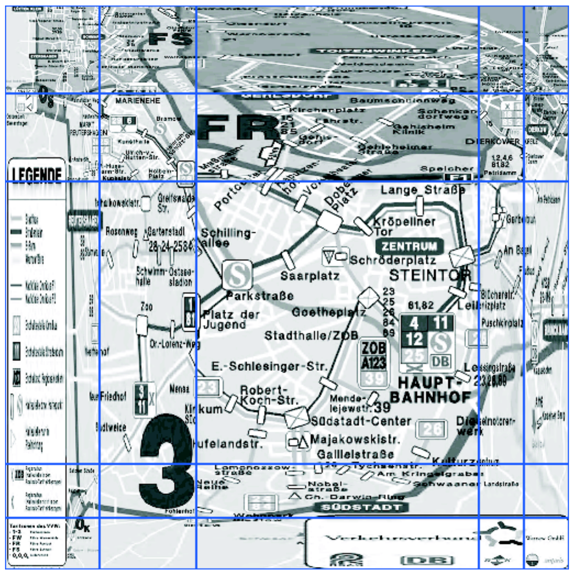

Figure 6: RECTANGULAR FISHEYE VIEW with the new focus, 79617 bytes transmitted.

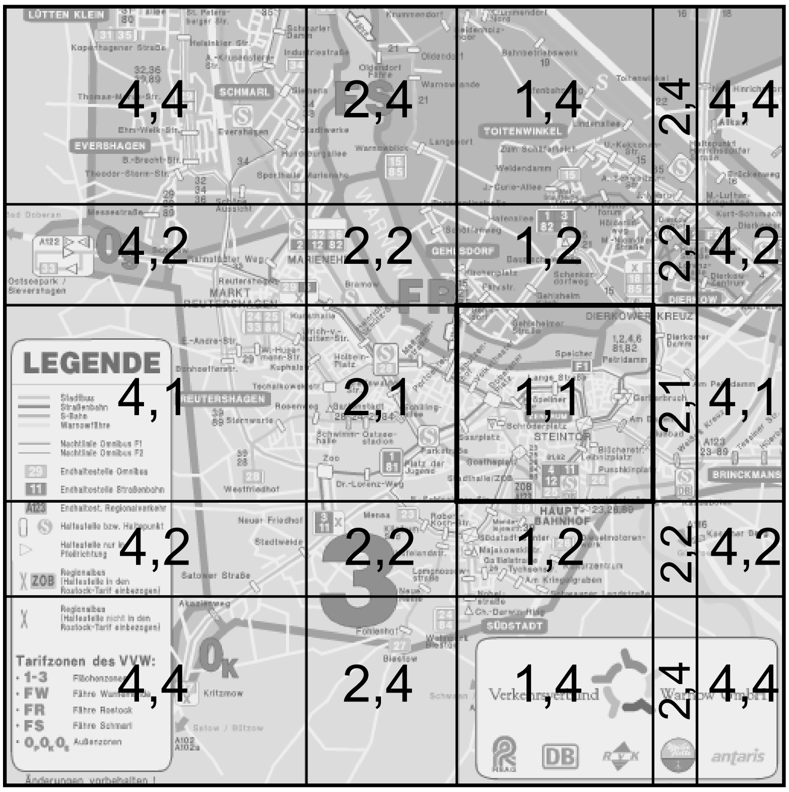

Figure 7: Original image with downscaling grid. 1024 x 1024 pixels.

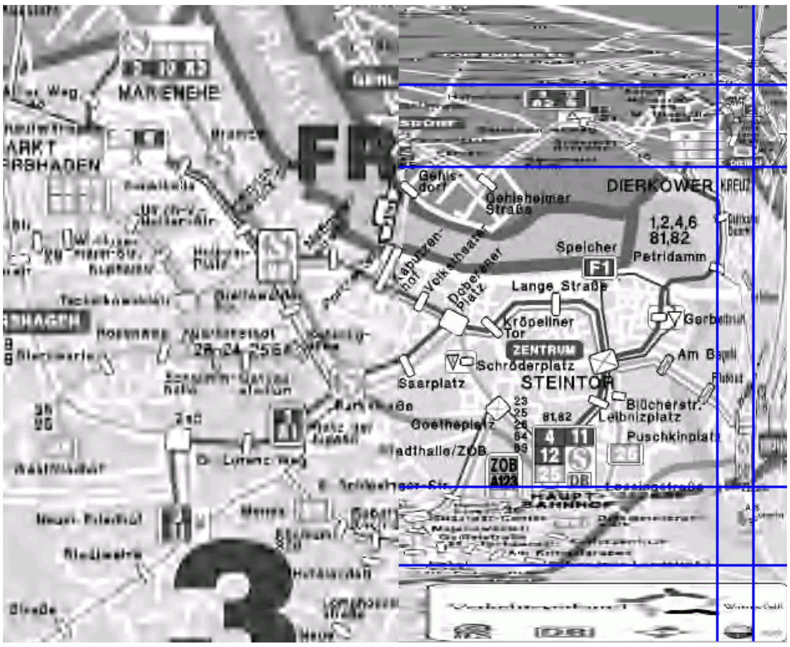

Figure 8: Quality comparison after transmitting 24562 bytes: RECTANGULAR FISHEYE VIEW from figure 3 (right) versus whole image (left).