Chair for Computer Aided Medical Procedures (I-16)

Lehrstuhl für Informatikanwendungen in der Medizin (XVI)

Diploma thesis – Stereopsis

Subtitle: Realtime calculation of depth maps

Advisor: Prof. Dr. Nassir Navab

Supervision by: Tobias Sielhorst & Jörg Traub

Author: Hauke Heibel

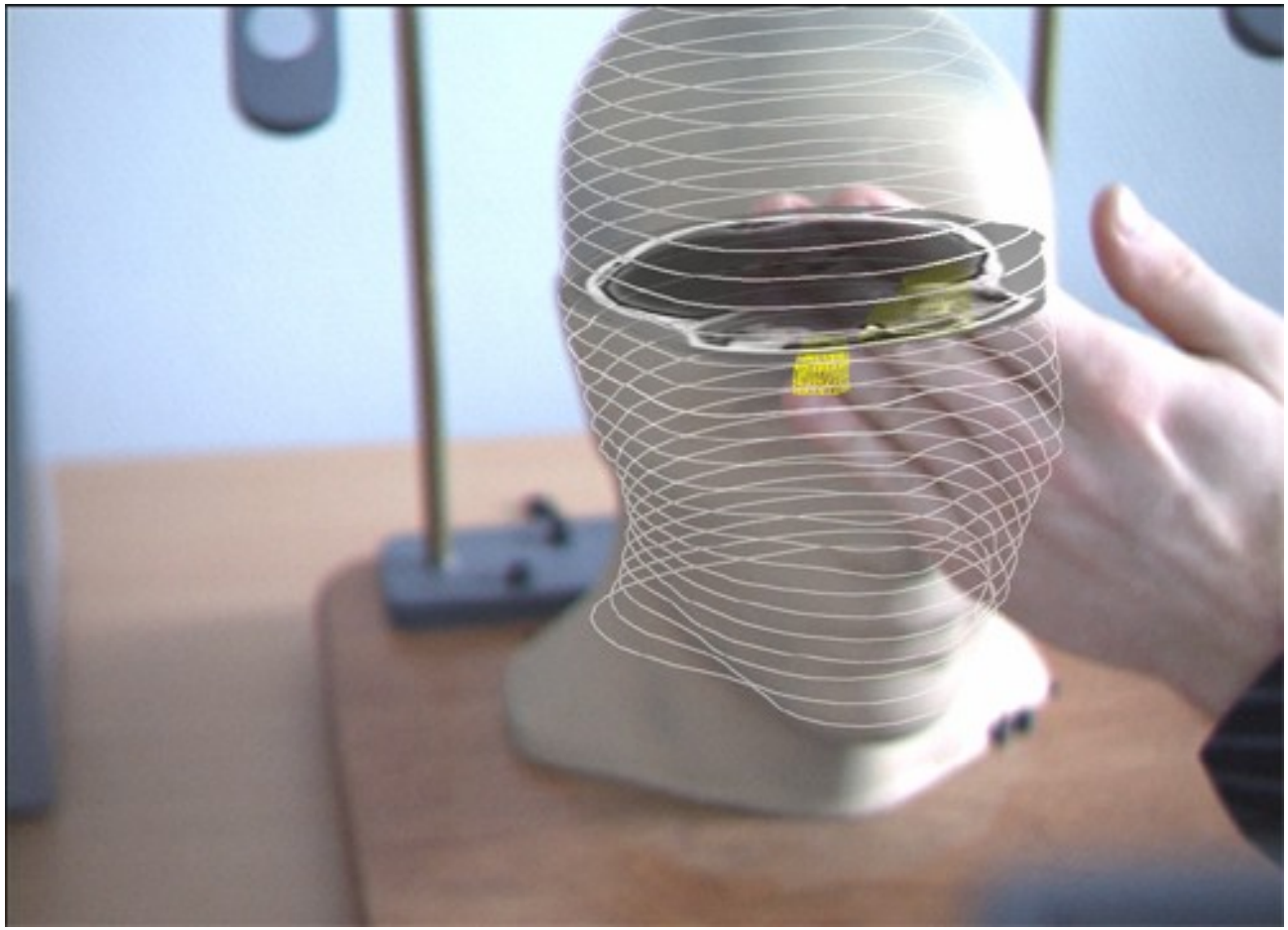
Contents

- Overview: steps for depth map computation
- Motivation: Applications in AR
  - Example „Occlusion handling“
  - Scene reconstruction
- Rectification
  - Definition & Advantages
  - Homographies
- Correspondences
  - Definition & Challenges
  - Choosing the Right Metric
  - FFT Block Matching Algorithm
- Contribution with this Thesis

Steps for depth map computation

1. Calibration of the stereo rig (computation fundamental matrix)
2. Rectification
3. Disparity map calculation
4. Filling the gaps of the disparity map & correction (e.g. occlusion of 2D points in one image ...)
5. Reconstruction phase $\longrightarrow$ Depth map
6. Optional
   - Triangulation on the depth map
   - Texture mapping of the image onto the map

Motivation – Occlusion handling in AR

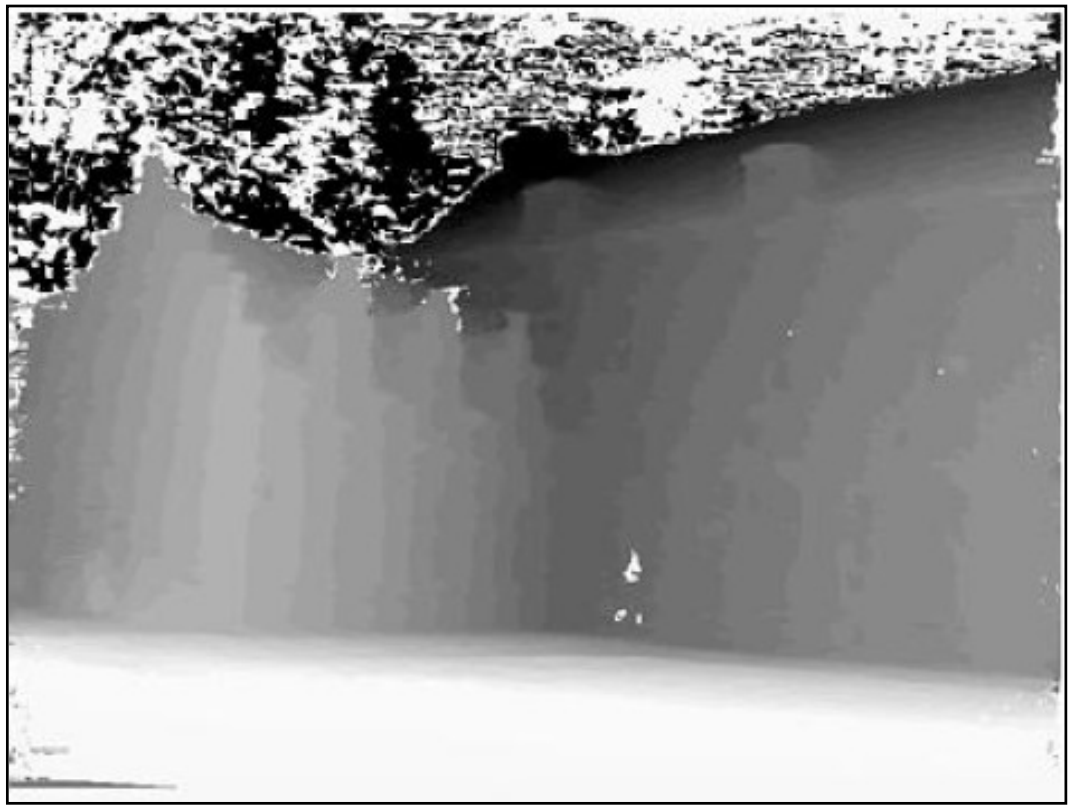


Motivation – Scene Reconstruction

Input

Motivation – Scene Reconstruction

Depth Map



Motivation – Scene Reconstruction

Shaded surface model

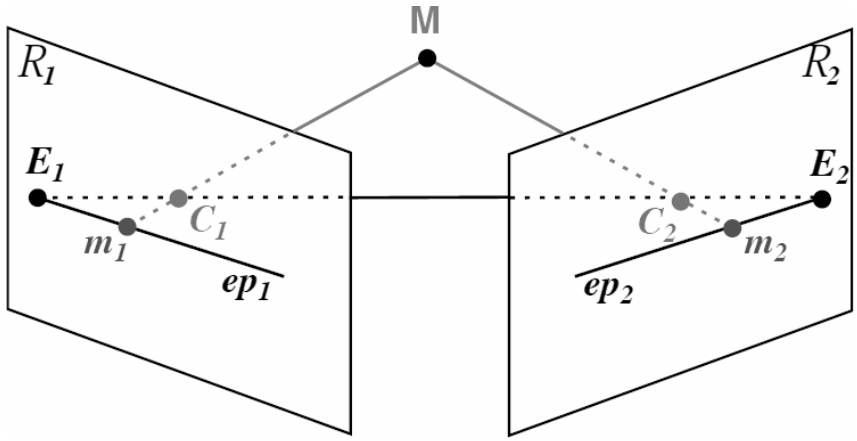
Contents

- Overview: steps for depth map computation
- Motivation: Applications in AR
  - Example „Occlusion handling“
  - Scene reconstruction
- Rectification
  - Definition & Advantages
  - Homographies
- Correspondences
  - Definition & Challenges
  - Choosing the Right Metric
  - FFT Block Matching Algorithm
- Contribution with this Thesis

Rectification – Definition & Advantages

- Corresponding points satisfy epipolar constraint

R_i:	retinal plane
C_i:	optical centers
E_i:	epipoles
M:	physical point
m_i:	point on retinal plane
ep_i:	epipole lines



- Epipolar lines are in general
  - not aligned with coordinate axis
  - not parallel

Rectification – Definition & Advantages

- Time consuming correspondence problem
- Vertically / horizontally align epipolar lines by applying 2D projective transformations or homographies
- Advantage…
  - Rectification reduces two dimensional searches for corresponding points to one dimension

Computing Rectifying Homographies for Stereo …

- Paper by
  - Charles Loop
  - Zhengyou Zhang
- Microsoft Research
- Appeared in Proceedings of ICVP in 1999
- Last corrections in June, 2001

Computing Rectifying Homographies for Stereo …

- Decompose Homographies into
  - projective component
  - affine component
- Determine the projective component
  - minimizes projective distortion criterium
- Decompose the affine component
  - satisfy the constraints of rectification
  - further reduce distortion introduced by the projective component

Computing Rectifying Homographies for Stereo …

- Distortion minimization criteria
  - Find projective component as close to affine as possible
- Prerequisites needed for the rectification
  - Fundamental matrix F must be known
  - The size of the images must be known
    - needed to compute the distortion minimization criteria

Computing Rectifying Homographies for Stereo …

- Formal representation of the decomposition

$$\overline{m} = Hm$$
$$\overline{m}' = H'm' \rightarrow F = H'^{T}[i]_x H$$
$$m'^{T} Fm = 0$$

- Exemplary decomposition of the Homography H

$$H = H_a H_p$$
$$H_a = H_s H_r \rightarrow H = H_s H_r H_p$$

Computing Rectifying Homographies for Stereo …

$$H = \begin{pmatrix} u_a & u_b & u_c \\ v_a & v_b & v_c \\ w_a & w_b & w_c \end{pmatrix}$$ homography

$$H_p = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ w_a & w_b & 1 \end{pmatrix}$$ projective component, maps epipoles at ∞

$$H_a = \begin{pmatrix} u_a - u_c w_a & u_b - u_c w_b & u_c \\ v_a - v_c w_a & v_b - v_c w_b & v_c \\ 0 & 0 & 1 \end{pmatrix}$$ affine component

$$H_r = \begin{pmatrix} v_b - v_c w_b & v_c w_a - v_a & 0 \\ v_a - v_c w_a & v_b - v_c w_b & v_c \\ 0 & 0 & 1 \end{pmatrix}$$ similarity transformation, rotates points into axis alignment with i

a translation v_c is found which aligns the scan-lines

$$H_s = \begin{pmatrix} s_a & s_b & s_c \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$ minimize distortion introduced by H_p

i $[1\ 0\ 0]^T$, a point at ∞

How to use the algorithm and the advantages

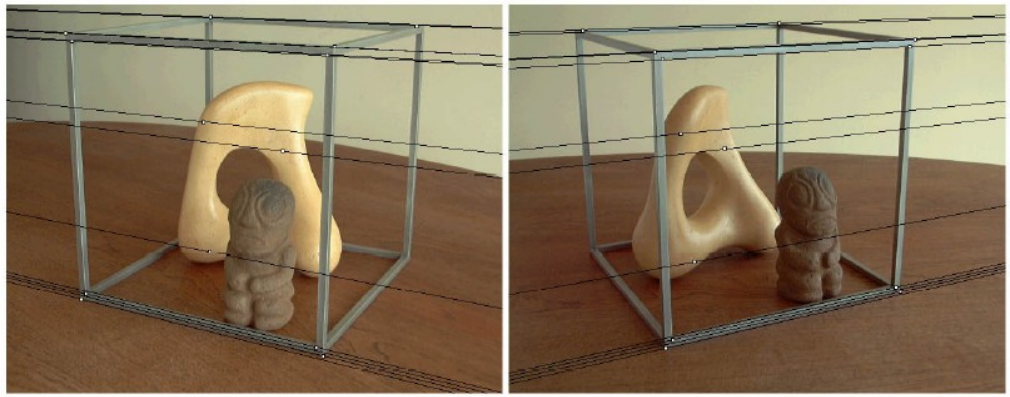
- Compute the scanlines once
  - the size of the images and F are constant
- For each pixel in the rectified images, store
  - the corresponding pixel in the left and right image

$$(x_{l\_rect}, y_{l\_rect}) \rightarrow (x_l, y_l)$$

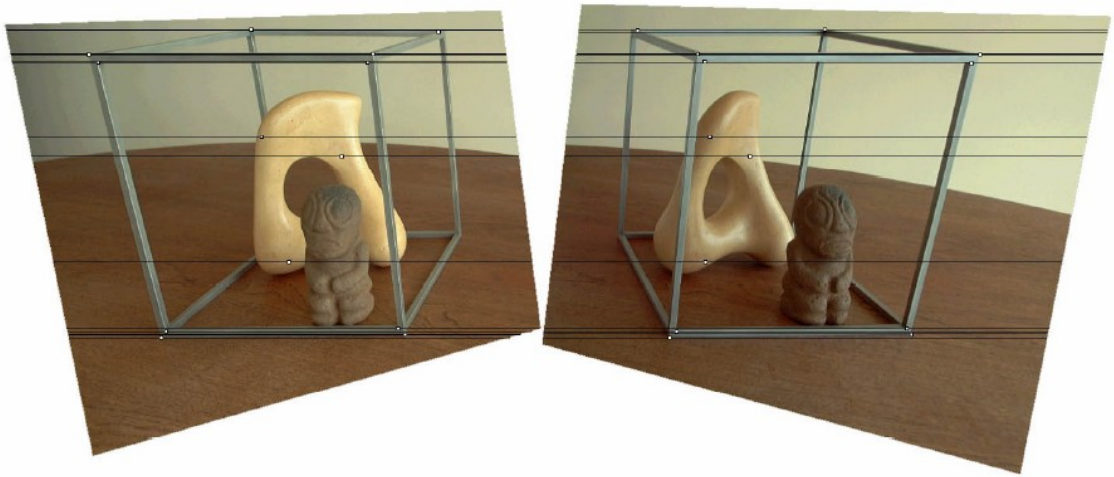
$$(x_{r\_rect}, y_{r\_rect}) \rightarrow (x_r, y_r)$$

- Corresponding pixels can be accessed through look-up tables

Example for the rectification process



Original images



After the rectification

Contents

- Overview: steps for depth map computation
- Motivation: Applications in AR
  - Example „Occlusion handling“
  - Scene reconstruction
- Rectification
  - Definition & Advantages
  - Homographies
- Correspondences
  - Definition & Challenges
  - Choosing the Right Metric
  - FFT Block Matching Algorithm
- Contribution with this Thesis

Correspondences – Definition & Challenges

- Correspondence problem

> *„Given a token in image 1, what is the corresponding token in image 2.“*
>
> (Three Dimensional Computer Vision, Oliver Faugeras)

- Problems
  - Does such a token exist?
  - Homogenous areas

Correspondences – Choosing the Right Metric

- Block matching
  - minimizing a similarity criteria between a template block and candidate blocks
- Metrics
  - Sum of absolute differences (SAD)
  - Sum of squared differences (SSD)
- Factors influencing the choice of the metric
  - Speed (SAD is faster than SSD)
  - Is a maximum peak-signal-to-noise-ratio desired

Correspondences – FFT Block Matching Algorithm

- Paper by
  - Steven L. Kilthau
  - Mark S. Drew
  - Torsten Möller
- Simon Frasier University
- Home page: http://www.cs.sfu.ca/~mark/ftp/Icip02/index.htm

Correspondences – FFT Block Matching Algorithm

Based on the SSD metric

$$\min_{\forall u,v} SSD(u,v) = \min_{\forall u,v} \sum_{j=0}^{B-1}\sum_{i=0}^{B-1} (f_l(i,j) - f_r(i+u, j+v))^2$$

$$= \min_{\forall u,v} \sum_{j=0}^{B-1}\sum_{i=0}^{B-1} \begin{bmatrix} f_l(i,j) + f_r(i+u, j+v)^2 \\ -2 f_l(i,j) f_r(i+u, j+v) \end{bmatrix}$$

The term $f_l(i,j)$ can be ignored

$$\min_{\forall u,v} \left(\begin{array}{l} \sum_{j=0}^{B-1}\sum_{i=0}^{B-1} f_r(i+u, j+v)^2 \\ -2\sum_{j=0}^{B-1}\sum_{i=0}^{B-1} f_l(i,j) f_r(i+u, j+v) \end{array} \right)$$

Correspondences – FFT Block Matching Algorithm

- The first term can be computed with dynamic programming

$$f_{SST}(i,j) = \begin{bmatrix} f(i,j)^2 + f_{SST}(i-1,j) + \\ f_{SST}(i,j-1) - f_{SST}(i-1,j-1) \end{bmatrix}$$

$$f_{WSST}(i,j) = \sum_{s=j-B+1}^{j} \sum_{t=i.B+1}^{i} f(s,t)^2$$

$$= [f_{SST}(i,j)]$$
$$+ [f_{SST}(-1,j) - f_{SST}(i-B,j)]$$
$$+ [f_{SST}(i,-1) - f_{SST}(i,j-B)]$$
$$+ [f_{SST}(-1,-1) + f_{SST}(i-B,j-B)]$$
$$- f_{SST}(i-B,-1) - f_{SST}(-1,j-B)$$

Correspondences – FFT Block Matching Algorithm

- The second term can be computed with FFT
  - Dot product for single template block
  - Convolution for all $(2P+1)^2$ candidate blocks
    - correlation of template block with (2P+B)x(2P+B) region
  - Convert this correlation to convolution
  - Finally use the FFT
- Outperforms exhaustive search SAD and SSD algorithms by 10% - 30%

Experiences with the FFT Block Matching Algorithm

All timings include reading the images, constructing the necessary data structures, and computing all motion vectors for each frame of the input image sequence.
Correspondence window size of 16 pixels.

P	1600 MHz P4 (756MB RAM)	1600 MHz Centrino (1GB RAM)
+/-8	Full search took 21.320 seconds for 80 frames. FFT search took 5.298 seconds for 80 frames. There were 3 errors over 80 frames.	Full search took 10.237 seconds for 80 frames. FFT search took 2.953 seconds for 80 frames. There were 3 errors over 80 frames.
+/-16	Full search took 75.789 seconds for 80 frames. FFT search took 13.488 seconds for 80 frames. There were 2 errors over 80 frames.	Full search took 35.881 seconds for 80 frames. FFT search took 8.255 seconds for 80 frames. There were 2 errors over 80 frames.
+/-24	Full search took 163.214 seconds for 80 frames. FFT search took 22.960 seconds for 80 frames. There were 0 errors over 80 frames.	Full search took 78.689 seconds for 80 frames. FFT search took 12.866 seconds for 80 frames. There were 0 errors over 80 frames.
+/-32	Full search took 315.545 seconds for 80 frames. FFT search took 44.753 seconds for 80 frames. There were 0 errors over 80 frames.	Full search took 138.128 seconds for 80 frames. FFT search took 26.690 seconds for 80 frames. There were 0 errors over 80 frames.
+/-8	Full search took 7.299 seconds for 100 frames. FFT search took 2.000 seconds for 100 frames. There were 39 errors over 100 frames.	Full search took 3.495 seconds for 100 frames. FFT search took 0.986 seconds for 100 frames. There were 39 errors over 100 frames.
+/-16	Full search took 26.696 seconds for 100 frames. FFT search took 4.961 seconds for 100 frames. There were 47 errors over 100 frames.	Full search took 12.996 seconds for 100 frames. FFT search took 2.982 seconds for 100 frames. There were 47 errors over 100 frames.
+/-24	Full search took 58.511 seconds for 100 frames. FFT search took 8.204 seconds for 100 frames. There were 36 errors over 100 frames.	Full search took 28.493 seconds for 100 frames. FFT search took 4.643 seconds for 100 frames. There were 36 errors over 100 frames.
+/-32	Full search took 116.697 seconds for 100 frames. FFT search took 16.191 seconds for 100 frames. There were 46 errors over 100 frames.	Full search took 49.885 seconds for 100 frames. FFT search took 9.652 seconds for 100 frames. There were 46 errors over 100 frames.

Contents

- Overview: steps for depth map computation
- Motivation: Applications in AR
  - Example „Occlusion handling“
  - Scene reconstruction
- Rectification
  - Definition & Advantages
  - Homographies
- Correspondences
  - Definition & Challenges
  - Choosing the Right Metric
  - FFT Block Matching Algorithm
- Contribution with this Thesis

Contribution with this Thesis

- Goals
  - desired frame rate
    - 20fps to 30fps
  - resolution of input images
    - 640x480 up to 1280x1024
- Roadmap
  - Implementation of a fast rectification algorithm
  - Adaption of the FFT Block Matching algorithm
  - Implementation of reconstruction algorithm