FULLY AUTOMATIC STEREO-TO-MULTIVIEW CONVERSION IN AUTOSTEREOSCOPIC DISPLAYS

Christian Riechert, Frederik Zilly, Peter Kauff, Jens Güther, Ralf Schäfer
Fraunhofer Institute for Telecommunications Heinrich Hertz Institute, Germany

ABSTRACT
Watching 3D without glasses will be the future of 3D home entertainment. However, current stereoscopic 3D content is not suited to drive autostereoscopic displays and a 3D format conversion is required. In this paper a system is presented which is capable of high quality stereo-to-multiview conversion in real-time. This allows playback of 3D Blu-ray or any other stereoscopic 3D video content without prior adaptation on most existing autostereoscopic displays in real-time. The algorithms described include image rectification, fast and reliable disparity estimation, adaptive post-processing and up-sampling of disparity maps, rendering of virtual views and interweaving these views into a format suitable for autostereoscopic displays.

INTRODUCTION
Currently most 3D content is only available as stereo productions and so far storage and transport media have only been standardized for stereo video. On the other hand, autostereoscopic displays which appear on the market require more than two views, in general 5 to 10 or even more. Therefore fully automatic stereo-to-multiview converters at the consumer side are required to display available stereo content on autostereoscopic displays.

In this paper, a system is described, which is capable of high quality stereo-to-multiview conversion. It has been developed at Fraunhofer Heinrich Hertz Institute and is running in real-time as software solution on a high performance computer. The conversion algorithm itself consists of five main parts:

- a preprocessing part which analyses and rectifies the incoming stereo pair in order to ensure that only horizontal disparities exist between left and right image.
- a very fast disparity estimator, which generates temporally and spatially consistent disparity maps
- an adaptive multi-step filter unit for post-processing and up-sampling of the disparity maps to full resolution
- a depth image based rendering (DIBR) unit for rendering an arbitrary number of virtual camera views as required by a specific autostereoscopic device
- a post-processing and interweaving unit for filling disocclusions in the rendered views and merging all views to an output image suitable for the output device
The algorithms used in this solution were specifically designed to be suitable to be ported onto hardware platforms such as FPGAs or SoCs.

**RELATED WORK**

Fully automatic stereo-to-multiview conversion is an extremely challenging field, since rectification as well as disparity estimation is highly ambiguous and image based rendering is very sensible to errors in the underlying disparity maps. Therefore very few solutions exist so far. In Berretty et al (1) a video plus depth (V+D) approach is described, which relies on the ability of the target autostereoscopic screen to do the actual view rendering on its own and thus only provides a disparity map. In Lang et al (2) a different approach is presented, which is based on image-based saliency estimates and stereoscopic warping, instead of dense disparity map estimation. Both approaches have the common problem that the depth impression on an autostereoscopic display is rather limited, because V+D as well as saliency maps only allow little extrapolation without image degradation.

In this paper a complete processing chain for stereo-to-multiview conversion including rectification, dense disparity estimation and multi-view depth (MVD) rendering as illustrated in Fig. 1 is presented. The concept of image rectification in order to align epipolar lines of stereo image pairs is well known in literature (Hartley et al (3)). The estimation of rectification parameters in real-time and without prior knowledge of the relevant camera parameters on the over hand is addressed properly only recently by systems like the stereoscopic analyzer (STAN) (Zilly et al (4) (5)).

With rectified images, a disparity between the input image pair can be estimated. There already exist a wide variety of stereo matching algorithms. A comprehensive overview is for example provided by Scharstein et al (6). Most of these algorithms have in common that they are either too slow to be used in a real-time environment in HD resolution or the quality is not sufficient to use the resulting disparity maps for DIBR. In this paper we therefore use the recently developed line-wise hybrid recursive matcher (L-HRM) coupled with a bilateral-filter based post-processing and up-sampling step, which is real-time capable and provides high quality disparity maps.

The disparity maps are used to render virtual camera views via depth image based rendering (DIBR) (Mark (7), McMillan (8)). Initially only one image and the corresponding disparity map (V+D) were used. Later approaches improved the render results by providing additional occlusion information. Layered depth video (LDV) (Zitnick et al (9)) added additional occlusion layers to the disparity map and Multiview Video (MVD) (Müller et al (10), Smolic et al (11)) added additional image and disparity pairs at other camera positions to fill disoccluded regions in the rendered image. For application in a stereo-to-
Multiview conversion the MVD approach is most suitable, because there are already two images at different positions available and disparity maps can easily be estimated for both images.

STEREO ANALYSIS AND CORRECTION

The disparity estimation and by that the whole processing chain is extremely dependent on a well rectified input image pair. While for 3D Blu-ray content, the left and right view can generally be assumed to be fairly well rectified, this is definitely not true for live camera images. To guarantee rectified input images, the stereoscopic analyzer (STAN) (4) (5) is used to analyze the input images and rectify them if necessary.

Feature Point based Stereo Analysis

The stereo analysis by the STAN is done by first detecting a set of reliable image feature correspondences between left and right input image. As feature detector the STAN uses the semantic kernels binarized (SKB) detector (Zilly et al (12)). The SKB detector is a very fast blob detector and is optimized to find reliable feature point matches between stereo image pairs. These matches are used in a RANSAC based optimization step to determine differences between important physical camera parameters of left and right camera. These parameters include the difference in vertical tilt, rotation, keystone, zoom, and differences in vertical position. They can be used to calculate a pair of rectification matrices, which, if applied to the input camera images, produce a rectified output image pair.

DISPARITY ESTIMATION

After rectification, the disparity maps are estimated using the line-wise hybrid recursive matcher (L-HRM). The L-HRM is based on the hybrid recursive matcher (HRM) (Atzpadin et al (13), de Haan et al (14)). Though the HRM is real-time capable for small image sizes, it cannot take advantage of multi-core environments due to its recursive structure which prevents parallelization by design. The L-HRM breaks up part of the recursion in the HRM which allows for a parallel execution of lines and columns.

The L-HRM consists of two essential parts: the line- and column-wise recursion and the pixel recursion. The pixel recursion step introduces new disparity candidates to the estimator and is essential for the initialization of the disparity estimation and scenes with fast changing content. The line- and column-wise recursion propagates disparities of surrounding pixels and previous frames to the current pixel. This ensures spatial and temporal stability of the resulting disparity maps.
Pixel Recursion

The pixel recursion of the L-HRM is highly similar to the pixel recursion of the HRM. In order to find a reliable disparity estimate the neighbourhood of the current pixel is examined. Following the principle of optical flow, a disparity update value is calculated on the basis of spatial gradients and gradients between left and right input image as shown in eq. (1).

\[
d_{n+1}(x, y) = d_n - \Delta l(x, y, d_n) \cdot \frac{\nabla I_L(x, y)}{\|\nabla I_L(x, y)\|^2}
\]  

(1)

The gradient between left and right image is approximated by the intensity difference of corresponding pixels as calculated by the current disparity estimate as shown in eq. (2).

\[
\Delta l(x, y, d_n) = I_L(x, y) - I_R(x + d_n, y)
\]  

(2)

Since vertical disparities are of no concern, the spatial gradient reduces to a simple derivation in x-direction. As with optical flow the disparity is updated iteratively choosing a different position in the neighbourhood in each iteration. The disparity update value with the smallest absolute intensity difference between corresponding pixels in left and right image is chosen as new candidate for the line-wise respectively column-wise recursion.

Line-wise and column-wise recursion

In the line-wise respectively column-wise recursion the disparity candidate from the pixel recursion is compared to further candidates from previously calculated pixels in the neighbourhood and the previous frame. This is done using block comparisons of the current pixel’s neighbourhood with the corresponding region in the second stereo image. As quality measure the normalized cross-correlation is used.

The final disparity is calculated in a two-step approach. First a line-wise recursion is performed. In this step three disparity candidates are considered for each pixel: the horizontally preceding pixel’s disparity, the disparity of the same pixel in the previous frame and the resulting disparity of the pixel recursion. The initial disparity estimate for the pixel recursion is the best candidate of the first two candidates. The direction of the line-wise recursion is switched from left-to-right to right-to-left for each consecutive line.

The second step is the column-wise recursion. Here, only two candidate disparities are considered: the resulting disparity of the preceding line-wise recursion and the vertically preceding pixel’s disparity. As in the line-wise recursion the direction is switched from top-to-bottom to bottom-to-top on a column-wise basis.

Consistency Check and Basic Post-processing

The consideration of the previous disparity value as candidate at a given pixel ensures a high degree of temporal stability. The resulting disparity maps may still have false disparities, especially in occluded regions. Thus, a left-right consistency check and a small median filter to remove isolated false disparities are applied as shown in Fig. 2.

POST-PROCESSING AND UPSAMPLING

In a refinement and up-sampling step the disparity maps estimated by the L-HRM are post-processed to reduce artifacts and to resample the disparity maps to full resolution if necessary. For this, different variations of cross-bilateral filters are applied iteratively,
similar to the methods introduced in Kopf et al (15) and Riemens et al (16). First the raw disparity map is filtered by a cross-bilateral median filter of rather large size similar to the one proposed in Müller et al (17). Afterwards a cross-bilateral up-sampling filter with a small filter window is applied. Each application of the up-sampling filter increases horizontal and vertical resolution of the disparity map by a factor of 2 until full resolution of the input images is reached. To ensure temporal stability during the refinement, in the last iteration of the up-sampling filter the disparity of the previous frame are also taken into account. Unlike the L-HRM the post-processing uses RGB images as input instead of luminance images.

**Cross-bilateral median filtering**

The cross-bilateral median filter essentially works like the common cross-bilateral filter, but instead of a linear combination of all disparity values in the filter window, the weighted median of all disparity values is calculated with weights as shown in eq. (3).

\[
    w(x_i, y_j) = e^{-\frac{(x_i-x_0)^2 + (y_j-y_0)^2}{2\sigma_r^2}} \cdot e^{-\frac{(I(x_i,y_j)-I(x_0,y_0))^2}{2\sigma_c^2}}
\]  

(3)

To increase performance the first factor in the equation is set to 1 in this application, thus instead of applying a Gaussian range filter a uniform one is used. The weighted median filter ensures that only existing disparity values are possible output values. This prevents the introduction of averaged non-existing disparity values at disparity edges with similar colours in the input image. For this application a typical filter window size of 25x17 pixels is used.

**Cross-bilateral up-sampling**

As up-sampling filter a common cross-bilateral up-sampling filter is employed. The filter window sizes of the up-sampling filters are significantly smaller than the pre-up-sampling filter size, typically 5x5. Instead of the rather complex filter window topology used in (16), a standard rectangular window is used, since the rather small reduction in execution time due to non-rectangular windows could not offset the loss in quality. In the last iteration of the cross-bilateral up-sampling the disparity values of the previous frame are also taken into account. It showed that considering only both the current and the most recent frame is enough to ensure temporal stability.

**VIRTUAL VIEW RENDERING**

The full-resolution dense disparity maps which are the output of the previous processing step can now be used to render an arbitrary number of virtual views. Since the input images have been rectified in a previous step, the view rendering simplifies to a horizontal pixel shift depending on the pixel’s disparity value. For the view interpolation, that is virtual views between left and right original view, an MVD approach is used, for view extrapolation a V+D approach with adaptive disparity map pre-filtering is applied.

In both cases, the rendering is done in two distinct steps. First, the disparity map is rendered to its new position. For this forward mapping is used because backward mapping is not possible. Forwards mapping has the disadvantage that in general not every target position is hit by a forward mapped pixel and there might be pixels in the target image which were not hit at all. In order to have these pixels filled and to prevent depth errors resulting from rounding artefacts of the rendered disparity map, a continuous disparity
mapping instead of a per-pixel mapping is used. For this, two consecutive disparities are warped to their exact target positions and the disparities at all integer pixel position in between are interpolated linearly.

This works fine as long as there is no disparity gap between consecutive disparities. In this case there are two possible outcomes. In the first case, the two pixel positions change order during warping. This happens when one object occludes another. In the second case a disoccluded region, a region where no image information is available, is created. Therefore simple gap detection was implemented. If the absolute disparity difference of two adjacent pixels exceeds a threshold, a gap is assumed and no disparities in between are interpolated. Since it is possible that multiple source positions are warped to the same target position, a depth test is applied for each warped pixel. Only if the new depth is nearer to the camera than the current depth, the target pixel's disparity is overwritten.

When the disparities have been warped, the actual image can be rendered via backward mapping in a second render step. The target pixel's colour value can be determined by any kind of standard interpolation filter in the source image.

**View interpolation using MVD2**

The MVD2 approach (illustrated in Fig. 3) is used for all views rendered between left and right original view. Both the left and the right view are warped to the virtual camera position independently as described above. In a second step they are combined to a single virtual view. Both original views are used, because there can be disocclusion in the virtual view from one camera that can be filled by the virtual view of the other camera. The mixing is done using the distances of the target view to the source views as mixing coefficients. To avoid visible colour differences in image areas where only one view is available, the borders of these areas are smoothly blended with the rest of the image.

**View extrapolation using Video-plus-Depth**

If the target view is not between left and right original view, only the nearest view is used to render the virtual view. In this case the disparity map is pre-filtered to avoid large disparity gaps, which would result in disocclusions in the target view. The filtering is done using an adaptive Gaussian blur, whose variance $\sigma$ depends on the disparity distance of adjacent pixels. It is taken specific care, that foreground objects are not excessively stretched by the filtering, because in general a stretching of the background introduces less visible artefacts. The V+D render scheme is illustrated in Fig. 4.
Even though in both view interpolation as well as view extrapolation various methods to reduce disoccluded areas are used, not all disocclusion could be taken care of. Thus, in the last step these rather small disocclusions have to be filled in a visually pleasing way. Since the whole algorithm chain is supposed to run in real-time and the disocclusions at this point are rather small, a sophisticated inpainting is not feasible and a simple pixel repetition is used. For this, the disparities of the rendered image are analysed and disoccluded areas are filled with the colour of the nearest background pixel.

A typical effect which occurs in rendered virtual views is that foreground objects look as if they were stamped into the background. This is caused by sharp object edges introduced by the DIBR. In natural images the change between foreground and background is rather smooth. Thus, an adaptive smoothing of the image at disparity edges is applied. This also reduces visible artefacts introduced by disparity flickering at object edges.

Interweaving

In a final step, all rendered views have to be interwoven into a single image of full display resolution in a display specific pattern. Since an Alioscopy autostereoscopic display is used, 8 views are rendered and the Alioscopy pattern is applied to merge them to a single full HD resolution image, which can be shown directly on the screen.

EVALUATION

The whole processing chain was implemented on a Dual-processor Intel Xeon X5690 system with two Nvidia Geforce GTX 590 graphics cards. The pre-processing including the STAN based stereo analysis and rectification as well as the disparity estimation was executed on CPU. For this, the initial disparities were only estimated on a 4x4 times subsampled image pair. The processing time saved by the sub-sampling was instead used to make several disparity estimation iterations per frame. This ensures a much better adaptation of the disparity maps to fast changing scenes. The disparity map post-processing and up-sampling as well as the rendering and disocclusion filling was implemented as CUDA code and executed on GPU.

In Fig. 5 the results of each consecutive step of the processing chain are visualized. The complete stereo-to-multiview conversion solution runs at a minimum of 24 fps using as input a pair of Full HD images. If not otherwise specified all processing, in particular the view rendering, is done at full resolution as well. This enables us to support future autostereoscopic displays with higher resolution than HD with only minor adaptations to the processing chain.
CONCLUSION

In this paper a processing chain was presented which enables high quality live playback of conventional stereo video content on an autostereoscopic display. The current implementation of the chain runs in real-time as software on any recent high-performance PC. The involved algorithms were designed to be suitable for implementation on hardware platforms such as FPGAs or SoCs.
REFERENCES


18. 3D4YOU, FP7 project funded by European Commission. FP7-ICT-2007-215075.