Proc. of the IEEE Workshop on Computer Architectures for Machine Perception CAMP´95, 18 - 20 September 1995, Como, Italy

# Towards Real-Time Stereo Employing Parallel Algorithms For Edge-Based And Dense Stereo Matching

Andreas Koschan, Volker Rodehorst

Institute for Technical Informatics, FR 3-11
Technical University of Berlin, Franklinstr. 28-29
10587 Berlin, Germany
e-mail: koschan@cs.tu-berlin.de, vr@cs.tu-berlin.de

## Abstract

*Only few problems in computer vision have been investigated more vigorously than stereo. Nevertheless, the main obstacle on the way to their practical application is the excessively long computation time needed to match stereo images. This paper presents parallel algorithms for edge-based stereo suitable for depth computation. Edge-based stereo techniques produce only sparse depth maps. Thus, we present in addition an efficient parallel algorithm for dense stereo matching that can be employed in scene reconstruction. Both approaches are implemented on several different computers to measure the performance. We compared single processor and multiple processor implementations to evaluate the profit of parallel realizations. Results are presented in this paper. We show that both approaches are very suitable for parallel implementations and that computing time can be considerably reduced with parallel implementations. Furthermore, we present the results that are obtained when employing the different approaches to stereo images.*

## 1 Introduction

Stereo is a well-known technique for obtaining depth information from digital images. The key problem in stereo is how to find the corresponding points in the left and in the right image, referred to as the correspondence problem. Whenever the corresponding points are determined, the depth can be computed by triangulation. Excessively long computation time needed to match stereo images is still the main obstacle on the way to the practical application of stereo vision techniques. Computational fast stereo techniques are required for real-time applications, especially for mobile robots and autonomous vehicles. General purpose computers are not fast enough to meet real-time requirements because of the algorithmic complexity of stereo vision techniques. Consequently, the use of parallel algorithms and/or special hardware is inevitable to reach real-time execution.

Stereo techniques can be distinguished by either matching edges and producing sparse depth maps or matching all pixels in the images and producing dense depth maps. The objective of the application always effects the decision whether the preference is given to dense stereo correspondence or to edge-based correspondence. For a successful reconstruction of complex surfaces it is essential to compute dense disparity maps defined for every pixel in the entire image.

Unfortunately, most of the existing dense stereo techniques are very time consuming (see e.g. [1, 2]). In an earlier investigation [3], we found the Block Matching technique using color information to be very suitable for dense stereo. The precision of the matching results always improved by 20 to 25 % when using color information instead of gray value information. Thus, high quality matching results can be easily obtained with this technique. In this paper, we present parallel algorithms for obtaining dense depth maps from color stereo images employing this approach. We believe that robotics applications do not neglect dense depth information if this information can be obtained quickly. Nevertheless, dense depth maps are not always required for every application. Therefore, we present in addition a parallel algorithm for edge-based stereo correspondence.

## 2 Chromatic Block Matching for Dense Stereo Correspondence

During the past couple of years some hardware stereo implementations were already presented. Neural networks and transputers are, for example, successfully employed for stereo [4, 5] and a parallel stereo algorithm implemented on the TMC Connection Machine was presented [6]. None of these implementations produces dense depth maps. As mentioned before, we found the Block Matching technique using color information to be very suitable for dense stereo matching [3]. The main idea of Block Matching is a similarity check between two equal sized blocks (n x m-matrices) in the left and the right image (area-based stereo). The mean square error *MSE* between the pixel values inside the respective blocks defines a measure for the similarity of two blocks. *MSE* is defined for gray value images as:

$$MSE(\Delta) = \frac{1}{m \cdot n} \sum_{i=1}^{n} \sum_{j=1}^{m} \left| I_L(i,j) - I_R(i+\Delta, j) \right|^2 , \quad (1)$$

where $I_L$ and $I_R$ are the intensity functions of the left and right image and $\Delta$ is an offset describing the difference ($x_L$ - $x_R$) between the column positions in the left and in the right image. This formula can be easily extended to color images, when employing a color measure. We found [3] that the selection of the color measure has no significant influence on the quality of the matching results. Therefore, we propose to employ an approximation of the Euclidean distance denoted as $D_c$. For two colors $f_1 = (R_1, G_1, B_1)$ and $f_2 = (R_2, G_2, B_2)$ in the *RGB* color solid the measure $D_c$ is defined as:

$$\begin{aligned} D_c(f_1, f_2) = & \left| R_1(i,j) - R_2(i,j) \right|^2 + \\ & \left| G_1(i,j) - G_2(i,j) \right|^2 + \\ & \left| B_1(i,j) - B_2(i,j) \right|^2 . \end{aligned} \quad (2)$$

The left color image $F_L$ and the right image $F_R$ may be represented in the *RGB* color space as $F_L(i,j) = (R_L(i,j), G_L(i,j), B_L(i,j))$ and $F_R(i,j) = (R_R(i,j), G_R(i,j), B_R(i,j))$. Now (1) changes to

$$MSE_{color}(\Delta) = \frac{1}{m \cdot n} \sum_{i=1}^{n} \sum_{j=1}^{m} D_c(F_L(i,j), F_R(i+\Delta, j)) . \quad (3)$$

The block (of size n x m) is shifted pixel by pixel inside the search area. The disparity *D* between the blocks in both images is defined by the distance between the positions (the difference in the columns) of the blocks, showing the minimum mean square error in both images. Furthermore, the search area in the right image is limited in the horizontal direction by a predefined maximum disparity $d_{\max}$:

$$D = \min_{|\Delta| \le d_{max}} \left\{ MSE_{color}(\Delta) \right\} . \quad (4)$$

Block disparities are median filtered to avoid outliers. An explicit disparity value is computed for every pixel by extending a pixel selection method [7] to color images. A dense disparity map is generated when applying this pixel selection technique to every pixel in the image. Afterwards, median filtering is applied to pixel disparities. For further details see [3].

## 3 Parallel Algorithms for Chromatic Block Matching

Several ways exist to develop parallel algorithms for chromatic Block Matching. The first variant we implemented was to use three processing units for computing *MSE* functions separately in every color channel. Once the *MSE*s are obtained, the results are combined to find a criterion for Block Matching. Unfortunately, no decrease in computing time occurred since the overhead for dividing the processes and for combining the results again was too large. This is due to our available hardware and software configuration and does not hold in principle. We intend to find a more suitable configuration to continue with this idea.

Currently, we divide both images into several segments and we compute *MSE*s for every segment in parallel. For example, both color images can be divided into 8 segments. Now, *MSE*s can be computed in parallel for every segment using 8 processing units (*PU*s). An illustration of this procedure is given in Fig. 1. In principle, both images can be divided into many segments (e.g., 70 segments for PAL resolution). Utilizing an individual processing unit for every segment will speed up the matching process.

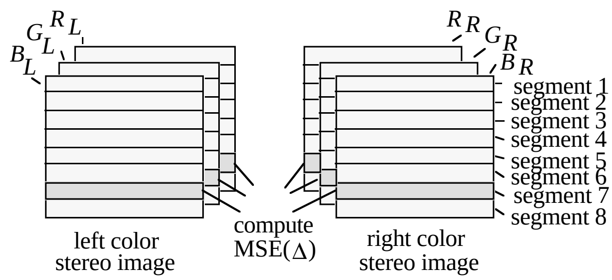


Figure 1: Illustration of the parallel Block Matching algorithm showing by example the computation of the MSE in the segment number 7 (shadowed area) in three color components in both color images.

In an earlier investigation [3], we found a slight improvement in the quality of the matching results when employing the $I_1 I_2 I_3$ color space suggested in [8] instead of the $RGB$ color space. The three coordinates are defined by

$$I_1 = \frac{R+G+B}{3},\ I_2 = \frac{R-B}{2},\ \text{and}\ I_3 = \frac{2G-R-B}{4}.$$

Image data have to be transformed from $RGB$ to $I_1 I_2 I_3$ when this color space is used. Nevertheless, the principle of dividing a color image into several segments holds for every tristimulus color solid.

A variant of the median filter, the separable *"median of medians"* [9], was implemented to accelerate image smoothing. A one-dimensional median is first determined in each row and afterwards in each column for all rows and columns inside a two-dimensional $(2n+1)$x$(2n+1)$ mask. The final result is the median of these 2 $(2n+1)$ median values. Although the output is not identical to the output of the two-dimensional filter, the quality is very close to it and the algorithm is easier to implement in real-time hardware. Furthermore, we implemented pixel selection for every segment in parallel. The resulting parallel algorithm for chromatic Block Matching is outlined in Fig. 2.

```
BEGIN
  { Transform the left and right image i
    from RGB to I1I2I3 color space }
  PARALLEL DO (in PU_i , 1 ≤ i ≤ 2)
      ConvertRGBtoI1I2I3_i ()
  END PARALLEL

  { Search for corresponding blocks in
    horizontal segments by minimizing the MSE }
  PARALLEL DO (in PU_s , 1 ≤ s ≤ 70)
      FOR d = 2 TO d_max DO
          BlockMatching_s (d)
      END FOR
  END PARALLEL

  { Filter the block disparity image with a
    median approximation in horizontal and
    vertical segments }
  PARALLEL DO (in PU_s , 1 ≤ s ≤ 70)
      BlockMedian_s ()
  END PARALLEL

  { Compute pixel disparities from block
    correspondences }
  PARALLEL DO (in PU_s , 1 ≤ s ≤ 70)
      SelectPixel_s ()
  END PARALLEL

  { Apply the median approximation to the
    pixel of the disparity image in horizontal and
    vertical segments }
  PARALLEL DO (in PU_s , 1 ≤ s ≤ 70)
      PixelMedian_s ()
  END PARALLEL
END
```

Figure 2: Parallel Algorithm of Chromatic Block Matching (with up to 70 processing units *PUs*).

## 4 A Parallel Algorithm for Edge-Based Stereo Correspondence

Dense depth maps are not always required for every application, and their computation is time consuming. Often, the computation of distances between the camera system and the objects in the scene is the exclusive goal of the stereo task. Thus, the correspondence search in stereo images can be reduced to the matching of the most prominent parts in the images. These are the edges. Edge-based stereo techniques have the advantage of being less sensitive to photometric variations. In an earlier investigation [10], we found that high quality edge matching results are obtained when a feature-based technique suggested by Shirai and Nishimoto [11] is applied to the stereo images. The main idea of this binocular approach is based on disparity histograms showing the distribution of disparity values in the neighborhood of matching candidates in multiple resolutions. A standard stereo geometry is used to reduce the search space to horizontal lines.

Edges are extracted in both (intensity) stereo images applying the Marr-Hildreth or LoG operator [12], respectively, in three resolutions ($\sigma_1$ = 1.41, $\sigma_2$ = 3.18, and $\sigma_3$ = 6.01). Zero-crossings (ZCs) in the LoG filtered images constitute the features in the succeeding matching process. We do not concentrate on the parallel implementation of the Marr-Hildreth operator since Tremblay, Savard, and Poussart [13] presented a hardware implementation. The so-called Multiport Access photo-Receptor (MAR) is a CMOS sensor and represents the sensory part of their integrated image acquisition system. The whole system implements the Marr-Hildreth edge detection scheme in 16 resolutions using VLSI architecture. Linear edge segments extracted from the image are transferred to the host computer. Currently, a 256 x 256 pixel version is realized but a 500 x 500 pixel version is announced. In this paper, we implemented edge detection in parallel for the three resolution channels.

Now, the basic idea of the edge-based stereo approach will be explained (cp. [11]). A ZC may be defined as two-dimensional unit vector $e(i,j)$ along the ZC line. A pair of ZCs in the right and left images is regarded as matching candidate, if the difference between the directions of ZCs is less than 30 degrees. This can be represented by the following matching functions $M^R$ and $M^L$ for the right and left image, respectively: if $e^R(i,j)$ and $e^L(i+d,j)$ are a matching pair, $M^R(i,j;d) = 1$ and $M^L(i+d,j;d) = 1$. Otherwise, $M^R(i,j;d) = 0$ and $M^L(i+d,j;d) = 0$, where $d$ denotes a disparity.

First, the global disparity histogram ($GDH$) is determined to find approximate disparity intervals. The $GDH$ represents the distribution of candidate disparities (including true and false matches) in the entire image. It is defined for the right image as:

$$GDH^{(R)}(d) = \frac{\sum_{(i,j)\in A} M^{(R)}(i,j;d)}{\sum_{(i,j)\in A} \left|e^{(R)}(i,j)\right|}, \tag{5}$$

where $A$ is the whole image plane. $GDH^{(R)}$ alone is sufficient to estimate the disparity distribution. Computing the $GDH$ can be easily implemented in parallel (see Fig. 3).

```
BEGIN
 {Compute global disparity histogram for each
 channel s_v with σ1 = 1.41, σ2 = 3.18, σ3 = 6.01}
 PARALLEL DO (1 ≤ v ≤ 3)
    PARALLEL DO (in PU_vw , 1 ≤ w ≤ 8)
       FOR k = (number of rows/8)*(w-1)
       TO (number of rows/8)*w  DO
          FOR l = 1 TO number of columns DO
             Compute GDH_vw^(R) (d)
          END FOR (l)
       END FOR (k)
    END PARALLEL
    Initialize GDH_v^(R)
    FOR k = 0 TO number of columns - 1 DO
       FOR j = 1 TO 8 DO
          GDH_v^(R) (k) ← GDH_v^(R) (k) +
                            GDH_vj^(R) (k)
       END FOR (j)
    END FOR (k)
 END PARALLEL
END
```

Figure 3: Parallel algorithm for the computation of global disparity histograms (with 24 processing units $PU_{vw}$, $1 \le v \le 3$, $1 \le w \le 8$).

Based on the $GDH$, a candidate disparity interval $I_\alpha$ is determined in the following equation.

$$I_\alpha = \left\{ d \mid GDH^{(R)}(d) > a H \right\},$$

where $H$ is the peak value of $GDH^{(R)}(d)$ and $a$ is a constant value with $0 < a < 1$. Local disparity candidates are estimated using local disparity histograms ($LDH$). The $LDH$ shows the disparity distribution of true and false matches within the window $W_\sigma$ of size $N_\sigma$ x $N_\sigma$ around a ZC point, where $N_\sigma = \sqrt{2}\pi\sigma$. The $LDH$ for a ZC at $(i,j)$ in the image with $d \in I_\alpha$ is defined as:

$$LDH^{(X)}(d;i,j) = \frac{\sum_{(i,j)\in W} M^{(X)}(i,j;d)}{\sum_{(i,j)\in W} \left|e^{(X)}(i,j)\right|}. \tag{6}$$

$LDH$s are determined for the left and the right image ($X = L$ and $X = R$). The parallel computation of $LDH$s can be easily realized employing the following algorithm outlined in Fig. 4.

```
BEGIN
{Compute local disparity histogram for each
channel σ_v with σ1 = 1.41, σ2 = 3.18,
σ3 = 6.01 and d ∈ I_α}
PARALLEL DO (1 ≤ v ≤ 3)
   PARALLEL DO (in PU_vw , 1 ≤ w ≤ 8)
      FOR k = (number of rows/8)*(w-1)
      TO (number of rows/8)*w  DO
         FOR l = 1 TO number of columns DO
            Compute LDH^(L) (d;k,l)
            Compute LDH^(R) (d;k,l)
         END FOR (l)
      END FOR (k)
   END PARALLEL
END PARALLEL
END
```

Figure 4: Parallel algorithm for the computation of local disparity histograms (with 24 processing units $PU_{vw}$, $1 \le v \le 3$, $1 \le w \le 8$).

Once $LDHs$ of all channels are computed, a best channel is selected for every window based on the first and second largest peaks in $LDH$. If the difference between the two peaks is the largest, the channel is selected. A function $F$ is defined to check the reliability of the selected channel. $F^{(X)}(d_X; i,j)$ is the difference between the largest peaks of the best channel in the window around $(i,j)$ in image $X$ ($X = L$ oder $X = R$). $d_X$ is the disparity showing the largest peak. Matching is established if the values of the $F$ functions are large and the difference between the disparity values in the right and left images is small.

Once the most probable disparity $d^*$ is obtained in $W_\sigma$, disparities of all ZC points in $W_\sigma$ and those of finer channels in $W_{\sigma'}$ ($\sigma' < \sigma$) are obtained by searching for the optimum disparity $d_k$ being the closest to $d^*$ (for further details see [11]). Whenever a pair of ZCs is matched, they are removed from ZC sets to reduce the number of remaining candidates. After trying to establish matches for all ZCs inside the window $W_\sigma$ and the windows of finer resolutions, the algorithm (starting with *GDH*s) is applied to the reduced feature lists. The matching process terminates, if no new matches can be established.

In our parallel implementation, we detect edges in the left and in the right image in three resolutions in parallel. Afterwards, the *GDH*, the candidate disparity intervals, and the *LDH*s are determined in parallel for the three resolutions. The resulting parallel algorithm is outlined in Fig. 5.

**BEGIN**
  *{ Search for zerocrossings in the left and right image i for each channel c with $\sigma_1$ = 1.41, $\sigma_2$ = 3.18, $\sigma_3$ = 6.01 }*
  **PARALLEL DO** (in $PU_{ic}$, $1 \le i \le 2$, $1 \le c \le 3$)
    FeatureExtraction$_{ic}$ ()
  **END PARALLEL**

  **DO**
    *{ Compute global disparity histogram and candidate disparity interval for all channels independent }*
    **PARALLEL DO** (in $PU_c$, $1 \le c \le 3$)
      Compute $GDH_c$ ()
      Compute $CDI_c$ ()
    **END PARALLEL**

    **FOR** (Each feature in channel 0) **DO**
      *{ For each channel c calculate the local disparity histogram and determine the existence and magnitude of a peak }*
      **PARALLEL DO** (in $PU_c$, $1 \le c \le 3$)
        Compute $LDH_c$ ()
        Compute $FXY_c$ ();
      **END PARALLEL**

      *{ Try to match the features in c and all channels with finer resolution}*
      $c \leftarrow$ SelectBestChannel ();
      **IF** (TestReliabilty ($c$) = OK) **THEN**
        MatchAndDeletePair ($c$);
      **END IF**
    **END FOR**
  **WHILE** (New features were matched)
**END**

Figure 5: Parallel algorithm of the edge-based stereo approach using disparity histograms.

## 5 Experimental Results with Different Hardware Configurations

We implemented the algorithms on the following machines: a Sun SPARC 10-40 (in the following denoted in brief as SPA10) using GNU C, a Sun SPARC 20-612 (denoted as SPA20) also using GNU C, a SGI Indigo (denoted as IND) with a R4400 processor (150 MHz) using IRIX C, and a SGI Power Challenge (denoted as POWER) with twelve R8000 processors (75 MHz) using IRIX Power C. We should like to emphasize that different compilers are used on the different machines. Therefore, we do not present a comparison between different computers. Opposed to this, we exclusively intend to illustrate the profit of parallel implementations. By way of example, we present some results obtained on different machines when using our different implementations.

Both approaches were applied to several images of different sizes. Due to lack of space, in this paper we present performances for two images. One image called BEETHOVEN represents a bust of Beethoven, a book, and a phone (see Fig. 6 and Fig. 7 in the appendix). The image has PAL resolution (752 x 566 pixel). A continuos acceleration occurred up to a factor of 8 when applying the chromatic Block Matching using 10 *PU*s (see Tab. 1). These results encourage an implementation on a highly parallel architecture. Opposed to this, the computing time consumed by the edge-based approach could only be reduced by a maximum factor of 3 when utilizing 6 *PU*s. Since our compiler does not support hierarchical parallelism, we did not obtain an increase in the performance by adding further *PU*s (see Tab. 3).

The computational cost of both approaches does not depend linearly on the image size. Thus, in addition we applied both approaches to an image of size 256 x 256 pixel. The image called PYRA represents a pyramid and a coffee cup. Results are shown in Tab. 2 and Tab. 4.

Due to lack of space only the matching and the reconstruction results for the BEETHOVEN image are presented in the appendix.

| Computing time [in sec] | SPA10 | SPA20 | IND | POWER 1 PU | POWER 3 PUs | POWER 6 PUs | POWER 10 PUs |
|---|---|---|---|---|---|---|---|
| Conversion RGB to I1I2I3 left | 0.75 | 0.53 | 0.41 | 0.17 | 0.17 | 0.17 | 0.17 |
| right | 0.75 | 0.53 | 0.40 | 0.17 | 0.17 | 0.17 | 0.17 |
| Estimating Block Disparities | 104.25 | 70.20 | 54.37 | 26.32 | 9.16 | 4.77 | 3.05 |
| Block Median | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 |
| Pixel Selection | 9.86 | 6.65 | 5.14 | 4.28 | 1.51 | 0.81 | 0.51 |
| Pixel Median | 0.92 | 0.42 | 0.26 | 0.36 | 0.13 | 0.08 | 0.07 |
| **Total** | 116.61 | 78.86 | 60.59 | 31.30 | 11.16 | 6.01 | 3.98 |

Table 1: Chromatic Block Matching applied to the image BEETHOVEN (752 x 566 pixels).

| Computing time [in sec] | SPA10 | SPA20 | IND | POWER 1 PU | POWER 3 PUs | POWER 6 PUs | POWER 10 PUs |
|---|---|---|---|---|---|---|---|
| Conversion RGB to I1I2I3 left | 0.12 | 0.08 | 0.06 | 0.02 | 0.02 | 0.02 | 0.02 |
| right | 0.12 | 0.08 | 0.06 | 0.02 | 0.02 | 0.02 | 0.02 |
| Estimating Block Disparities | 10.87 | 6.85 | 5.65 | 2.45 | 0.89 | 0.50 | 0.35 |
| Block Median | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| Pixel Selection | 1.63 | 1.05 | 0.79 | 0.63 | 0.23 | 0.13 | 0.08 |
| Pixel Median | 0.10 | 0.06 | 0.03 | 0.06 | 0.02 | 0.01 | 0.01 |
| **Total** | 12.85 | 8.12 | 6.59 | 3.18 | 1.18 | 0.68 | 0.48 |

Table 2: Chromatic Block Matching applied to the image PYRA (256 x 256 pixels).

| Computing time [in sec] | SPA10 | SPA20 | IND | POWER 1 PU | POWER 3 PUs | POWER 6 PUs | POWER 10 PUs |
|---|---|---|---|---|---|---|---|
| Feature extraction left channel 0 | 25.48 | 16.79 | 5.94 | 2.95 | 3.17 | 3.24 | 3.30 |
| channel 1 | 67.81 | 44.31 | 15.35 | 5.78 | 6.03 | 6.14 | 6.21 |
| channel 2 | 210.16 | 136.78 | 61.49 | 16.23 | 16.52 | 16.69 | 16.71 |
| right channel 0 | 25.34 | 16.47 | 5.85 | 2.92 | 2.94 | 3.32 | 3.29 |
| channel 1 | 67.63 | 44.52 | 15.74 | 5.80 | 5.82 | 6.10 | 6.19 |
| channel 2 | 207.99 | 136.92 | 62.35 | 16.24 | 16.37 | 16.73 | 16.80 |
| Subtotal | 604.41 | 395.79 | 166.72 | 49.96 | 22.53 | 16.78 | 16.97 |
| Edge Matching | 183.13 | 115.58 | 74.90 | 62.40 | 31.29 | 32.28 | 33.67 |
| **Total** | 787.54 | 511.37 | 241.62 | 112.36 | 53.82 | 49.06 | 50.64 |

Table 3: Edge-based stereo applied to the image BEETHOVEN (752 x 566 pixels).

| Computing time [in sec] | SPA10 | SPA20 | IND | POWER 1 PU | POWER 3 PUs | POWER 6 PUs | POWER 10 PUs |
|---|---|---|---|---|---|---|---|
| Feature extraction left channel 0 | 3.70 | 3.02 | 1.29 | 0.42 | 0.47 | 0.49 | 0.52 |
| channel 1 | 10.65 | 8.63 | 4.92 | 0.88 | 0.93 | 0.96 | 0.96 |
| channel 2 | 34.41 | 27.50 | 19.94 | 2.55 | 2.61 | 2.64 | 2.65 |
| right channel 0 | 3.68 | 2.98 | 1.28 | 0.44 | 0.43 | 0.48 | 0.51 |
| channel 1 | 10.76 | 8.60 | 4.91 | 0.89 | 0.92 | 0.95 | 0.97 |
| channel 2 | 34.42 | 27.45 | 19.92 | 2.56 | 2.60 | 2.63 | 2.6 |
| Subtotal | 97.63 | 78.19 | 52.26 | 7.77 | 3.56 | 2.66 | 2.73 |
| Edge Matching | 4.34 | 3.47 | 0.54 | 0.82 | 0.58 | 0.63 | 0.65 |
| **Total** | 101.97 | 81.66 | 52.80 | 8.59 | 4.14 | 3.29 | 3.38 |

Table 4: Edge-based stereo applied to the image PYRA (256 x 256 pixels).

## 6 Conclusion

Several approaches for parallel stereo matching have been presented. It has been shown that chromatic Block Matching can be implemented very efficiently in parallel. The results obtained so far encourage an implementation on a highly parallel architecture. Although we analyze color information for stereo matching due to quality requirements, we achieve high speed execution. Furthermore, we presented parallel implementations of the edge-based approach. Their performance is rather acceptable when using a non-hierarchical implementation. Nevertheless, a more efficient realization is expected by implementing our hierarchical parallel algorithm. Additional tests and investigations are necessary to improve the results and to decrease computing time. Currently, this is under investigation and further results will be presented soon.

In summary, we should like to emphasize that computing time is considerably reduced if our parallel algorithms are implemented on multi-processor machines. Therefore, we believe that precise results can be efficiently obtained in dense stereo matching and in edge-based stereo matching when one of the presented parallel algorithm is utilized.

## Acknowledgment

This work was funded by the Deutsche Forschungsgemeinschaft (DFG). Furthermore, we should like to thank Karsten Schlüns and Kathrin Spiller for helpful assistance and we should like to thank the ZEDAT at Free University of Berlin for hardware supply.

## References

[1] J.R. Jordan III and A.C. Bovik, "Using Chromatic Information in Dense Stereo Correspondence," *Pattern Recognition*, Vol. 25, No. 4, pp. 367-383, 1992.

[2] M. Okutomi, O. Yoshizaki, and G. Tomita, "Color Stereo Matching and its Application to 3-D Measurement of Optic Nerve Head," *Proc. 11th IAPR Int. Conf. on Pattern Recognition*, the Hague, the Netherlands, Vol. I, pp. 509-513, 1992.

[3] A. Koschan, "Dense Stereo Correspondence Using Polychromatic Block Matching," *Proc. of the 5th Int. Conf. on Computer Analysis of Images and Patterns CAIP'93*, D. Chetverikov, W. Kropatsch (Eds.), Budapest, Hungary, pp. 538-542, 1993.

[4] J.P. Muller, G.P. Otto, K.W. Chau, K.A. Collins, N.M. Dalton, T. Day, I.J. Dowman, D. Gugan, D. Morris, M.A. O´Neill, J.G.B. Robert, A. Stevens, and M. Upton, "Real-time stereo matching using transputer arrays," *Proc. IGARSS 88*, Edinburgh, GB, pp. 1185-1186, 1988.

[5] N.M. Nasrabadi and C.Y. Choo, "Hopfield Network for Stereo Vision Correspondence," *IEEE Trans. on Neural Networks* , Vol. 3, No. 1, pp. 5-13, 1992.

[6] J.-S. Chen and G. Medioni, "Parallel Multiscale Stereo Matching Using Adaptive Smoothing," *Proc. 1st ECCV*, Antibes, France, pp. 99-103, 1990.

[7] T. Reuter, "HDTV standards conversion," *Proc. IEEE-ASSP & EURASIP 5th Workshop on Multidimensional Signal Processing*, Nordwijkerhouth, the Netherlands, 1987, (reprint).

[8] Y.-I. Ohta, T. Kanade, and T. Sakai, "Color Information for Region Segmentation, " *CGIP 13*, pp. 222-241, 1980.

[9] P. M. Nerendra, "A Separable Median Filter for Image Noise Smoothing," *IEEE Trans. on PAMI*, Vol. 3, pp. 20-29, 1981.

[10] A. Koschan, "A framework for area-based and feature-based stereo vision," *Machine Graphics & Vision*, Vol. 2, No. 4, pp. 285-308, 1993.

[11] Y. Shirai and Y. Nishimoto, "A Stereo Method Using Disparity Histograms of Multi-Resolution Channels," *Proc. 3rd Int. Symp. on Robotics Research*, Gouvieux, France, pp. 27-32, 1985.

[12] D. Marr and E. Hildreth, "Theory of Edge Detection," *Proc. Royal Soc. of London*, B207, pp. 187-217, 1980.

[13] M. Tremblay, M. Savard, and D. Poussart, "Medium Level Scene Representation Using a VLSI Smart Hexagonal Sensor with Multiresolution Edge Extraction Capability and Scale Space Integration Processing," *Proc. CVPR´94*, Seattle, Washington, USA, pp. 632-637, 1994.

## Appendix

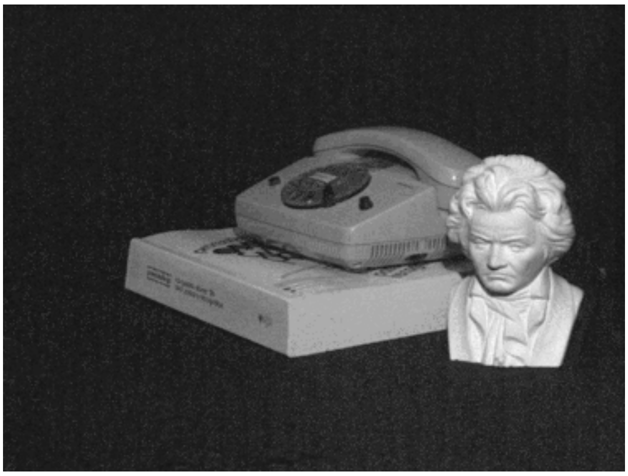

Figure 6: Gray value print of the original left color stereo image BEETHOVEN (752 x 566 pixel in PAL resolution).

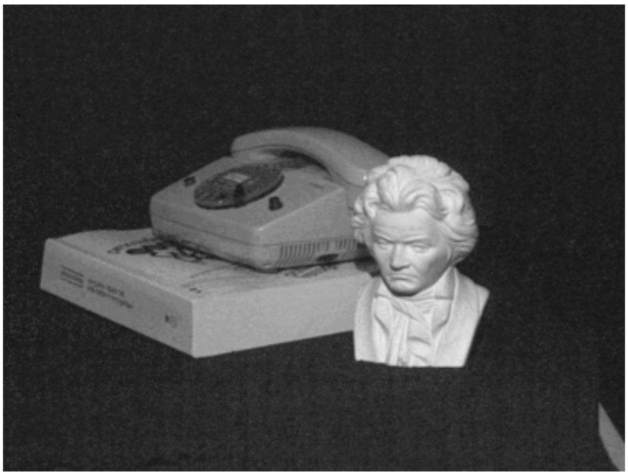

Figure 7: Gray value print of the original right color stereo image BEETHOVEN (752 x 566 pixel in PAL resolution).

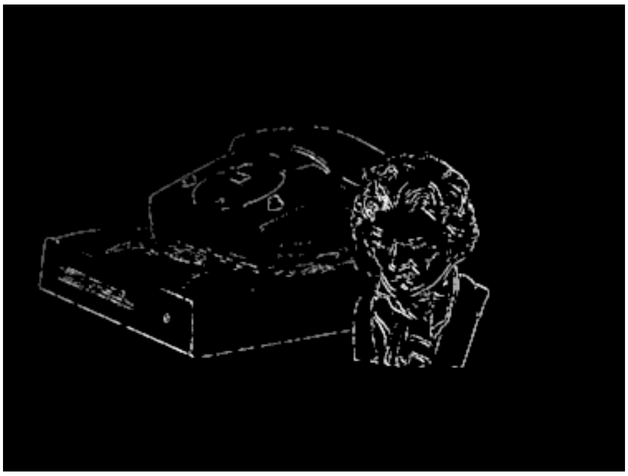

Figure 8: Intensity encoded representation of the depth map obtained with the edge-based approach.

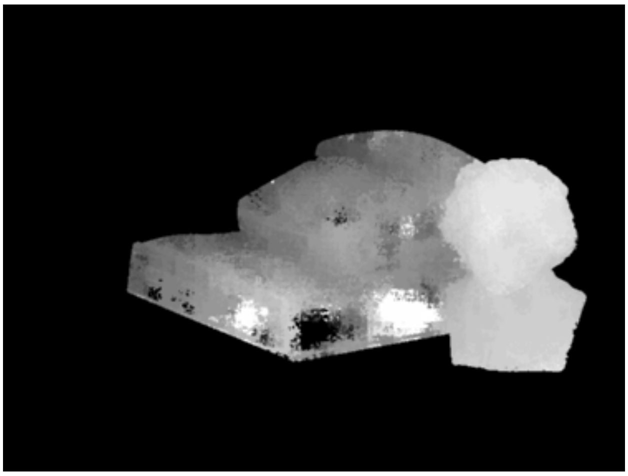

Figure 9: Intensity encoded dense depth map obtained with chromatic Block Matching.

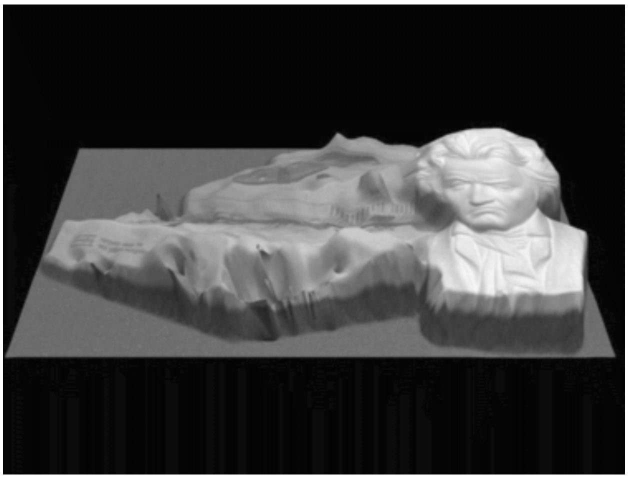

Figure 10: Gray value print of the reconstructed BEETHOVEN scene using the dense depth map shown in Fig. 9.

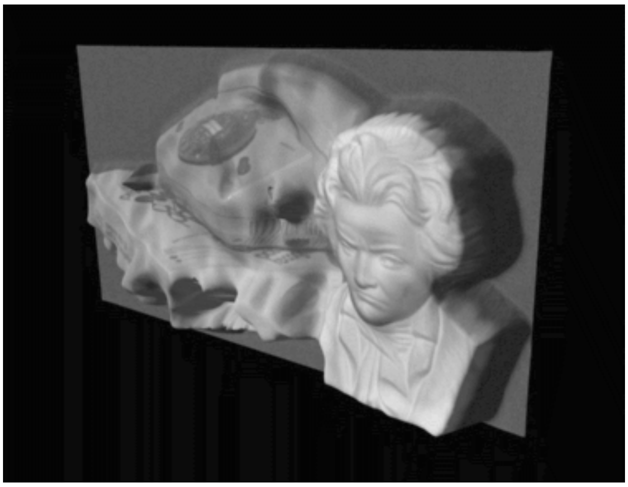

Figure 11: A different view of the reconstructed BEETHOVEN scene using the dense depth map shown in Fig. 9.