Rendering Real-World Objects without 3-D Model

Tomáš Werner
Computer Vision Laboratory
Department of Control Engineering
Faculty of Electrical Engineering
Czech Technical University
Karlovo náměstí 13, Prague 12135, Czech Republic
e-mail werner@vision.felk.cvut.cz
phone ++42 2 24357458, fax ++42 2 290159

CTU Technical Report No. K335-95-92

Acknowledgements

I would like to thank all my colleagues that provided me with advises and help during my postgradual studies — namely PhD. Ing. Radim Šára, Ing. Vladimír Smutný, Ing. Tomáš Pajdla and Ing. Michal Meloun. Most of all, I thank my advisor Doc. Ing. Václav Hlaváč, CSc. for his help with choosing the topic of my research and his constant scientific and organizational support. Also, I would like to thank Prof. Roger Hersch from the EPFL, Lausanne for the financial support of my research.

This research is supported (1) by the Peripheral Devices Laboratory, École Polytechnique Fédérale de Lausanne, Switzerland, (2) by the grant No. 102/93/0954 of the Grant Agency of the Czech Republic,(3) by the grant of the Czech Technical University No. 38196.

Contents

Chapter 1

Introduction

1.1 The Problem Specification

This report describes a new approach to rendering real-world objects, captured by a capturing device, viewed from an arbitrary viewpoint. That means, at the very end of the way to which my research is directed there is a device which is able:

1. To capture a real-world object whose geometric and photometric properties are not restricted in any way except being a solid body.

2. To store the data obtained in this process with as low redundancy as possible.

3. Using this data, to recover any view on the object from an arbitrarily chosen viewpoint outside the object.

While the rendering of 3-D objects has been successfully solved in computer graphics (though it is still a rapidly developing area), the described task is much more difficult due to the requirement of handling *real-world* objects. This is also the reason why the work has more to do with computer vision that with computer graphics.

All approaches known to me use the 3-D model of the object to solve the above task. Thus, the whole problem is in fact reduced to *3-D model reconstruction*. The real-world object model reconstruction is an extremely difficult problem and it still remains unsolved for objects of complex geometry or topology.

In this report it will be shown that the 3-D model reconstruction is not necessary, provided that only rendering the object is required. The new object representation is proposed: *a set of primary views* along with information about their mutual *correspondence*. The primary views can be accessed directly and any view that is not the primary one can be obtained by interpolating among close primary views, using information about their correspondence.

The bottleneck of this new approach is the *correspondence problem*. As the correspondence problem is much more simple than 3-D model reconstruction, the approach gives a chance to handle objects of complicated shapes.

During my postgradual study, I intend to develop a method which is able to render a real-world object, captured by a monocular camera, using this new approach. The final objective of this work of mine is more modest than the general one described above. It differs in the following:

1. Objects are opaque.

2. Objects have such size so that they can be placed on a calibrated rotation table.

3. The set of allowed viewpoints from which objects are rendered is restricted. Changing the viewpoint is equivalent to an arbitrary rotation of the object in front of a still camera at a constant distance.

4. The loss of some minor details or a slight distortion of objects' shape in the interpolated views are allowed.

Further in this report, theoretical results concerning view interpolation, as well as results of encouraging experiments are presented.

1.2 The Proposed Approach

In this section, we outline how to render a 3-D object from an arbitrary viewpoint without having its model. A set of views covering the whole visible surface of the object is captured. These views can be accessed directly; all we need to do is display them. What is demanded are the intermediate views. If information about primary views correspondence is available, it is possible to obtain any interpolated view using a subset of the primary views close to it. This allows us to avoid the difficult fusion of the views that is necessary for the 3-D model reconstruction.

To succeed, the following problems must be solved:

1. How to determine the position and intensity of a point in the interpolated view if positions and intensities of corresponding points in the primary views are known?
2. How to determine the visibility of points in the interpolated view?
3. How to find the possibly smallest set of necessary primary views?
4. How to find correspondences between primary views?

In the following sections, we give answers to some of these questions. We show that position and intensity of a point in the interpolated view can be up to a certain error expressed as a linear combination of positions and intensities of corresponding points in primary views. We expect that this error can be kept small for the primary and views close to each other and to the interpolated view.

We show that the visibility problem for the interpolated view can be solved. Yet, any point of the object's surface must be visible at least from a certain number of its primary views.

The problem of the choice of the smallest and still sufficient set of the primary views is non-trivial. The more restricted fundamental problem of how to choose a set of so-called *characteristic views* (i.e., the minimum set of views in which all points of a given surface are visible) still remains unsolved for general non-convex objects. In our case the primary views set would have to fulfill additional requirements, e.g., a reasonable sampling frequency for all points on the surface. We do not deal with question 3 in this paper.

We describe the algorithm for correspondence acquisition. It is a modification of an already published algorithm. Yet, we have further improved this algorithm and developed the necessary calibration procedure.

1.3 Previous Work

The most important result of our analysis of the state of the art is that we have found that no information has been published thus far about a system which is able to capture and render 3-D objects by view interpolation. Further, we will describe some most relevant works.

The work of Ullman and Basri [UB89] is inspiring, as it describes an approach to 3-D object recognition using a set of views on an object instead of its model. It is shown that any instance of the object can be expressed as a linear combination of these views assuming orthogonal projection. A similar and even earlier work is presented in [CF82].

Faugeras and Robert [FR93] deal with a prediction of coordinates, tangents and curvatures in a view on the basis of knowing other views. The cameras need to be only weakly calibrated.

In the work of Gudmundsson and Randen [GR91], the view interpolation is used to speed up the generation of ray-traced sequences of objects rotating in 3-D space. They achieve speed-up factors of 10–15 compared to the full rendering.

Skerjanc and Liu in [SL91, SL92] want to use intermediate views to achieve an effect of 3-D TV (the viewpoint changes according to the position of the viewer). Only the two extreme views are to be transmitted and the interpolated ones obtained by interpolation. In [ZL93], an object is scanned along a pre-defined path, the correspondence is found, several views are selected, and an arbitrary interpolated view is constructed.

Chapter 2

Solution

2.1 View Interpolation

Let us assume a sufficient set of primary views and information about their correspondence are available. In this section we show how interpolated views can be constructed, using a subset of the primary views close to each other.

2.1.1 Position and Intensity of the Interpolated Point

Let us assume n corresponding points in n primary views, i.e., $2n$-tuple $[\mathbf{x}_1, I_1, \ldots, \mathbf{x}_n, I_n]$, where $\mathbf{x}_i$ are image coordinates of i-th point and I_i, its intensity. The i-th point

is a projection of some point $\mathbf{X}$ on the object's surface in the i-th view with view parameters[1] $\mathbf{p}_i$. The goal is to obtain the coordinates $\mathbf{x}$ and intensity I of the projection of $\mathbf{X}$ in a interpolated view with parameters $\mathbf{p}$.

If all the views are calibrated, the following system of equations is usually to be solved:

$$\begin{aligned} \mathbf{x}_i &= f(\mathbf{X}, \mathbf{p}_i),\ i = 1, \ldots, n \\ \mathbf{x} &= f(\mathbf{X}, \mathbf{p}), \end{aligned} \tag{2.1}$$

where f is a function assigning its projection to a point in 3-D space (f and $\mathbf{p}_i$ are known from the calibration). If $n = 2$ the system has a unique solution, if $n \geq 2$ the solution can be obtained by a least squares fit. The case of $n = 2$ is equal to finding all three coordinates of $\mathbf{X}$ by triangulization, followed by a projecting to the virtual camera coordinate system.

We show that this approach can be simplified. Let us express the parameters of the interpolated view $\mathbf{p}$ as a linear combination of parameters of n primary views, $\mathbf{p} = \sum \alpha_i \mathbf{p}_i$. If $\sum \alpha_i = 1$ and $\mathbf{p}_i$ are close enough to each other, it holds that:

$$f(\mathbf{X}, \sum_{i=1}^{n} \alpha_i \mathbf{p}_i) \approx \sum_{i=1}^{n} \alpha_i f(\mathbf{X}, \mathbf{p}_i), \tag{2.2}$$

i.e.,

$$\mathbf{x} = \sum_{i=1}^{n} \alpha_i \mathbf{x}_i. \tag{2.3}$$

The proof can be made by substituting $\mathbf{p}_i = \mathbf{q}_0 + \Delta \mathbf{q}_i$, $\mathbf{p} = \mathbf{q}_0 + \sum \alpha_i \Delta \mathbf{q}_i$, and by expanding the function $\mathbf{g}(\mathbf{p}) = f(\mathbf{X}, \mathbf{p})$ to its Taylor series along the point $\mathbf{q}_0$, neglecting the terms higher than first order.

[1] We consider ideal perspective projection acting in image formation. *View parameters* do not mean a complete set of camera parameters, but rather a point in a variety of allowed views. E.g., if an object is viewed by a camera moving on a spherical surface and looking to the center of this sphere, the view parameters are three angles.

The same procedure can be used for interpolation of image intensities (or values of color components). If we formally replace f with reflectance (or color) model, we obtain $I = \sum \alpha_i I_i$.

The numbers α_i can be obtained by solving the equations:

$$\begin{aligned} \sum_{i=1}^{n} \alpha_i \mathbf{p}_i &= \mathbf{p},\ i = 1, \ldots, n \\ \sum_{i=1}^{n} \alpha_i &= 1 \end{aligned} \tag{2.4}$$

The (2.4) has a solution for $n \geq dim(\mathbf{p}) + 1$. It implies that we need at least $dim(\mathbf{p}) + 1$ primary views to determine the interpolated one, without any camera calibration.

It is reasonable to call the construction of a new view an *interpolation*, if for all i holds $0 \leq \alpha_i \leq 1$), and an *extrapolation*, if the converse is true. We expect the error due to the linearization to be smaller for the interpolation.

2.1.2 Visibility of Interpolated Points

Let us make a restriction to opaque objects (we will not deal with transparent objects in this paper). The interpolated view can be constructed as a set of combinations $\sum \alpha_i f(\mathbf{X}_k, \mathbf{p}_i)$ of corresponding projections of the object's surface points $\mathbf{X}_k$ from a given subset of the whole primary view set. Let us denote p_i and p the projections of a constructed point $\mathbf{X}_k$ in views with parameters $\mathbf{p}_i$ and $\mathbf{p}$, respectively. Let us now distinguish six cases (see the fig. VisibilityCasesFig) that can arise for the visibility of $\mathbf{X}_k$ in particular views and discuss the way they affect the interpolated view construction:

1. All p_i are visible, p is visible.
   The interpolated point p can be constructed from the given subset and it is visible. This case should occur for as many interpolated points as possible.

2. All p_i are visible, p is invisible.
   The point $\mathbf{X}_k$ is occluded in the interpolated view, but interpolated point p can be constructed from the given subset. This case occurs if for some other n-tuple $f(\mathbf{X}_l, \mathbf{p}_i), l \neq k$, the equality $\sum \alpha_i f(\mathbf{X}_l, \mathbf{p}_i) = \sum \alpha_i f(\mathbf{X}_k, \mathbf{p}_i)$ holds[2] (and it can be also detected like this). It is necessary to decide which of the two points $\mathbf{X}_k$, $\mathbf{X}_l$ is visible. This decision can be made using their disparity, because disparity is a descending function of depth. The suitable data structure for the implementation is a z-buffer — the buffered value can be the disparity of the interpolated points, referred to a chosen primary view.

3. Some p_i are visible and some invisible, p is visible.
   The point $\mathbf{X}_k$ is occluded in one of the primary views. Therefore the n-tuple $f(\mathbf{X}_k, \mathbf{p}_i)$ does not exist, if a real device was used to capture it, and the interpolation at the point p cannot be computed. The areas where these situations occur remain empty[3] in the constructed view. The missing points can be obtained by an extrapolation of a different (e.g. neighboring) subset of primary views, provided that they are not occluded in them.

4. Some p_i are visible and some invisible, p is invisible.
   The point $\mathbf{X}_k$ is not visible from the interpolated view and it is not constructed.

[2] This occurs if and only if the so-called ordering constraint is violated in some pair of the primary views. Therefore, a matching algorithm utilizing this constraint cannot be used for acquiring correspondence. Unfortunately, the majority of them does so.

[3] These areas must be distinguished from the holes caused by unequal sampling frequencies in the primary and interpolated views.

5. All p_i are invisible, p is visible.
   The point $\mathbf{X}_k$ is visible in the interpolated view but occluded in all primary views in the subset. If it is occluded in all other primary views (which is likely to occur if p is constructed by view interpolation), there is no information about it. This case is undesirable and it should be prevented by choosing the optimal set of primary views. It can be detected the same way as case 2.

6. All p_i are invisible, p is invisible.
   This is the same case as 4.

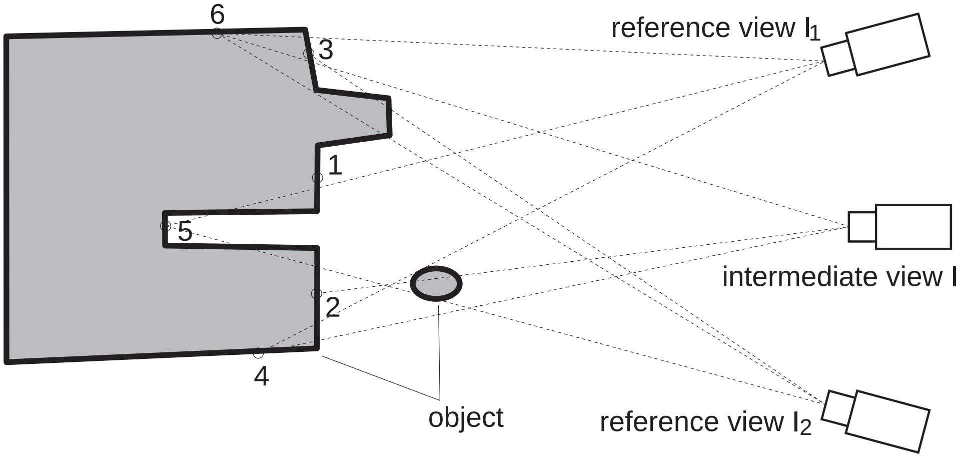

Figure 2.1: The six cases which can arise for the visibility of points in the interpolated view. The numbers correspond to the notation in text.

We conclude that during the construction of the interpolated view, it is necessary (and possible) to detect the cases 2, 3 and 5. The correct handling of the cases 2 and 3 can be ensured, while the case 5 must be prevented in advance by the proper choice of the set of primary views.

2.2 Correspondence Acquisition

We need to find correspondence for the subsets of the primary views from which the interpolated views will be constructed. In other words, we are looking for all n-tuples $[\mathbf{x}_1, \ldots, \mathbf{x}_n]$ of pixels coordinates (each from a different view) so that every n-tuple contains projections of a single point on the object's surface, up to errors caused by discretization and noise. This *correspondence problem* is difficult mainly due to: (1) the change of geometric and photometric properties with the viewpoint, (2) the presence of noise, (3) discretization, and (4) the lack of information in areas of constant intensity.

Rendering by view combination brings an important advantage in comparison to the approaches using a 3-D model: low sensitivity to correspondence errors in areas of (almost) constant intensity. The areas in which these errors show up in the interpolated view will also have a constant intensity, so the image will *look* the same. Even if the visibility of the interpolated pixel is inferred from disparities (which is quite a rare situation), only minor artifacts can be expected. This insensitivity allows us to use passive methods for acquiring correspondence, having the potential to capture both indoor and outdoor objects, and the possibility to provide directly dense correspondence.

2.2.1 Matching Algorithm

There are two alternatives for determining correspondence of n views: (1) n-ocular stereo, and (2) composition of pairwise correspondences[4]. We chose the alternative (2). For that, we need an algorithm matching two near views on a single object.

The algorithm we use is inspired by the binocular stereo matching algorithm [CHMR92]. It provides dense correspondence, as it is intensity-based. It is similar to the more known algorithm [OK85], but rather than intervals between edges, the raw pixel intensities are matched. Epipolar, unicity, and ordering constraints are utilized. For each pair of corresponding epipolars, such a pair of non-descending transformation functions is sought, that transforms the intensity functions so that the sum of costs of either matches or occlusions is minimized. This optimization problem is solved by dynamic programming. The cost of a match and occlusion is derived — the problem being formulated as a Bayesian sensor fusion. The advantage of the algorithm is that it directly produces dense correspondence.

Further in this section, we describe the matching algorithm in more detail, along with the necessary background.

Epipolar Constraint

The character of the matching algorithm is strongly influenced by the fact that the *epipolar constraint* is utilized. This constraint is important as it *a-priori* reduces the search space in which a pixel $\mathbf{m}_2$ in the image 2, corresponding to the pixel $\mathbf{m}_1$ in the image 1 is searched. In the fig. 2.2, $\mathbf{m}_1$ and $\mathbf{m}_2$ are the projections of a point $\mathbf{X}$ lying on an object's surface. The camera systems of the views have centers of projections C_1, C_2 and projection planes P_1, P_2. For a fixed $\mathbf{m}_1$, the point corresponding to it in the image 2 can be located only on the line $\mathbf{l}_2$, called *epipolar line*. The symmetrical assertion holds for $\mathbf{m}_2$ and $\mathbf{l}_1$. Moreover, any point on $\mathbf{l}_1$ can correspond only to a point located on $\mathbf{l}_2$ and vice versa. This implies the important conclusion: the image pair can be matched by matching proper epipolar line pairs separately.

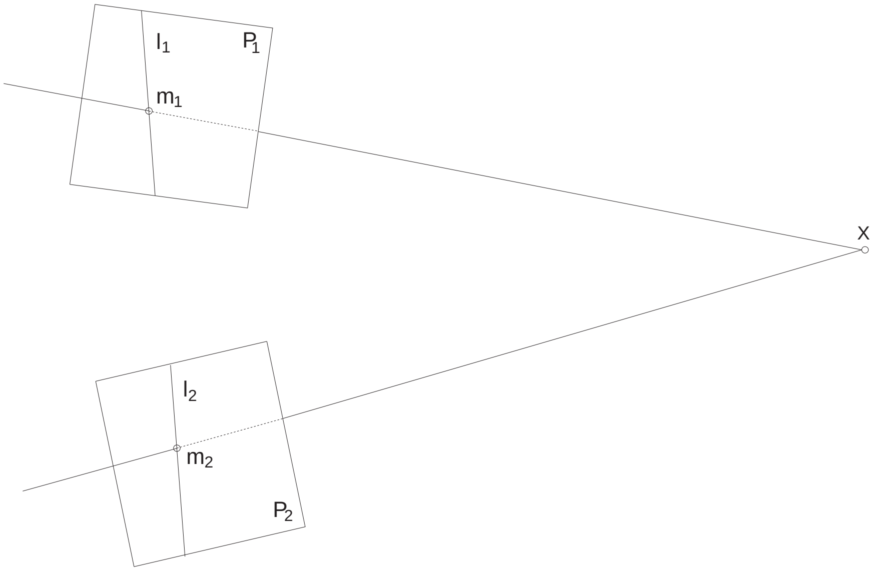

Figure 2.2: The epipolar geometry of the matched image pair.

Epipolar lines $\mathbf{l}_1$ and $\mathbf{l}_2$ are the intersections of the plane C_1C_2X and projection planes P_1 and P_2, respectively.

[4] The accumulation error limits the number of compositions. This error arises by accumulating errors in composed pairwise correspondences, always present due to noise.

Expressed in terms of projective geometry, a simple linear relation exists between a point and its epipolar line:

$$\mathbf{l}_2 = \mathbf{F}_{12}\mathbf{m}_1,\ \mathbf{l}_1 = \mathbf{F}_{21}\mathbf{m}_2 \tag{2.5}$$

(lines and points in homogeneous coordinates). $\mathbf{F}_{12}$ (or $\mathbf{F}_{21}$) is the *fundamental matrix* [FR93, Fau93]. It is singular and determined up to a scale factor — i.e., it has seven independent elements. It also can be shown that $\mathbf{F}_{21} = \mathbf{F}_{12}^T$. Given $\mathbf{F}_{12}$ (or $\mathbf{F}_{21}$), the complete epipolar geometry is known and the view pair is said to be *weakly calibrated.*

Matching algorithms utilizing the epipolar constraint usually match *rectified images* in which epipolars are parallel to scanlines. The rectified image can be obtained by a proper transformation of the original one. It can be shown that this transformation is projective, i.e., linear in projective space[5]:

$$\mathbf{m}' = \mathbf{A}\mathbf{m}, \tag{2.6}$$

where $\mathbf{m}'$ and $\mathbf{m}$ are the coordinates of points in the rectified image and the original image, respectively. $\mathbf{A}$ can be calculated using four pairs $[\mathbf{m}'_i, \mathbf{m}_i]$, $i = 1, \ldots, 4$, of original and transformed points (on condition that no point triple is collinear). After finding the correspondence, the coordinates of matched pixels are transformed back by the inverse transform.

Our solution to weak calibration is described in the section 2.2.2.

The Dynamic Programming Solution

We have shown that the matching of an image pair can be done by matching proper epipolar line pairs separately, and how to obtain rectified images that have epipolar lines parallel to their scanlines. Now we will show how to match an epipolar line pair.

Let us make the following assumptions about correspondence of pixels in the epipolar line pair:

1. *Uniqueness.* A pixel on an epipolar line corresponds to at most one pixel on the other epipolar line.

2. *Monotonicity* (or *ordering constraint*). The order of arbitrarily chosen pixels on an epipolar line is the same as the order of corresponding pixels on the other epipolar.

3. *Photommetric constraint.* Intensities of the corresponding pixels are the same or differ only slightly.

Although these assumptions are violated in some situations[6], they simplify the matching considerably. That is the reason why they are frequently utilized in matching algorithms.

Let us assume that the points on the two epipolar lines have respectively indices $1, \ldots, M$ and $1, \ldots, N$. Any correspondence of the epipolar line pair can be conveniently depicted as a *continuous path* in the system $l_1 l_2$ (see the fig. 2.3). If the correspondence meets the uniqueness and monotonicity assumptions, the path can be always *non-descending*, and it is determined uniquely by this requirement. It starts at the origin $O = [0, 0]$ and it ends at the node $[M, N]$. The path segment connecting the nodes $[i-1, j-1]$ and $[i, j]$ means that the pixel pair $[i, j]$ is matched. The path segment connecting the nodes $[i-1, j]$ and $[i, j]$ or the nodes $[i, j-1]$ and $[i, j]$ means that one pixel of the pair $[i, j]$ is occluded.

[5] This is due to the fact that all epipolars in the original image intersect in a single point called *epipole*. The original image can be approached as a perspective projection of the rectified one, having the epipole as the vanishing point of the pencil of scanlines in the rectified image.

[6] Uniqueness can be violated if the object is transparent. Correspondence problem for transparent objects is much more complicated — that is one of the reasond why we consider only opaque objects in this work. Monotonicity is violated (1) for transparent objects and (2) if some thin part of the object occludes other parts. (2) is is quite a rare situation. Photommetric constraint is violated if reflectivity of the object's surface is near to specular.

Let us assign *costs* to all path segments. The cost of the path segment connecting the nodes $[i-1, j-1]$ and $[i, j]$ is $c(i, j)$. $c(i, j)$ represents the measure of dissimilarity between the pixel intensities. The cost of the path segment connecting the nodes $[i-1, j]$ and $[i, j]$ or the nodes $[i, j-1]$ and $[i, j]$ is D. It is the penalty for occlusion. The cost of the whole path is the sum of costs of its segments. The path with a minimum cost (minimum path) represents the desired correspondence of the epipolar pair.

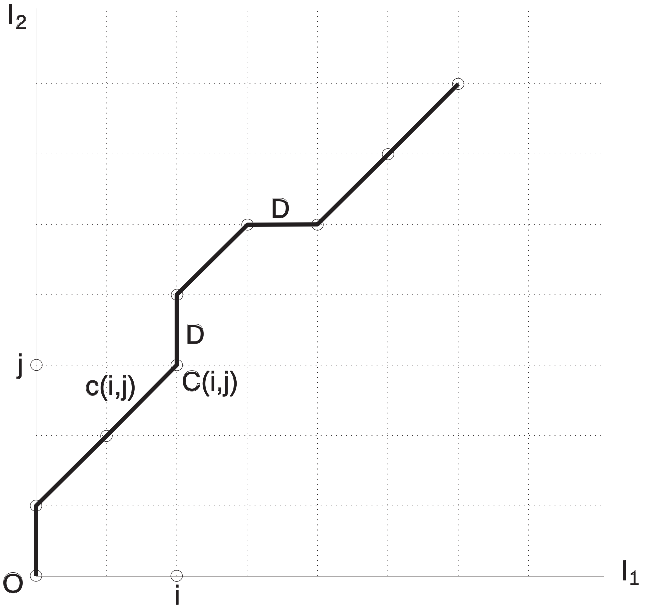

Figure 2.3: The path representing correspondence of the epipolar line pair.

Let us assign to every node $[i, j]$ the cost $C(i, j)$ of the minimum path starting at the origin $O = [0, 0]$ and ending at this node $[i, j]$. As the path is non-descending it holds:

$$C(i, j) = \min\{c(i, j) + C(i-1, j-1), D + C(i, j-1), D + C(i-1, j)\}. \tag{2.7}$$

The values $C(i, j)$ for $i = 0$ or $j = 0$ can be directly computed: $C(i, 0) = iD, C(0, j) = jD$. If we proceed from smaller indices to larger ones, every time when the (2.7) is evaluated the values $C(i-1, j-1), C(i, j-1), C(i-1, j)$ in the right-hand side of the expression (2.7) have been already computed. If we store $C(i, j)$, we always need to compute only $c(i, j)$. Thus, all $C(i, j)$ can be determined with the complexity $O(MN)$. This technique is known as *dynamic programming*. The minimum path can be found by back-tracking the recursion process by which $C(M, N)$ was computed.

In the original paper [CHMR92], the derivation of $c(i, j)$ and D is given, the problem being formulated as a Bayesian sensor fusion. On condition that the pixel intensities are normally distributed random variables with equal dispersions, it holds:

$$c(i, j) = a[I_1(i) - I_2(j)]^2, \tag{2.8}$$

where $I_1(i)$ and $I_2(j)$ are mean values of intensities of the pixels i, j, respectively, and a is a constant. The constant a as well as the cost of occlusion D can be determined if (1) the dispersion of the intensity distribution and (2) the probability that a pixel on an epipolar line is occluded on the other epipolar line, are known *a priori*[7]. Yet, the sensitivity to the correct choice of a and D is not critical.

[7] For details, see the original paper.

We have observed that the algorithm yields more accurate results if the maximized quantity is a sum of correlations of windows having the matched pixels as their centers, rather than costs of matches of the pixels alone[8]. Thus, we have improved the original algorithm by combining it with a correlation method.

2.2.2 Calibration

As the epipolar constraint is utilized, the two views must be *weakly calibrated*, e.g., the fundamental matrix $\mathbf{F}_{12}$ found.

Determining Fundamental Matrix

The fundamental matrix can be determined from a set $\{[\mathbf{m}_{1i}, \mathbf{m}_{2i}]\}, i = 1, \ldots, N$, of non-coplanar corresponding pairs. If there is a pixel in the view 2 corresponding with $\mathbf{m}_1$, it is $\mathbf{m}_2^T \mathbf{l}_2 = 0$, hence, using eq. (2.5):

$$\mathbf{m}_2^T \mathbf{F}_{12} \mathbf{m}_1 = 0 \tag{2.9}$$

(Longuet-Higgins equation). It can be rewritten to:

$$\mathbf{Mf} = 0, \tag{2.10}$$

where $\mathbf{M}$ is a $9 \times N$ matrix containing coordinates of the corresponding pairs, and $\mathbf{f}$ is a 1×9 vector of $\mathbf{F}_{12}$ elements. In order to eliminate the scale factor, we introduce the constraint $\mathbf{f}^T\mathbf{f} = 1$. If the system of equations (2.10) is overdetermined, we obtain $\mathbf{f}$ as a solution to the optimization problem:

$$\min_{\mathbf{f}^T\mathbf{f}=1} \{(\mathbf{Mf})^T \mathbf{Mf}\}. \tag{2.11}$$

It can be easily shown that the solution is the norm eigenvector of the matrix $\mathbf{M}^T\mathbf{M}$ associated with the eigenvalue that has the smallest absolute value.

This procedure does not ensure the exact singularity of $\mathbf{F}_{12}$. For that, a non-linear optimization method would have to be used. Yet, the linear method is sufficient for we do not need $\mathbf{F}_{12}$ to be exactly singular, we need it only ro meet the relation 2.5 as accurately as possible.

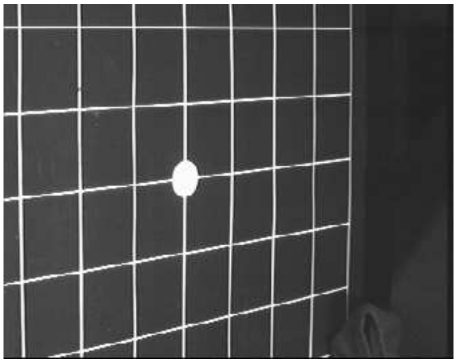

Figure 2.4: The calibration grid.

Acquisition of Calibration Points

We have shown that the fundamental matrix can be determined using a set of corresponding pixel pairs. The usual and convenient way how to obtain it is to use a *calibration grid.*

[8] We are aware that this assertion requires a more exact proof.

We use the calibration grid shown in the fig. 2.4, placed vertically on a calibrated rotating table. The calibration points are the grid intersections. To prevent ambiguity, the intersections are numbered relatively to the spot in the grid center. Since the grid is a plane surface, the non-coplanarity of the calibration points must be ensured by capturing several mutually rotated images of the grid for each view.

Let us outline how the grid intersection are detected in the image. First, the loci of the intersections are roughly determined as intersections of the grid lines, detected by the Hough transform [BB82]. Then, the precise positions of the intersections are obtained by template matching.

In the process of finding lines using the Hough transform, the search of local maxima in the Hough accumulator is a difficult step. We have observed that under certain assumptions, this 2-D search can be reduced to 1-D one and thus it can be considerably simplified. Let us denote the Hough accumulator $H(r, \Theta)$, where the line represented by the point $[r, \Theta]$ meets the relation $x \cos \Theta + y \sin \Theta = r$ ($[x, y]$ is the line point). Under the assumption that all lines have different Θ, $H(r, \Theta)$ can be reduced to $H_\Theta(r) = \sum_{all\, \Theta} H(r, \Theta)$. It can be shown that we are allowed to make this assumption in our case if we use the domain $\langle -R, R \rangle \times \langle 0, \pi \rangle$ for $[r, \Theta]$ and detect vertical and horizontal lines separately.

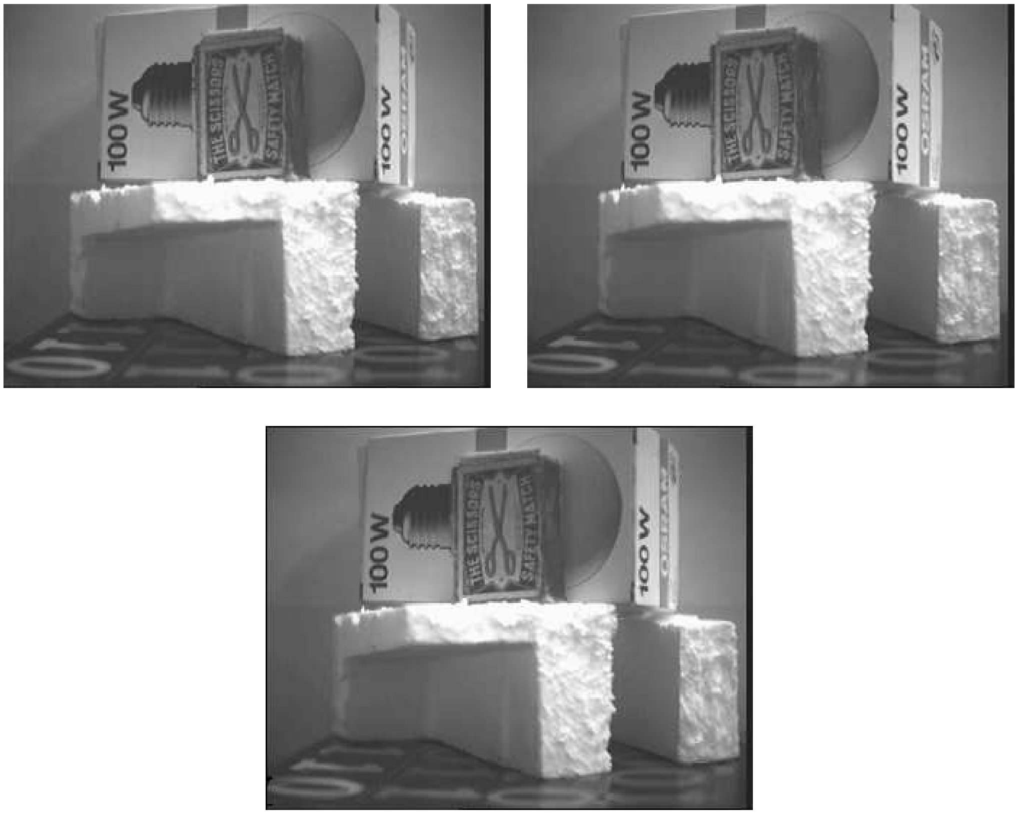

Figure 2.5: Two primary views of the first object (consisting of a book, a piece of styrofoam, a box, and a matches box), $\mathbf{p}_1 = [0°], \mathbf{p}_2 = [5°]$ (top) and the interpolated view $\mathbf{p}_1 = [2.5°]$, $\alpha_1 = \alpha_2 = 0.5$ (bottom).

2.2.3 Synthetic Images

Due to noise in images, it is impossible for any algorithm matching real images to yield error-free results. Though, this reliable information of correspondence is needed for testing the rendering algorithm (the results of the section 2.1). For that reason, the ray-tracer implemented in [Wer92] was extended to generate the correspondence of the synthetic images. Successful experiments with view interpolation were carried out, using these synthetic images as the primary views.

2.3 Experiments

We have conducted several experiments with the synthesis of interpolated views. We used real objects of quite complex shapes, for which the 3-D model reconstruction would have been difficult. For simplicity, we allowed objects only to rotate around a vertical axis, the view parameters thus having one component (azimuth angle).

Our experimental setup consisted of a camera, a calibrated rotating table and a calibration grid. The objects were placed on the table, and the views changed by rotating the table.

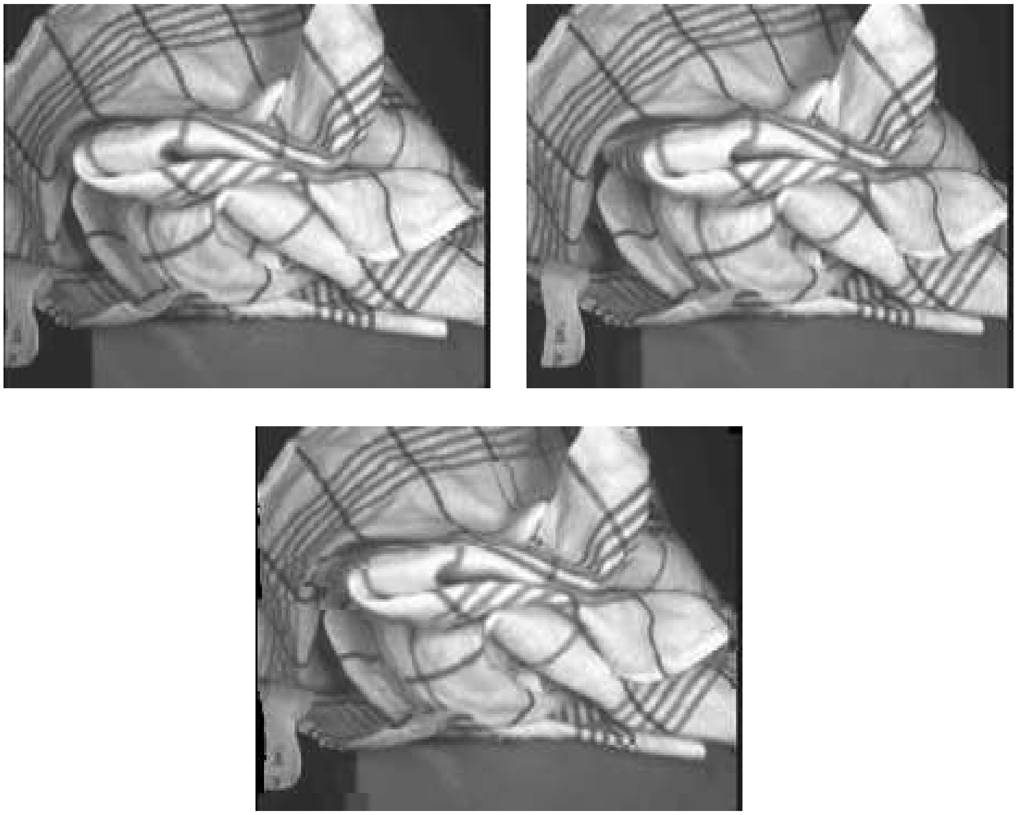

Figure 2.6: Two primary views of the second object (a linen towel), $\mathbf{p}_1 = [0°], \mathbf{p}_2 = [10°]$ (top) and the interpolated view $\mathbf{p}_1 = [5°]$, $\alpha_1 = \alpha_2 = 0.5$ (bottom).

As $dim(\mathbf{p}) = 1$, at least two real images are needed for calculating interpolated views. We captured two primary views of an object and matched them (after rectifying) using the described algorithm. Then we calculated interpolated views by view interpolation.

Fig. 2.5 shows one interpolated view of the first object. The difference between azimuth angles was 5°. Occluded areas, as well as holes due to unequal sampling frequencies were filled in by linear interpolation of intensities on epipolars, so the results of the section 2.1.2 were not utilized. Even then, no artifacts in the interpolated image are visible.

Fig. 2.6 shows a more complex object. The difference of the azimuth angles is 10°. As large areas of the object's surface are occluded, pixels in or near these areas were matched incorrectly. While this would cause rough errors in the 3-D model, the interpolated image is quite acceptable.

Figures 2.7 and 2.8 show primary and interpolated views of other objects.

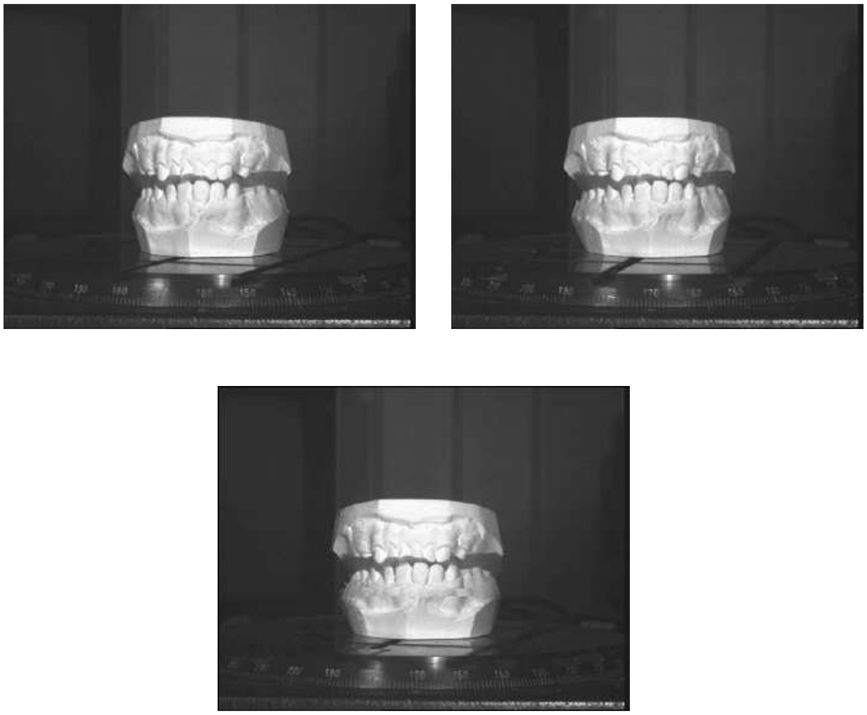

Figure 2.7: Two primary views of the object object (plaster cast of a set of teeth), $\mathbf{p}_1 = [0^\circ], \mathbf{p}_2 = [10^\circ]$ (top) and the interpolated view $\mathbf{p}_1 = [5^\circ]$, $\alpha_1 = \alpha_2 = 0.5$ (bottom).

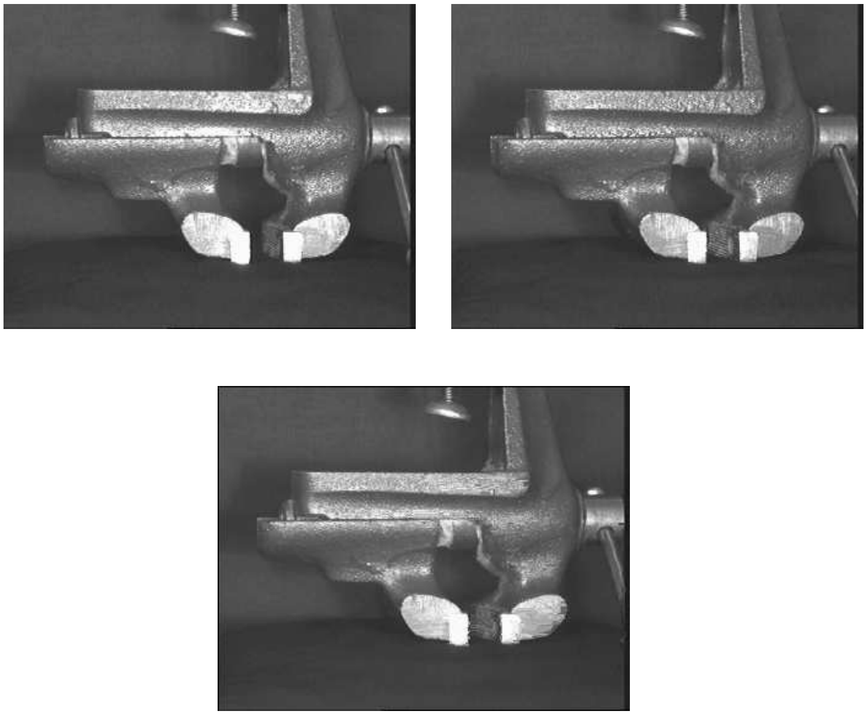

Figure 2.8: Two primary views of the object (a gripper), $\mathbf{p}_1 = [0^\circ], \mathbf{p}_2 = [7^\circ]$ (top) and the interpolated view $\mathbf{p}_1 = [3.5^\circ]$, $\alpha_1 = \alpha_2 = 0.5$ (bottom).

Chapter 3

Conclusion

3.1 Achieved Results

In this section the results achieved until presentation of this report are surveyed. These are:

- The thorough analysis of the state of the art in view interpolation and correspondence problem has been done.
- The promising direction for correspondence acquisition has been found. The binocular stereo matching algorithm [CHMR92] has the necessary robustness and directly yields the needed dense correspondence.
- The ray-tracer [Wer92] has been extended to generate synthetic correspondence. Thus reliable data for testing the rendering algorithms are available.
- The theoretical results have been achieved, indicating that the view interpolation is a feasible approach to rendering complex real-world objects, viewed from an arbitrary viewpoint.
- The encouraging experiments with view interpolation have been conducted, further supporting this assertion.
- The prototype which is able to find correspondence and interpolate between two primary views has been developed. In fact this is already sufficient for rendering any view of a real object with one degree of freedom of the viewpoint position (rotation around a fixed axis at a constant distance). Yet, the visibility problem solution has not been incorporated in this prototype.
- By studying the literature and experimenting, I have gained the sufficient theoretical knowledge and practical experience with almost all aspects of the task. I believe this will make possible the rapid progress in solving the remaining problems. The only problem with which I have not dealt, yet, is the choice of the smallest sufficient set of primary views.

For implementation of the described algorithms, I used the software system MATLAB and the programming language C and C++.

3.2 Further Work and Plans for the Thesis

To bring the research to a successful end, I still plan to do the following:

- The analysis of errors caused by linearization in the eq. 2.2.
- To test the rendering algorithm on the synthetic data.

- Further work on the matching algorithm. There is a number of possibilities how to improve its performance.
- Analysis of the state of the art in the problem of characteristic views.
- To find at least a partial solution to the problem of the choice of the smallest sufficient set of primary views.
- To make an extension for color images.
- To implement the final version of the prototype on the multiprocessor-multidisk system [GKLH94] in Lausanne. I expect to achieve a frame generation rate sufficient for real-time moving the objects on the screen.
- If there is some time left, I would also like to carry out some experiments with self-calibration[1]. Avoiding the explicit process of calibration gives a chance to handle objects of potentially any size (e.g., outdoor objects).

I believe that the direction of my research till the end of my postgradual study is determined, and that the results achieved are sufficient for finishing my doctor thesis successfully.

[1] The weak self-calibration is meant here (though this term is not usual).

Bibliography

[AWR90] Amir A. Amini, Terry E. Weymouth, and Jain C. Ramesh. Using dynamic programming for solving variational problems in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(9):855–867, Sep 1990.

[BB82] Dana H. Ballard and Christopher M. Brown. *Computer Vision*. Prentice-Hall, Inc.,, Englewood Cliffs, New Jersey 07632, first edition, 1982.

[BK88] K. L. Boyer and A. C. Kak. Structural stereopsis for 3-d vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(2):144–166, Mar 1988.

[BT80] Stephen T. Barnard and William B. Thompson. Disparity analysis of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(4):333–340, Jul 1980.

[CF82] I. Chakravarty and H. Freeman. Characteristic views as a basis for three-dimensional object recognition. In *Proc. of the Society for Photo-Optical Instrumentation Engineers Conference on Robot Vision*, pages 37–45, 1982.

[DKY89] Stanley M. Dunn, Richard L. Keizer, and Jongdaw Yu. Measuring the area and volume of the human body with structured light. *IEEE Transactions on Systems, Man and Cybernetics*, 19(6):1350–1364, 1989.

[Fau93] Olivier Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. The MIT Press, 1993.

[FLM92] Olivier D. Faugeras, Q. T. Luong, and S. J. Maybank. Camera self-calibration: Theory and experiments. In *European Conference on Computer Vision*, pages 321–333, 1992.

[FR93] Olivier Faugeras and Luc Robert. What can two images tell us about a third one? Technical Report 2018, INRIA, France, 1993.

[GKLH94] B. A. Gennart, B. Krummenacher, L. Landron, and R. D. Hersch. Gigaview parallel image server performance analysis. In *World Transputer Congress'94*, pages 120–135. IOS Press, September 1994.

[GR91] Bjorn Gudmundsson and Michael Randen. Fast generation of volume projection sequences. Linkoping University, Dept. of Electrical Eng., Sweden, 1991.

[HS89] Radu Horaud and Thomas Skordas. Stereo correspondence through feature grouping and maximal cliques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1168–1180, Nov 1989.

[HS81] K. P. Horn and B. G Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[CHMR92] Ingemar J. Cox, Sunita Hingorani, Bruce M. Maggs, and Satish B. Rao. Stereo without regularization. Technical report, NEC Research Institute, Princenton, 4 Independence Way, Princeton, NJ 08540, USA, Oct 21 1992.

[JW87] Lowell Jacobson and Harry Wechsler. Derivation of optical flow using spatiotemporal-frequency approach. *Computer Vision, Graphics and Image Processing*, (38):29–65, 1987.

[Kos92] Andreas Koschan. Methodic evaluation of stereo algorithms. In Reinhard Klette and Walter G. Kropatsch, editors, *Theoretical Foundations of Computer Vision.* Akademie Verlag, 1992.

[Kos93] Andreas Koschan. What is new in computational stereo since 1989: A survey on current stereo papers. Technical Report 93–22, Technische Universitaet Berlin, 1993.

[LS93] Jin Liu and Robert Skerjanc. Stereo and motion correspondence in a sequence of stereo images. *Signal Processing: Image Communication*, V(4):305–318, June 1993.

[Nis84] H. K. Nishihara. Practical real-time imaging stereo matcher. *Optical Engineering*, 23(5):536–545, September, October 1984.

[OK85] Yuichi Ohta and Takeo Kanade. Stereo by intra- and inter-scanline search using dynamic programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(2):139–154, Mar 1985.

[OWI86] Yuichi Ohta, Masaki Watanabe, and Katsuo Ikeda. Improving depth map by right-angled trinocular stereo. Institute of Sciences and Electronics, University of Tsukuba, Japan, 1986.

[Sha93] A. Shashua. On geometric and algebraic aspects of 3d affine and projective structures from perspective 2d views. Technical Report AI Memo No. 1405, Massachusetts Institute of Technology, Artifical Intelligence Laboratory, July 1993.

[SL92] Robert Skerjanc and Jin Liu. Computation of intermediate views for 3dtv. In Reinhard Klette and Walter G. Kropatsch, editors, *Theoretical Foundations of Computer Vision.* Akademie Verlag, 1992.

[SL91] Robert Skerjanc and Jin Liu. A three camera approach for calculating disparity and synthesizing intermediate pictures. *Signal Processing: Image Communication*, 4(1):55–64, 1991.

[SP90] Doron Sherman and Shmuel Peleg. Stereo by incremental matching of contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(11):1102–1106, Nov 1990.

[Sze94] Richard Szeliski. Image mosaicing for tele-reality aplications. Technical Report CRL 94/2, Digital Equipment Corporation, Cambridge Research Lab, May 1994.

[UB89] Shimon Ullman and Ronen Basri. Recognition by linear combinations of models. Memo 1152, MIT Artificial Intelligence Laboratory, August 1989.

[VO90] P. Vuylsteke and A. Oosterlinck. Range image acquisition with a single binary-encoded light pattern. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(2), Feb 1990. reprint.

[WAH92] Juyang Weng, Narendra Ahuja, and Thomas S. Huang. Matching two perspective views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(8), Aug 1992.

[Wer92] Tomáš Werner. Illumination models in computer graphics. Master's thesis, Czech Technical University, Karlovo náměstí 13, 12135 Praha 2, Czech Republic, 1992. In Czech.

[YKK86] Masahiko Yachida, Yoshifumi Kitamura, and Masatoshi Kimachi. Trinocular vision : New approach for correspondence problem. Dept. of Control Eng., Osaka University, Japan, 1986.

[ZL93] Avideh Zakhor and Franacesco Lari. 3d camera motion estimation with applications to video compression and scene reconstruction. In *SPIE Symposium on Electronic Imaging*, pages 1–14, Dept. of Electrical Eng. and Computer Sciences, University of California, Berkeley CA94720, 1993.