Matching in Videoimages: Camera Registration and CNN Based Feature Extraction

Zoltán Szlávik

A thesis submitted for the degree of
Doctor of Philosophy

Scientific adviser:
Tamás Roska, D. Sc.

Supervisor:
Tamás Szirányi, D. Sc.

Analogic and Neural Computing Laboratory
Computer and Automation Research Institute
Hungarian Academy of Sciences

Budapest, 2006

"Oh Man, strive on, strive on, have faith; and trust!"

„Mondottam ember! Küzdj, és bízva bízzál!"

Imre Madách

Acknowledgements

I would like to say thank

To *my family* for their undiminished faith and support.

To *prof. Tamás Roska* for his patience, help and support during my study and work.

To *prof. Tamás Szirányi* for his support and for inspiring discussions and ideas.

To *prof. Ronald Tetzlaff* for the time that I could spend in Frankfurt working on research projects.

To my first supervisors *prof. Naum N. Aizenberg* and *Igor N. Aizenberg* for supporting and educating me in many ways during my university years.

To my friend and colleague *László Havasi* whose critical view infected me, I believe, in a positive way.

To my colleagues, with whom I discussed lot of questions and worked together during the last years in different projects:

Csaba Benedek, Levente Kovács, László Orzó, István Petrás, Csaba Szepesváry, Levente Török.

To the MTA-SzTAKI for the support of my Ph.D. studies.

To my professors and teachers in

Uzhgorod National University

Dajka Gábor High School, Ungvár

I am grateful to *Katalin Keserű* and *Gabi Kékné* from MTA-SZTAKI for their practical and official aid.

Finally, thanks National R&D Program of Hungary, TeleSense project, Grant 035/02/2001; NoE MUSCLE project of the European Union, Economic Competitiveness Operative Programme of Hungary, Video Indexing project, Grant AKF388.

Contents

List of Figures

List of Tables

1 Introduction

Computer-assisted observation of human or vehicular traffic movements, natural reserves or any kind of activities using multiple cameras is now a subject of great interest for many applications. Examples are semi-mobile traffic control using automatic calibration, or tracking of objects in a surveillance system. The use of single camera limits the number of possible applications - even simplest applications need multiple cameras. Typical scenarios of multiple camera surveillance could be found in banks, airports, parking lots, stations etc. Here the observation with a single camera is not possible because of occlusions and cameras' limited field of view. Such systems/algorithms should work robustly and real-time. In computer vision the development of fast and/or robust algorithms is still a great challenge. Typical multi camera system performs the following steps:

1. change or motion detection;
2. object detection;
3. classification of objects – position, class, features etc.
4. tracking of objects;
5. event detection.

During my work I tried to answer some of the above problems. On the one hand, I have developed real-time algorithms for the analysis of face images and on the other hand, I have developed robust algorithms for the matching of images.

Object detection and tracking is a well defined problem for single images and image sequences. The detected object must be found in the later frames, which is a correspondence problem between the current frame and next frames. Multiple camera tracking means finding the same object in different views. To exploit information exchange between different cameras a correspondence must be established between them. A possible solution is the matching of different views of the same scene.

Matching different images of a single scene may be difficult, because of occlusion, aspect changes and lighting changes that occur in different views. Still-image matching algorithms [23][26][53][54] search for still features in images such as: edges, corners, contours, color, shape etc. They are usable for image pairs with small differences; however they may fail at occlusion boundaries and within featureless regions and may fail if the chosen primitives or features cannot be reliably detected. The views of the scene from the various cameras may be very different, so we cannot base the decision solely on the color or shape of objects in the scene. In a multi-camera observation system the video sequences recorded by cameras can be used for estimating matching correspondences between different views. Video sequences in fact also contain information about the scene dynamics besides the static frame data. Scene dynamics is an inherent property of the scene independently of the camera positions, the different zoom-lens settings and lighting conditions. References [39] and [25] present approaches in which motion-tracks of the observed objects are aligned. In practice, the existing algorithms can be used only in restricted situations. In case of scenes including several objects in random motion, successful registration of images from separate cameras conventionally requires some *a priori* object definition or some human interaction. But in most cases the extra information of *a priori* object models or human interaction is not available. During my work I focused on approaches that can establish a correspondence between different views without a priori defined object models and scene structures in fully automatic way.

Fast algorithms must be developed to be able to integrate several algorithms, e.g. object detection, tracking, classification, into a single system. The most natural way of identification of human in images or videos is the analysis of their faces' image. Humans' face is a non-intrusive biometric feature, which means that the identification can be done without disturbing the observed human.

In the literature of the computer vision many examples about how the face images can be analysed are described. Usually a face detection of face recognition algorithm [58][62] is built up from the following steps: i) detection of face like images (face candidates); ii) verification of face candidates; iii) identification. In the verification step about each face candidate the decision of being a face or not must be made. A possible solution when the face candidate is compared to an average face or to a face model [57]. Another possible solution is when facial features are extracted and their geometrical relationship verifies the face candidate [56]. Such algorithms need a fast facial feature extractor method. During my work I have developed a real-time facial feature extraction algorithm, which was implemented on ACE4K Cellular Visual Microprocessor, a real-hardware for real time image processing.

2 Motion based matching of images

Image to image matching algorithms are restricted to the information contained in single static images, e.g. the spatial variations within images, which capture the scene appearance. However, the dynamics of the scene contains much more information than a still image does. It is a property of the scene that is common to all videos recording the same scene, even when taken from different cameras.
This chapter focuses on motion based image matching algorithms. We will show that matching can be done without assuming any preliminary information about objects' appearance, scene's structure and dynamics.

2.1 Introduction

Image registration or matching is a basic task in many computer vision and digital photogrammetry applications. It is an important subtask of image based 3D reconstruction, camera-view registration and calibration in multicamera systems. The use of single camera for observation limits the number of possible applications, even simplest applications need multiple cameras. Typical scenarios of multiple camera surveillance could be found in banks, airports, parking lots etc. It is not possible for one camera to observe activity in the whole scene because of occlusions and cameras' limited field of view. To exploit the information exchange among additional cameras, it is necessary to find a correspondence (a registration or matching) between different views. Consequently, transforming the activity captured by separate individual video cameras from the respective local image coordinates to a common spatial frame of reference is a prerequisite for global analysis of the activity in the scene.

In the literature of computer vision, many examples about how the registration of different views has been achieved are described, together with the associated problems. Matching different images of a single scene may be difficult, because of occlusion, aspect changes and lighting changes that occur in different views. Basically, the existing methods can be divided into two groups: those that are still-image based, and image sequence based ones.

2.1.1 Still-image based methods

Still-image matching algorithms can be classified into two categories. In *"template matching"* the algorithms attempt to correlate the gray levels of image patches, assuming that they are similar for a given object-element in the two images [22][54]. This approach appears to be usable for image pairs with small differences; however it may fail at occlusion boundaries and within featureless regions. In the other category, *"feature matching"*, the algorithms first extract salient primitives (edges, contours etc.) from images, and match them in two or more views [23][26][53]. An image can then be described by a graph with primitives as nodes and geometric relations defining the links between nodes. The registration is then performed by the mapping of the two graphs (subgraph isomorphism). These methods are fast in execution because the subset of the image pixels that needs to be processed is small; but they may fail if the chosen primitives cannot be reliably detected. Wide-baseline camera positions also call for more distinguished analysis. The views of the scene from the various cameras may be very different, so we cannot base the decision solely on the color or shape of objects in the scene.

2.1.2 Motion based methods

In the other major group, the *motion-based methods*, the employed algorithms try to establish correspondences between different views by analyzing the dynamics of the scene as recorded by different cameras. Image sequences in fact contain information about the scene dynamics as well as the static data in each frame. The dynamics is an inherent property of the scene which is independent of camera positions, zoom-lens settings or lighting conditions. In [25] and [39] the tracks of moving objects are the basic features for matching the different views. In this case the capability of robust object tracking is assumed, which is the weak point of the method. In [6] a method is reported that finds matching points by calculating co-motion statistics. We employ the same idea of co-motion here; but the drawback of the exhaustive search method as implemented in [6] is the need for huge amounts of memory for storing co-motion statistics for each pixel. In [9] the approach of using co-motion statistics was extended by using gait analysis of human subjects in the scene; this technique makes registration possible even for non-overlapping views. In [37] a correspondence only between the tracked objects is achieved by analyzing entry/exit events as seen by the different cameras. However, the authors also assume that a robust tracker method is available. In [36] two non-overlapping views were registered by registering two static cameras to a moving camera the view of which had a view overlapping the views of the static cameras. The main drawback of this method is its exponential complexity. However, they also assume that a robust tracker method is available. In a recent work [42] registration of views is based on the extraction of "interesting" segments of planar trajectories. The success of image-registration is mainly reliant on the accuracy of the object tracker. The authors assume that only a limited number of objects are observed in both views, which is not an acceptable restriction in most of practical situations. These methods also assume that the objects of interest are moving on the ground plane and that the cameras are far distant from the scene, so that the height of the moving objects is small enough for them to appear in the recorded videos as "moving blobs on the ground", as seen in Figure 2-1 (right).

2.1.3 The proposed approach

In practice, the existing algorithms can be used only in restricted situations. The reported methods focus on the solution of the view-registration problem in respect of outdoor scenes for a given plane where points are assigned, and neglect the additional difficulties, which tend to arise for indoor scenes. In case of indoor cameras, see Figure 2-1 (left), the still-image based methods may fail due to the variability of conditions: occlusions, changing illumination etc. Due to the larger size of the moving objects, the cited motion-based methods will also fail; the observed motions are not necessarily on the ground-plane – while for outdoor scenes, such an assumption can safely be made.

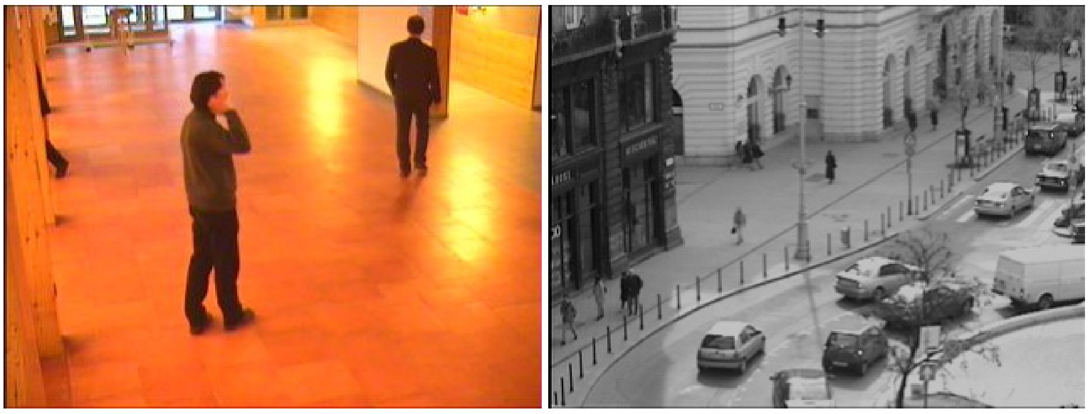


Figure 2-1. Examples of indoor (left) and outdoor (right) images.

In case of scenes including several objects in random motion, successful registration of images from separate cameras conventionally requires some *a priori* object definition or some human interaction. The extra information of *a priori* object models or human interaction is not available in most cases. We therefore tend to prefer approaches that can establish a correspondence between different views in fully automatic way without a priori defined object models.

Usually an algorithm for registration of different views contains the following steps:

(i) extraction of feature points;

(ii) extraction of candidate corresponding points;

(iii) estimation of the model(or transformation) that does the matching

    a. model based rejection of outliers;

    b. estimation of the model.

Our method follows the above general algorithm. The feature extraction is divided into three steps; in each of these steps, a probability of being interesting point is estimated about every output point of the preceding step and a decision has been made about interesting points for later processing. The algorithm is looking for feature points where objects are moved through and significant changes are detected. Detection of candidate corresponding points is performed by estimation of concurrently moving feature points. A moving object in scene produces simultaneously (concurrently) changing pixels (the projections of the objects in the images) in both cameras. Similarity between concurrently changing pixels (the feature points) is measured by comparing their history of change detection. Then outliers are rejected by fitting a model to the detected candidate corresponding points. After that the matching transformation is estimated from the inlier corresponding points.

2.2 Assumptions, notations and definitions

We assume that two time-synchronized video cameras (synchronized by any algorithm preferred by user, e.g. Berkeley algorithm [52]) are observing the same scene, and that their fields of view partially overlap. We assume standard static cameras having no lens distortions. However, possible lens distortions produced of cameras with wide-angle lenses can be corrected by using well-known methods described in Chapter 7.4 of [34].

The widely used pinhole model, outlined below, is used to describe the imaging produced by the cameras. A 3D point $\underline{M} = (x, y, z)^T$ in a world-coordinate system and its projected image $\underline{m} = (u, v)^T$ are related by:

$$s\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = P\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}, \qquad (1)$$

where s is an arbitrary scale-factor. P is a 3×4 matrix, called the perspective projection matrix (or the camera matrix). Denoting the homogeneous coordinates of an arbitrary vector $\underline{x} = (x, y)^T$ by $\underline{\tilde{x}} = (x, y, 1)^T$, we get $s\underline{\tilde{m}} = P\underline{\tilde{M}}$.

The perspective projection matrices of camera 1 and camera 2 are P_1 and P_2; the images of the cameras are I_1 and I_2. Point $\underline{m}$ in the ith image plane I_i is noted by $\underline{m}_i$, where the subscript $i = 1,2$ denotes the point in the image produced by the corresponding camera.

Using the above notation we can formally define the overlapping field of view (OFV) of two cameras with camera matrices P_1 and P_2 as $OFV = \{\underline{M} \mid \exists \underline{\tilde{m}}_i, P_i\underline{\tilde{M}} = \underline{\tilde{m}}_i \in I_i; i = 1,2\}$. The projections of OFV in I_1 and I_2 are: $OFV_i = \{\underline{m}_i \mid \underline{M} \in OFV;\ \underline{\tilde{m}}_i = P_i\underline{\tilde{M}}\}$ where $i = 1,2$.

Points m_{1i} and m_{2k} are called corresponding points projected from the same real-world point M if $P_1\tilde{M} = \tilde{m}_{1i}$ and $P_2\tilde{M} = \tilde{m}_{2k}$.

2.3 Mathematical background

In this chapter we briefly describe the main mathematical definitions and methods that we used in our experiments.

2.3.1 Models for matching

Applying motion based methods for image matching implies different models for matching in case of different types of motions. When the observed motions are on the groundplane the matching can be modelled by a 2-D homography. In case of 3D motions image matching can be modelled by epipolar geometry.

2.3.1.1 Homography

The 2-D homography H is a projective transformation that can be represented by a 3*3 matrix and can be calculated from at least 4 corresponding points by implementing the Direct Linear Transform algorithm [34].

The first question is how many point correspondences are required to compute the projective transformation. On the one hand, the matrix H contains 9 entries, but is defined only up to scale. On the other hand, each point-to-point correspondence accounts for two constraints, since for each pointing the first image the two degrees of freedom of the point in the second image must correspond to the mapped point. A 2D point has two degrees of freedom, each of which may be specified separately. Alternatively, the point is specified as a homogeneous 3-vector, which also has two degrees of freedom since scale is arbitrary. As a consequence, it is necessary to specify four point correspondences in order to constrain H fully.

The transformation is given by the equation $m'_{1i} = Hm_{2i}$. Note that this is an equation involving homogeneous vectors, thus the vectors m'_{1i} and Hm_{2i} are not equal, they have the same direction but may differ in magnitude by a non-zero scale factor. It can be shown by elementary manipulations that this equation is equivalent to the system of equations:

$$\begin{bmatrix} 0^T & -\omega_i' m_{2i}^T & v_i' m_{2i}^T \\ \omega_i' m_{2i}^T & 0^T & u_i' m_{2i}^T \end{bmatrix} \begin{pmatrix} h^1 \\ h^2 \\ h^3 \end{pmatrix} = 0, \tag{2}$$

where ω_i' is the scaling factor for m_{1i}'. Writing (2) in matrix form:

$$A_i h = 0, \tag{3}$$

where A_i is now a 2×9 matrix. The equations hold for any homogeneous coordinate representation of the point m_{1i}'. Each point correspondence gives rise to two independent equations in the system. Given a set of four such point correspondences, we obtain a set of equations $Ah = 0$, where A is built from the matrix rows A_i contributed from each correspondence. A has rank 8, and thus the system of equations has a non-zero solution, which can only be determined up to a non-zero scale factor. However, H is in general only determined up to scale, so the solution h gives the required homography.

2.3.1.2 Epipolar geometry

The epipolar geometry between two views is essentially the geometry of the intersection of the image planes with the pencil of planes having the baseline (the line joining the camera centres) as axis. The fundamental matrix is the algebraic representation of epipolar geometry. In the following we derive the fundamental matrix from the projective camera model as in [54].

Considering the case of two cameras as shown in Figure 2-2. Let C_1 and C_2 be the optical centers of the first and second cameras, respectively. Given a point m_{1i} in the first image, its corresponding point in the second image is constrained to lie on a line called the epipolar line of m_{1i}, denoted by $l_{m_{1i}}$. The line $l_{m_{1i}}$ is the intersection of the plane Π, defined by m_{1i}, C_1 and C_2 (known as the epipolar plane), with the second image plane I_2. This is because image point m_{1i} may correspond to an arbitrary point on the semi-line C_1M (M may be at infinity) and that the projection of C_1M on I_2 is the line $l_{m_{1i}}$. All epipolar lines of the points in the first image pass through a common point e_2, which is called the epipole. e_2 is the intersection of the line C_1C_2 with the image plane I_2. For each point m_{1i} in the first image I_1, its epipolar line $l_{m_{1i}}$ in I_2 is the intersection of the plane Π^k, defined by m_{1i}, C_1 and C_2, with image plane I_2. All epipolar planes Π^k thus form a pencil of planes containing the line C_1C_2. They must intersect I_2 at a common point, which is e_2. The symmetry of epipolar geometry leads to the following observation. If m_{1i} (a point in I_1) and m_{2k} (a point in I_2) correspond to a single physical point M in space, then m_{1i}, m_{2k}, C_1 and C_2 must lie in a single plane. This is the well-known co-planarity constraint or epipolar equation in solving motion and structure from motion problems when the intrinsic parameters of the cameras are known.

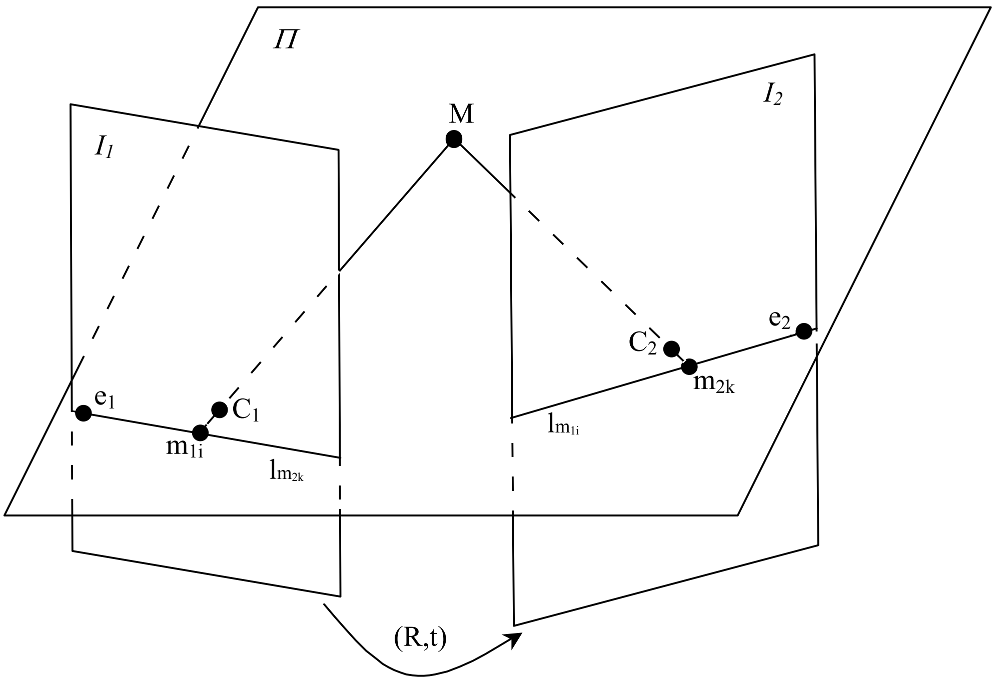


Figure 2-2. The epipolar geometry

Let the displacement from the first camera to the second be (R, t). Let m_{1i} and m_{2k} be the images of a 3-D point M on the cameras. We assume that M is expressed in the coordinate frame of the first camera. Under the pinhole model, we have the following equations:

$$\begin{aligned} s_1\tilde{m}_1 &= A_1\begin{bmatrix} I & 0 \end{bmatrix}\begin{bmatrix} M \\ 1 \end{bmatrix} \\ s_2\tilde{m}_2 &= A_2\begin{bmatrix} R & t \end{bmatrix}\begin{bmatrix} M \\ 1 \end{bmatrix}, \end{aligned} \tag{4}$$

where A_1 and A_2 are the intrinsic matrices of the first and second cameras, respectively. Eliminating M, s_1 and s_2 from the above equations, we obtain the following fundamental equation:

$$\tilde{m}_2^T A_2^{-T} TRA_1^{-1} \tilde{m}_1 = 0 , \tag{5}$$

where T is such that $Tx = t \wedge x$ for all 3-D vector x ($\wedge$ denotes the cross product). Equation (5) is a fundamental constraint underlying any two images if they are perspective projections of one and the same scene. Let $F = A_2^{-T} TRA_1^{-1}$, F is known as the fundamental matrix of the two images. Without considering 3-D metric entities, we can think of the fundamental matrix as providing the two epipoles and the 3 parameters of the homography between two epipolar pencils. This is the only geometric information available from two uncalibrated images [41]. This implies that the fundamental matrix has only seven degree of freedom. Equation (5) says no more than that correspondence in the right image of point m_{1i} lies on the corresponding epipolar line.

Properties of Fundamental Matrix:

- Geometrically, the fundamental matrix represents a mapping from the 2-dimensional projective plane of the first image to the pencil of epipolar lines through the epipole. Thus, it represents a mapping from a 2-dimensional onto a 1-dimensional projective space, and hence must have rank 2.
- F has seven degrees of freedom: a 3*3 homogeneous matrix has eight independent ratios (there are nine elements, and the common scaling is not significant); however, F also satisfies the constraint $\det F = 0$ which removes one degree of freedom.
- If F is the fundamental matrix of the pair of cameras (P_1, P_2), then F^T is the fundamental matrix of the pair in the opposite order (P_1, P_2).

2.3.1.3 The normalized 8-point algorithm for the computation of fundamental matrix

The fundamental matrix is defined by the equation

$$m_1^T F m_2 = 0 \tag{6}$$

for any pair of matching points in two images. Given at least 7 point matches this equation can be used to compute the unknown matrix F. Denote by f the 9-vector made up of the entries of F in row-major order. Then from a set of n point matches we can write a set of n linear equations with unknown f.

$$Af = \begin{bmatrix} u_{11}u_{21} & u_{11}v_{21} & u_{11} & v_{11}u_{21} & v_{11}v_{21} & v_{11} & u_{21} & v_{21} & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_{1n}u_{2n} & u_{1n}v_{2n} & u_{1n} & v_{1n}u_{2n} & v_{1n}v_{2n} & v_{1n} & u_{2n} & v_{2n} & 1 \end{bmatrix} \mathrm{f} = 0 . \tag{7}$$

This is a homogeneous set of equations and F can only be determined up to scale. Solution exists if and only if matrix A have rank 8. If the input data is not exact, because of noise in point coordinates, then the rank of A may be greater than 8. In this case a least square method is used to find the necessary solution. The least square solution for f is the singular vector corresponding to the smallest singular value of A. The solution found in this way minimizes $\|Af\|$ subject to the condition $\|f\| = 1$. This algorithm is known as the 8 point algorithm.

To ensure the singularity of the obtained fundamental matrix further calculations must be done. A convenient way of this is the use of SVD. Let $F = UDV^T$ be the SVD of F, where $D = diag(r,s,t)$ satisfying $r \geq s \geq t$. Then the matrix $F' = Udiag(r,s,0)V^T$ will be the closest singular matrix to F under a Frobenius norm [34].

Application of a simple translation and scaling of the input points in the image before formulating the linear equations leads to great improvement in the conditioning of the problem and in the robustness of the result [34]. The suggested normalization is a translation and scaling of each image so that the centroid of the reference points is at the origin of the coordinates and the distance of the points from the origin is equal to $\sqrt{2}$. The overall normalized 8 point algorithm for the computation of the fundamental matrix is as follows.

1. Normalization: transform the image coordinates according to $\hat{x}_i = Tx_i$ and $\hat{x}'_i = T'x'_i$, where T and T' are transformations consisting of a translation and scaling.
2. Find the fundamental matrix $\hat{F}'$ corresponding to the matches $\hat{x}_i \leftrightarrow \hat{x}'_i$ by
    a. Linear solution: determine $\hat{F}$ from the singular vector corresponding to the smallest singular value of $\hat{A}$, where $\hat{A}$ is composed from the matches $\hat{x}_i \leftrightarrow \hat{x}'_i$ as in ().

b. Constraint enforcement: replace $\hat{F}$ by $\hat{F}' = \hat{U} diag(\hat{r}, \hat{s}, 0)\hat{V}^T$, where $\hat{F} = \hat{U}\hat{D}\hat{V}^T$ and $\hat{D} = diag(\hat{r}, \hat{s}, \hat{t})$.

Denormalization: Set $F = T'^T \hat{F}' T$.

2.3.2 Robust model estimation

In many practical situations the assumption that we have a set of ideal correspondences only perturbed with measurement error is not valid. In practice many of correspondences are mismatched. These mismatched points are the outliers for the model to be estimated and they can severely disturb the estimated model. The goal is then to determine a set of inliers from the given set of correspondences so that the model can be estimated in an optimal manner from them.

The estimation of models and rejection of outliers is performed by the RANSAC procedure. The RANSAC procedure is opposite to that of conventional smoothing techniques: rather than using as much of data as possible to obtain an initial solution and then attempting to eliminate the invalid data points, RANSAC uses as small initial data as feasible and enlarges this set with consistent data when possible [31]. The steps of RANSAC algorithm for fitting a model (which can be estimated from s points) to a data set S are:

1. Randomly select a sample of s data points from S and instantiate the model from this subset.
2. Determine the set of data points S_i which are within a distance threshold C_1 of the model. The set S_i is the consensus set of the sample and defines the inliers of S.
3. If the size of S_i (number of inliers) is greater than some threshold C_2, re-estimate the model using all the points in S_i and terminate.
4. If the size of S_i is less than C_2, select a new subset and repeat the above.
5. Calculate ε and N as in (8).
6. After N trials the largest consensus set S_i is selected, and the model is re-estimated using all the points in the subset S_i.

N is chosen sufficiently high to ensure with a probability (0.99 in our experiments) that at least one of the random samples of s data points is free from outliers.

$$N = \log(1-p) / \log(1-(1-\varepsilon)^s), \tag{8}$$

where $\varepsilon = 1 - (numberofinliers) / (totalnumberofpoints)$.

For the homography estimation the symmetric transfer error is calculated to determine whether a point is inlier or outlier:

$$E_{STE} = \sum_i D(x_i, H^{-1}x_i')^2 + D(x_i', Hx_i)^2, \tag{9}$$

where $D(\cdot)$ is the Euclidean distance of point $x_i(x_i')$ to its transformed pair $H^{-1}x_i'(Hx_i)$; H is the estimated homography; x_i and x_i' are coordinates of point correspondences in the input images of different views.

In case of 3D scene, in the estimation of the fundamental matrix the following error function is calculated to determine whether a point is inlier or outlier:

$$E_{DTEL} = \sum_i D(x_i, Fx_i')^2 + D(x_i', F^T x_i)^2, \tag{10}$$

where $D(\cdot)$ is the Euclidean distance of point $x_i(x_i')$ to its epipolar line $Fx_i'(F^T x_i)$; x_i and x_i' are coordinates of point correspondences in the input images.

2.3.3 Introduction to Markov chains

Let us consider a stochastic process $\{X_n, n = 0,1,2,...\}$ that takes on a finite or countable number of possible values. The set of possible values of the process will be denoted by the set of nonnegative integers $\{0,1,2,...\}$. If $X_n = i$, then the process is said to be in state i at time n. Suppose that whenever the process is in state i there is a fixed probability P_{ij} that it will next be in state j. So, it is supposed that:

$$P\{X_{n+1} = j \mid X_n = i, X_{n-1} = i_{n-1}, ..., X_1 = i_1, X_0 = i_0\} = P_{ij} \tag{11}$$

for all states $i_0, i_1, ..., i_{n-1}, i, j$ and for all $n \geq 0$. Such a stochastic process is known as a *Markov chain* [48]. Equation (11) means that, for a Markov chain, the conditional distribution of any future state X_{n+1} given the past states $X_0, X_1, ..., X_{n-1}$ and the present state X_n is independent of the past states and depends only on the present state.

The value P_{ij} represents the probability that the process will, when in state i, next make a transition into state j. Since probabilities are nonnegative and since the process must make a transition into some state, we have that:

$$\sum_{j=0}^{\infty} P_{ij} = 1, \mathrm{i} = 0,1,\dots . \tag{12}$$

The Markov property allows to express the probability of any finite sequence $\{X_{0,} X_1, \dots, X_n\}$ in terms of the initial distribution $\mu_0(i) = P\{X_0 = i\}$ and P_{ij}. Indeed, the Markov property implies that

$$P\{X_n = i_n, X_{n-1} = i_{n-1}, \dots, X_1 = i_1, X_0 = i_0\} = P_{i_n i_{n-1}} P_{i_{n-1} i_{n-2}} \cdot \dots \cdot P_{i_1 i_0} \mu_0(i_0) . \tag{13}$$

From this we also get

$$P\{X_n = i \mid X_0 = j\} = \sum P_{i\, i_{n-1}} P_{i_{n-1} i_{n-2}} \cdot \dots \cdot P_{i_1 i_j} = P_{ij}^{(n)} . \tag{14}$$

$P_{ij}^{(n)}$ may be viewed as the $(i,j)-th$ entry of the matrix P^n, where $P = (P_{ij})$ is the transition matrix of the Markov chain. According to the (12) matrix P is a stochastic matrix. Given the initial distribution of the Markov chain μ_0, the distribution of X_n is then given by:

$$\mu_n = P^n \mu_0 . \tag{15}$$

A Markov chain is called an ergodic chain if it is possible to go from every state to every state (not necessarily in one move). A Markov chain is called a regular chain if some power of the transition matrix has only positive elements. In other words, for some n, it is possible to go from any state to any state in exactly n steps. It is clear from this definition that every regular chain is ergodic. On the other hand, an ergodic chain is not necessarily regular.

Any transition matrix that has no zeros determines a regular chain. However it is possible for a regular chain to have a transition matrix that has zeros. Of special interest are the long-time properties of a Markov chain. An interesting question whether the probabilities $P_{ij}^{(n)}$ are converging to some value as $n \to \infty$ or not. The following theorems state two important results for regular chains.

Theorem. Let P be the transition matrix for a regular chain. Then, as $n \to \infty$, the powers P^n approach a limiting matrix W with all rows the same vector w. The vector w is a strictly positive probability vector.

This theorem states that the probability of being in X_j in the long run is approximately w_j and is independent of the starting state.

Theorem. Let P be the transition matrix for a regular chain, let $W = \lim_{n\to\infty} P^n$, let w be the common row of W, and let c be the column vector all of whose components are 1. Then

- $wP = w$, and any row vector v such that $vP = v$ is a constant multiple of w.
- $Pc = c$, and any column vector x such that $Px = x$ is a multiple of c.

An immediate consequence of the theorem is that there is only one probability vector v such that $vP = v$. Then the limiting probabilities can be calculated from the system of linear equations:

$$\begin{aligned} &wP = w, \\ &\sum_j w_j = 1, \\ &w_j \geq 0. \end{aligned} \tag{16}$$

2.4 *Accumulated motion statistics for camera registration*

In this chapter we will show that accumulated motion statistics can be used for robust and precise camera registration without any *a priori* information about observed objects.

2.4.1 Change detection

In the implemented experimental system the videos of urban traffic were recorded with standard digital cameras, so the images are blurred and noisy. The size of moving objects has a great variety; there are small blobs (walking people) and huge blobs (trams or buses), too. Background extraction in outdoor videos is still a challenging task and cannot be done perfectly because of changes in illumination. A good comparison of different methods can be found in [33]. In our method we do not need precise motion detection and object extraction, these minor errors are irrelevant because of the later statistical processing.

The motion blobs are extracted by using simple running-average background subtraction [33] by using the reference background image $B_c(x,y;t)$. A thresholding operation is performed to classify a pixel as foreground if (17) holds.

$$|I_c(x,y;t) - B_c(x,y;t)| > \tau , \tag{17}$$

where $c = \{1,2\}$ is the camera index; $t \in Z$ is the frame index and $B_c(x,y;t)$ is estimated by:

$$B_c(x,y;t) = \beta I_c(x,y;t-1) + (1-\beta)B_c(x,y;t-1), \quad 0 < \beta < 1 . \tag{18}$$

In the experiments we use $\beta = 0.8$ and noise threshold $\tau = 0.2$ according to [33]. In outdoor videos of urban traffic fast adaptation of the background model is quite important. The system should be able to follow changes in background caused by illumination or by traffic control. Large β corresponds to fast adaptation of $B_c(x,y;t)$. After thresholding morphological closing has been performed to eliminate isolated pixels and to connect object parts [33]. This approach seems to be a usable solution to detect the significant motions in the scene.

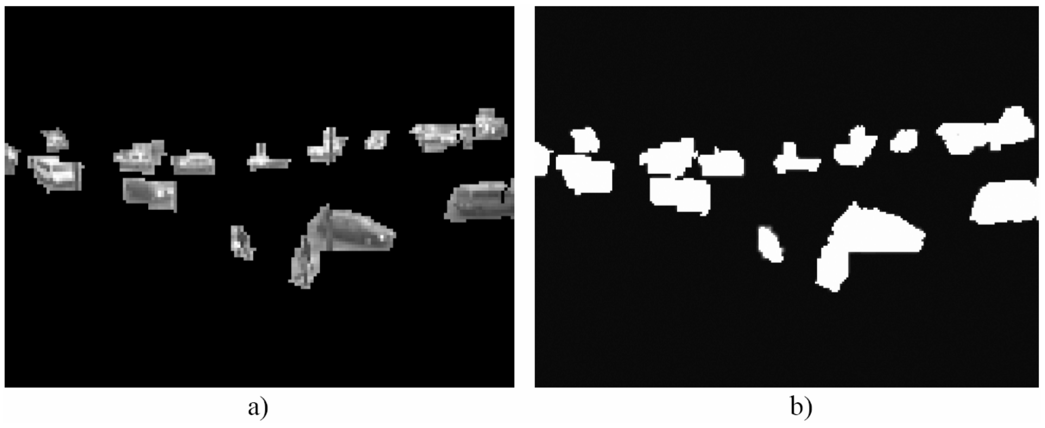


Figure 2-3. **The detected objects are shown in image (a); the corresponding change map is shown in (b).**

The image with the detected objects is the change map (*M*), which value for a given pixel (i, j) is equal to 1 if change is detected at that pixel (i, j) and 0 else:

$$M(I_c, t) = \{m_{ij}(I_c; t)\}, \qquad 1 \le i \le h, \qquad 1 \le j \le w,$$
$$m_{ij}(I_c; t) = \begin{cases} 1, & \text{change is detected at pixel } (i, j) \in I_c, \\ 0, & \text{else.} \end{cases} \tag{19}$$

where h and w are the height and width of the input image I_c respectively. Sample result of the detected changes and the corresponding change map can be seen in Figure 2-3.

2.4.2 Definition of co-motion statistics

For finding point correspondences between two views in case of video sequences we have decided to analyze the dynamics of the scene. To do it co-motion statistics (accumulated statistics of concurrent motions) were used. In the case of a single video sequence a motion statistical map for a given pixel $(p,q) \in I_c$ can be recorded as follows: when change is detected at pixel $(p,q) \in I_c$, the coordinates are recorded for all pixels (in the same view – local, or in the other – remote) where change is also detected at that moment. In the motion statistical map the values of the pixels at the recorded coordinates are updated. Thus, it can be considered as the accumulation of the motion maps with respect to a given pixel. After all, this statistical map is normalized to have global maximum equal to 1. Formally, it can be expressed as:

$$MSM(I_c,p,q)=\frac{1}{\max\limits_{\substack{1\le i\le h\\1\le j\le w}}\Big(\sum\limits_{0<t\le T} M(I_c,t)m_{pq}(I_c,t)\Big)}\sum_{0<t\le T} M(I_c,t)m_{pq}(I_c,t), \qquad (20)$$

where I_c is the input image of the c-th camera; $M(I_c,t)$ is the change map of the t-th frame of the c-th camera, $t\in Z$; $m_{pq}(I_c,t)$ is the value of the motion map at the given pixel $(p,q)\in I_c$ at time t (t-th frame); T is the index of the last frame. Values in $MSM(I_c,p,q)$ assigned to pixel $(p,q)\in I_c$ are the conditional probabilities that change was detected at pixels of I_c when motion was detected at pixel $(p,q)\in I_c$. Examples of such motion statistics, shown as images, can be seen in the left image of Figure 2-4. The higher the value at a given position of $MSM(I_c,p,q)$ the brighter is the corresponding pixel in the image.

In case of stereo video sequences two motion-statistic maps are assigned to each point in the images: a local and a remote. Local map means that the motion-statistical map is calculated for the image from which the pixel is selected; the remote motion-statistical map refers to the correlated motions in the other image. After the change detected on the local side, for the points defined by the local motion map the local statistical map updated by the local motion map (21a). For each point where change is detected on the local side, the local motion map of the remote side updates the corresponding remote statistical map Eq. (21b). In this case the co-motion statistics express the conditional probability of concurrent changes in two different cameras. Examples of co-motion statistics are given in Figure 2-4 and Figure 2-5.

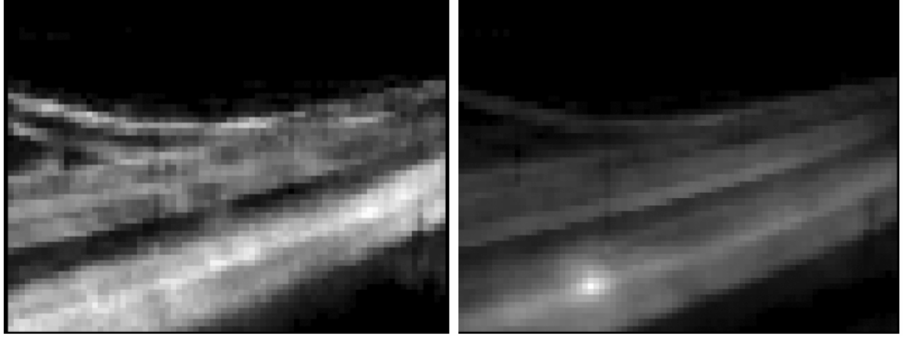


Figure 2-4. Remote statistical maps for different cases are in the pictures. In the left one for a pixel, which is not in the cameras' common field view; in the right one for a pixel from cameras' common field of view. The higher the value at a given position of RMSM the brighter is the corresponding pixel in the image.

So, according to the above definition the local and remote motion statistical maps (*LMSM* and *RMSM* respectively) for a given pixel $x = (p,q) \in I_1$ are calculated by the formulas:

$$LMSM(I_1; p,q) = \frac{1}{\max\limits_{\substack{1\le i\le h\\1\le j\le w}}(\sum\limits_{0<t\le T} M(I_1;t)m_{pq}(I_1;t))} \sum_{0<t\le T} M(I_1;t)m_{pq}(I_1;t), \tag{21a}$$

$$RMSM(I_1; p,q) = \frac{1}{\max\limits_{\substack{1\le i\le h\\1\le j\le w}}(\sum\limits_{0<t\le T} M(I_2;t)m_{pq}(I_1;t))} \sum_{0<t\le T} M(I_2;t)m_{pq}(I_1;t)' \tag{21b}$$

where $I_c, c \in \{1,2\}$ are the input images of size $h \times w\,(\text{height} \times \text{width})$ pixels; $M(I_c;t)$ is the change map of image I_c at time t (t-th frame); $m_{pq}(I_k;t)$ is a scalar value of the motion map of image I_k at the given pixel (p,q) at time t (t-th frame); t is the index of the current frame and T is the index of the last frame.

According to the above definition, $RMSM(I_c, p,q)$ for a given pixel (p,q) is a distribution over the pixels of the second view. It expresses the probability of concurrent changes in the second camera with respect to changes detected at (p,q). The pixel $(p_{\max}, q_{\max})$ from the other camera, see (22), will be the pixel which grey value changes most concurrently with the changes detected at (p,q).

$$(p_{\max}, q_{\max}) = \underset{(p_c, q_c)\in I_{other}}{\arg\max}\ RMSM(I_c; p,q) \tag{22}$$

If (p,q) is from the common view of the cameras, then $(p_{\max}, q_{\max})$ should be the corresponding point of (p,q) in the other view. Thus, by extracting maxima of $RMSM(I_c, p,q)$ we will get the set of point correspondences $\{((p,q) \in I_c; (p_{\max}, q_{\max}) \in I_{other})\}$. An example of such a correspondence is illustrated in Figure 2-5. It also follows from the definition of $RMSM(I_c, p,q)$ that $(p_{\max}, q_{\max})$ could be a pixel where changes are detected continuously. In this case, the extracted corresponding points are mostly outliers. An example of such a failure is shown in Figure 2-6. The next step of the algorithm is the rejection of outliers. For this, additional constraints are used.

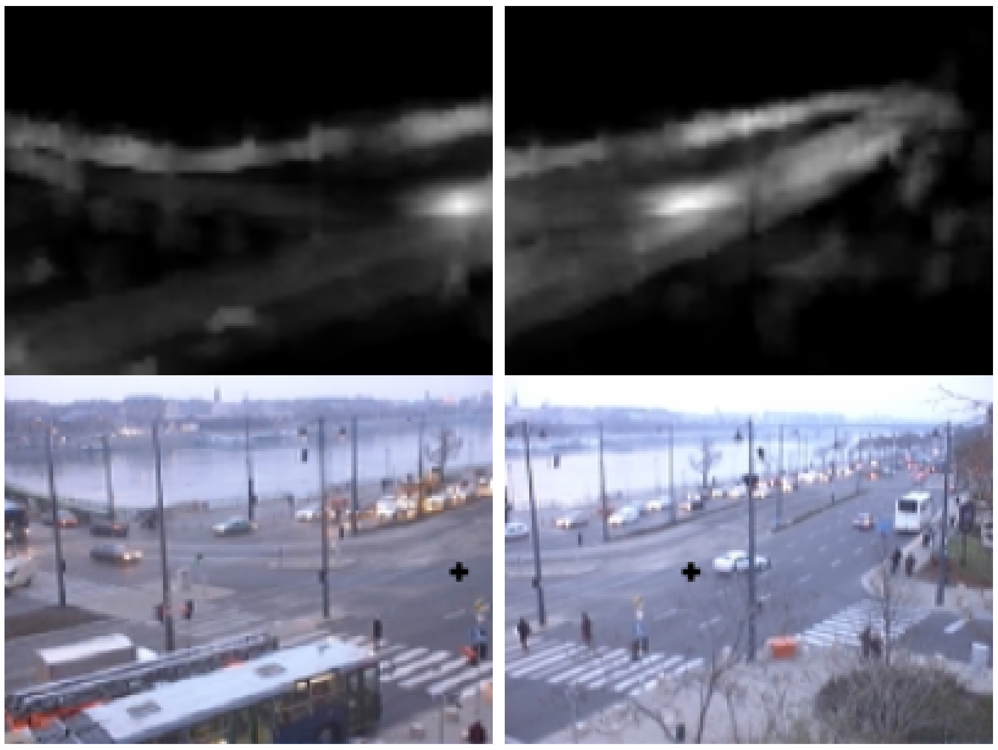


Figure 2-5. Top images: example of co-motion statistics for inlier point-pairs. Below: a corresponding point-pair is shown in the images of the left and right cameras.

2.4.3 Outlier rejection

As candidate matches we choose global maximums on local and remote statistical images. For the rejection of outliers with respect to the models from the set of candidate point-pairs we have implemented the RANSAC algorithm [31]. The proportion of outliers in the set of point candidates is essential for the RANSAC algorithm [31] that we have used for the rejection of outliers and estimation of the geometric relation between the views. Larger rate of outliers means a longer running time of the RANSAC algorithm [34]. In order to reduce the number of outliers we have implemented an algorithm based on some neighborhood rules, which is then followed by the robust estimation of the geometric model.

2.4.3.1 Neighborhood rules

For the rejection of outliers from the set of point correspondences we applied the principle of “good neighbors” [54]. The principle of “good neighbors” says that if we have a true match, then we will have many other true matches in some neighborhood of it. Consider a candidate match (m_1, m_2) where m_1 is a point in the first image and m_2 is a point in the second image. Let $N(m_1)$ and $N(m_2)$ be the neighbors of m_1 and m_2. If (m_1, m_2) is a true match, we will expect to see many other matches (n_1, n_2), where if $n_1 \in N(m_1)$ then $n_2 \in N(m_2)$. So, candidate pairs for which less other candidate pairs could be found in their neighborhood were eliminated to avoid noisy correspondences, following the method in [54].

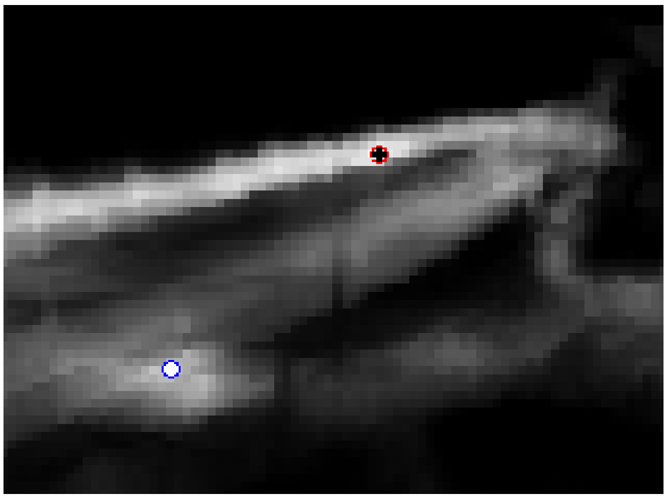


Figure 2-6. The global maximum is the circle, while it should be the cross.

The reduced set of point-correspondences also has erroneous matches due to the errors caused by properties of co-motion statistics:

From the global motion statistics (23) we know image regions where much more moving objects are detected than in other places. These image regions correspond to regions where the observed objects are moving continuously. If we have a point $(p,q) \in I_c$ where the correspondent scene location is not in the field of the view of the other camera, then the maximum of $RMSM(I_c, p, q)$ will usually be in a wrong place, in a point where the value of the motion statistics in the global motion statistical map is high, see Figure 2-6.

Because of the size of the moving objects and the averaging in the accumulation of motion maps the global maxima of $RMSM(I_c, p, q)$ could be shifted and it will be somewhere in the neighborhood of the desired corresponding point. These "shifting"-s result in cases where different points from local statistical images are "mapped" onto the same point in remote statistical images. Thus, the correspondences are not unique in such cases.

To solve the first problem we need to eliminate points from the set of candidate matches if the global maximum on remote statistical images is a pixel where the value of the global motion statistics (23) is greater than a threshold value τ_1. Global motion statistics ($GMSM$), defined in (23), express the probability of motion detection in a single view. Because of normalization in (23) the values of $GMSM$ are from interval $[0;\ 1]$. A higher value of $GMSM$ at a pixel means that observed objects are moving continuously through that pixel, which is a typical situation when observing urban traffic. We have selected threshold τ_1 to be equal 0.8 to eliminate pixels corresponding to nearly continuous motions.

$$GMSM(I_c) = \frac{1}{\max\limits_{\substack{1\le i\le h\\ 1\le j\le w}}\Big(\sum\limits_{\substack{1\le p\le h\\ 1\le q\le w}} M(I_c;p,q)\Big)} \sum_{\substack{1\le p\le h\\ 1\le q\le w}} M(I_c;p,q), \tag{23}$$

where $GMSM(I_c),\ c\in\{1,2\}$ stands for global motion statistics of image I_c. To get over on the second problem we also need to eliminate points from the set of candidate matches if the global maximum on remote statistical image is a pixel with multiple assignments. This preliminary outlier rejection step is followed by a robust geometric model based outlier rejection that based on projection geometry.

2.4.3.2 Model estimation

Geometric models are used for the final alignment of the extracted point correspondences. For planar scenes (motions) the homography H is a projective transformation that can be represented by a 3*3 matrix and can be calculated from at least 4 point-pairs by implementing the Direct Linear Transform algorithm [34]. In case of 3D motions the fundamental matrix F relates the input images, which can be represented by a 3*3 matrix and can be calculated from at least 8 point-pairs by using the Normalized 8-point Algorithm [34]. The rejection of outliers with respect to the geometric models and robust estimation of the homography and fundamental matrix was performed by the RANSAC algorithm [31].

2.4.4 Fine tuning of point correspondences

The above described outlier rejection algorithm results in point correspondences, but these results must be fine-tuned for the precise estimation of point correspondences. The point coordinates can contain errors; they can be shifted by some pixels, due to the nature of co-motion statistics recording. Even if we have 1 pixel error in point coordinates the precise alignment of the views cannot be done. The implemented outlier rejection algorithm must be followed by a robust optimization to fine tune point correspondences and obtain subpixel accuracy.

An iterative technique is used to refine both the point placements and the geometric model. The method used is the Levenberg-Marquardt iteration [44]. The entries of the fundamental matrix or the homograhy as well as the coordinates of points in right camera's image are treated as variable parameters in the optimization, but the point coordinates of the left camera's image are kept constant. The initial condition for this iteration is the entries of the fundamental matrix or homography and point's coordinates estimated by using the above-described algorithm.

For the homography estimation the symmetric transfer error is minimized which is defined by:

$$E_{STE} = \sum_i D(x_i, H^{-1}x_i')^2 + D(x_i', Hx_i)^2 , \tag{24}$$

where $D(\cdot)$ is the Euclidean distance of point $x_i(x_i')$ to its transformed pair $H^{-1}x_i'(Hx_i)$; H is the estimated homography; x_i and x_i' are coordinates of point correspondences in the input images of different views.

In case of 3D scene in the optimization of point correspondences the fundamental matrix is parameterized in the form (25) to ensure that the estimated fundamental matrix will have rank 2 [34].

$$F = \begin{pmatrix} b & a & -ay_e - bx_e \\ -d & -c & cy_e + dx_e \\ dy_e' - bx_e' & cy_e' - ax_e' & -cy_e y_e' - dy_e' x_e + ay_e x_e' + bx_e x_e' \end{pmatrix}, \tag{25}$$

where (x_e, y_e) and (x_e', y_e') are the affine coordinates of the two epipoles; a, b, c, d are the parameters. The following error function is minimized in the optimization:

$$E_{DTEL} = \sum_i D(x_i, Fx_i')^2 + D(x_i', F^T x_i)^2 , \tag{26}$$

where $D(\cdot)$ is the Euclidean distance of point x_i (x_i') to its epipolar line Fx_i' ($F^T x_i$); x_i and x_i' are coordinates of point correspondences in the input images.

2.4.5 Time synchronization

Until now, we have assumed that the cameras' clocks are synchronized. For time synchronization many algorithms have been developed, e.g. the Berkeley algorithm [52]. In our case, if the cameras are not synchronized then the generated co-motion statistics should no longer refer to concurrent motions detected in two stereo sequences. So, when we apply our algorithm for outlier rejection, we do not get a "large" set of point correspondences, but more point correspondences can be extracted in the case of synchronized sequences.

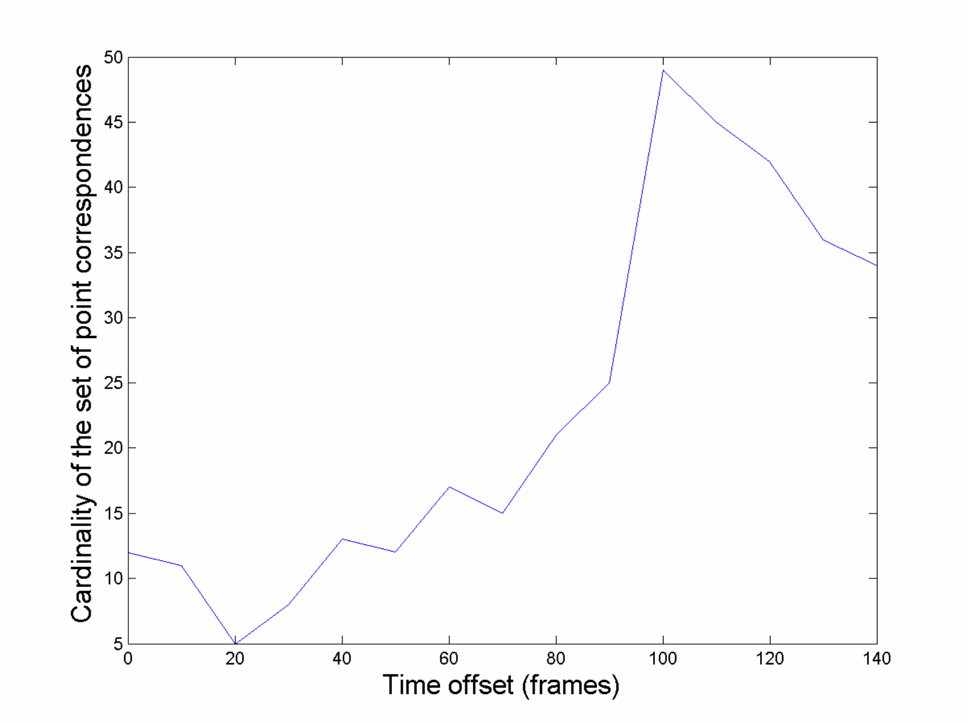


Figure 2-7. Cardinality of the set of point correspondences for different time offset values. The maximum is at 100 frames, which means that the offset between two sequences is 100 frames.

Since this observation is obvious and true in practice, we calculate point correspondences for different time offset values and then perform a one-dimensional search for the largest set of point correspondences to synchronize the sequences, see Figure 2-7.

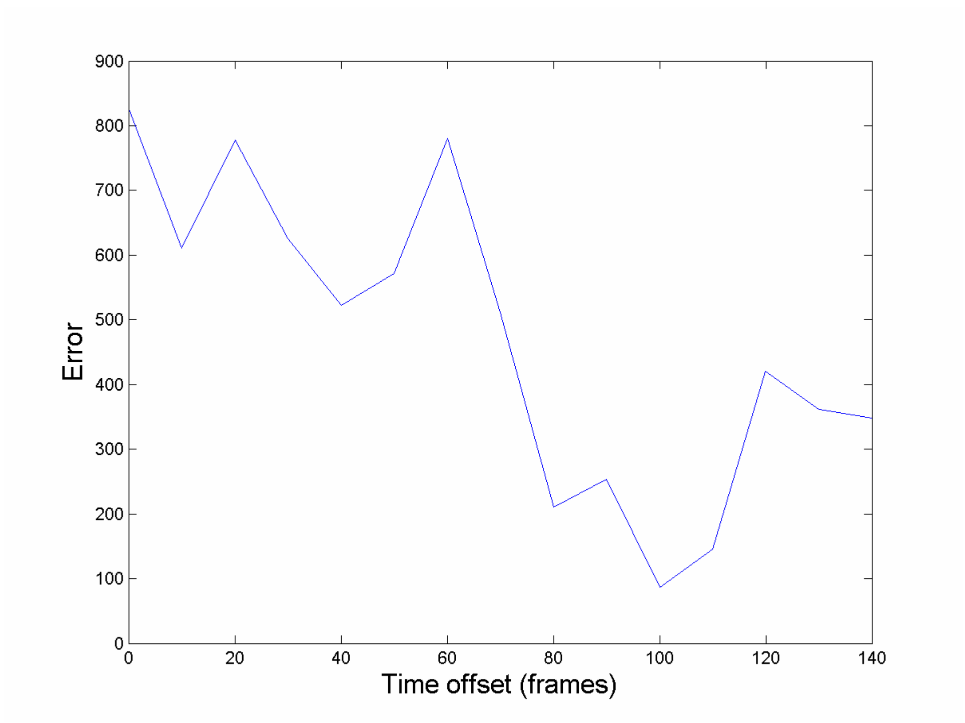


Figure 2-8. The change of the error rate for different offset values is in the diagram. The minimum is at 100 frames as the maximum for the cardinalities of sets of point correspondences.

It can be seen in Figure 2-7 that even in the case of unsynchronized sequences the algorithm produces point correspondences. But if we analyze the sum-of-square differences score (the reprojection error in this case), see Figure 2-8, we found that the global minimum is at offset value 100 frames, as the maximum in Figure 2-7 for the cardinalities of sets of point correspondences. This means that the global optimum is at offset value 100 frames, in all other cases the obtained point correspondences mean that the algorithm finds a local optimum.

2.4.6 Experimental results

The above-described approach was tested on videos captured by two cameras, having partially overlapping views, at Gellert (GELLERT videos) and Ferenciek squares (FERENCIEK videos) in Budapest. The GELLERT videos are captured at resolution 160×120, at same zoom level and with same cameras while the FERENCIEK videos are captured at resolution 320×240, at different zoom levels and with different cameras. The proposed outlier rejection algorithm rejects most (98%) of the candidate point pairs. For the GELLERT videos it results in 49 point-correspondences and in 23 for FERENCIEK videos. The computation time of the whole statistical procedure was about 10 minutes for 10 minutes of video presented in the figures.

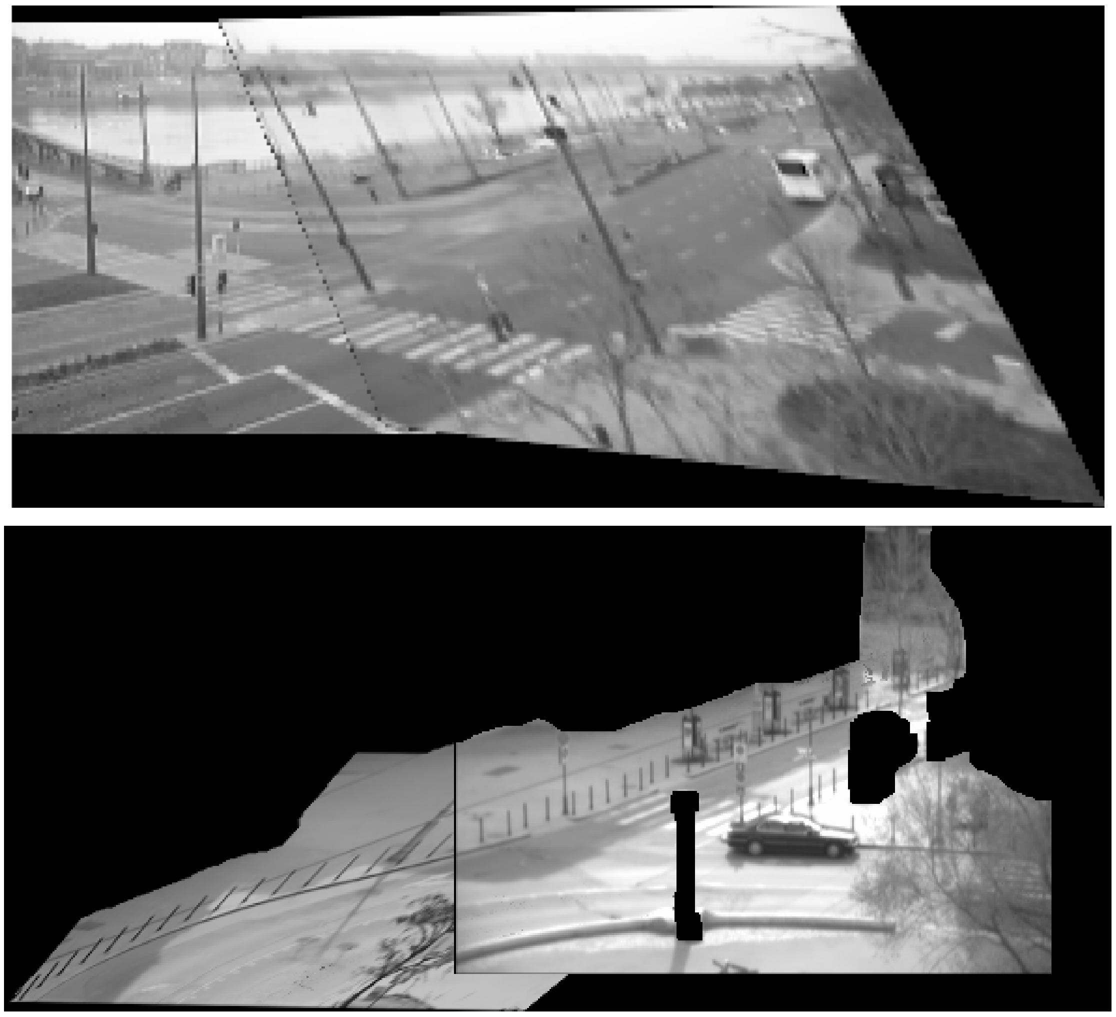


Figure 2-9. The constructed composite views are in the pictures. The upper one is generated for the GELLERT videos; the lower one is for the FERENCIEK videos.

2.5 Bayesian model for the extraction of concurrently changing points

This chapter will describe a feature extraction method for the extraction of concurrently changing pixels based on the analysis of their change histories. The aim of the feature extraction step is to reduce the region of interest before the extraction of matching points and model estimation. We have implemented a three-step procedure in which at each step the number of interesting points is reduced. The output is two sets of feature points, one from each image, which represents pixels where real objects moved through and with high probability have a corresponding point in the other image.

2.5.1 Change detection

For the purposes of our algorithm, we do not need precise (e.g. motion field estimation) change-detection and object-extraction. This is an advantage, since achieving such results is very time consuming. Any misclassifications are excluded in the course of the later statistical processing. In our experiments we have used the simplest change detection method: change between two consecutive frames was calculated as the absolute value of their difference image. This also covers the feature maps of most of the sophisticated other methods [45][50], but these higher level methods are usually based on some a priori information (motion/change speed, structure or shapes) and we wanted to avoid that. The output of change detection is a set of absolute values of differences denoted by c_{ij}, where i denotes the image (1,2), and j denotes the index of a single pixel within the image. For each pixel in I_1 and I_2 the change history, as a series of scalar values, is stored in a vector $\underline{c}_{ij}(t), t = 1,2,...,T$, where T is the length of the recorded image sequence. This vector will be used in further calculations to narrow-down the region of interest to be used in subsequent processing.

2.5.2 Entropy based preselection of feature points

Then, we use an entropy-based preselection of features to find possible corresponding pixel change histories, since the value of change detection itself ($\underline{c}_{ij}(t)$) is not a measure of importance. The algorithm filters out pixels of noise flickerings[1] and continuous motion, since their change histories cannot relate the input images. The aim is to detect pixels where "sometimes" (suddenly, not continuously) significant changes were observed. These changes will serve as markers in change history, which relate the potential corresponding points. We define pixels of significant changes with an entropy-based analysis of change histories. We calculate the complexity of motion history time-series $\underline{c}_{ij}(t)$ ($i = 1,2; j = 1,2,...,|I_i|$) by means of the entropy (27).

$$entropy = -\frac{1}{\log L}\sum_{q=1}^{L} p(x_q)\log p(x_q)\,, \tag{27}$$

where $p(x_q)$ is the probability that $x_q \leq \underline{c}_{ij}(t) < x_{q+1}$ for the qth bin of the histogram in which the bin-widths were set according to Scott's rule[2] [49], and L is the number of bins.

Noise flickering causes a uniform or wide-Gaussian histogram, resulting in a relatively high entropy value. In case of continuous motion, the value of continuous change is added to the same noise, resulting in a similar distribution and entropy value. In our case, changes of definite time-stamps are preferable, meaning that some sudden changes of significantly high value broaden the argument of the histogram. In this case the range of noise is shrunk into lower values, while we get additional peaks at the higher values: entropy decreases due to the less uniform distribution.

Experiments with different real-life indoor and outdoor videos demonstrate that the distribution of entropy of change history statistics can be approximated by a Gaussian distribution, as is shown plotted in Figure 2-10 for real life video samples. To evaluate the distribution of entropies, ground truth masks have been manually generated classifying into pure noisy pixels and motion-affected pixels.

[1] Under noise flickerings we mean the usual camera noise and similar effects that is always present in videos captured by digital cameras.

[2] Scott rule for interval width is $h = 3.5\sigma / \sqrt[3]{T}$, where σ is the standard deviation of $\underline{c}_{ij}(t)$.

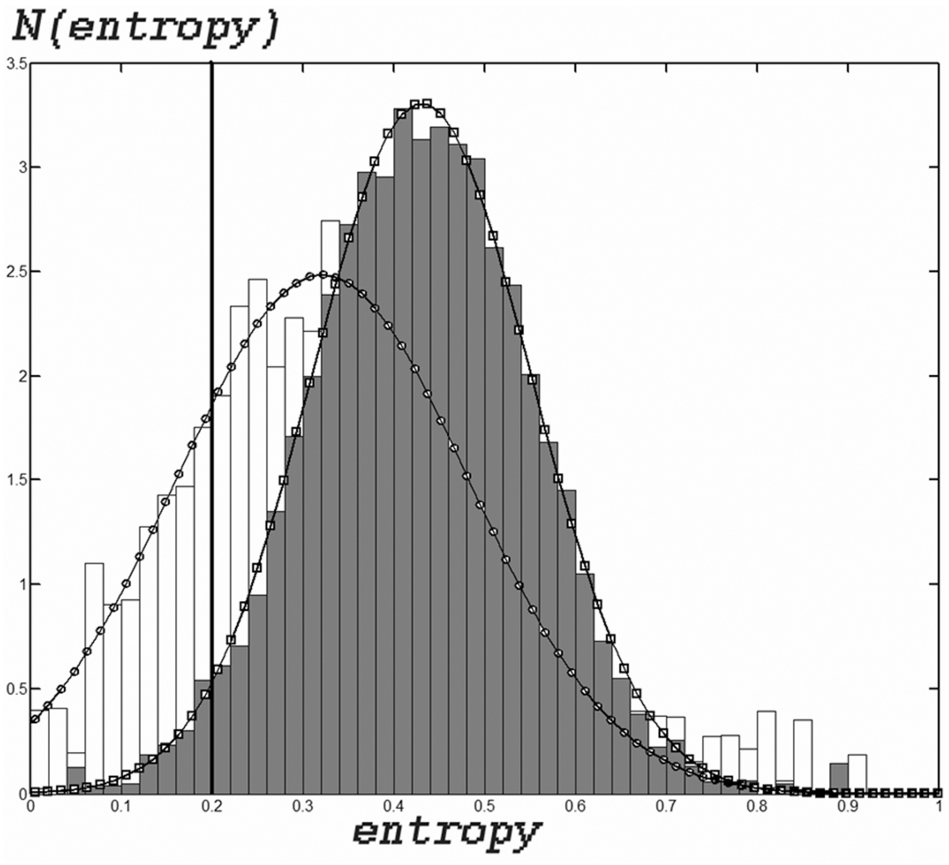


Figure 2-10. The distribution of entropy (24) for noise flickerings (marked with boxes) and deterministic motions (marked with circles) in real life videos. The black line shows a selected threshold at which the proportion of potential outliers is 15%. *N(.)* means Gaussian distribution.

Table 2-1. Parameters of entropy distribution for different test videos. Last column shows the proportion of noise flickerings among pixels of detected changes if the threshold value is 0.2.

	Real motions		Noise flickerings		Proportion of noise flickerings
	exp. value	variance	exp. value	variance	
Video 1	0.23	0.14	0.43	0.14	12%
Video 2	0.26	0.11	0.41	0.16	13%
Video 3	0.31	0.05	0.47	0.1	19%
Video 4	0.28	0.16	0.49	0.11	15%

A reasonably low proportion of potential outliers in the set of point candidates is essential for the RANSAC algorithm [31] that we used for the rejection of outliers and for model estimation. Based on the above considerations and Figure 2-10, choosing the set of potential correlating points is accomplished by collecting them starting from the ones with minimum entropy towards the greater values. If we have

enough points (and later the RANSAC can result in a small error) then we can stop the selection. If not, we can go up to a reasonable limit of entropy, namely 0.2, where the proportion of noise flickerings (the potential outliers) in the set of candidate feature points is still in the order of 15% in our testing examples.
As it was mentioned before, the aim of the algorithm is to select those pixels where "sometimes" (suddenly, not continuously) significant changes were observed. Continuous motion causes additional fluctuation in the detected change values of a given pixel and its entropy does not differ significantly from a common background pixel, due to the more uniform distribution of change values than in the case of sudden changes. Thus entropy at pixels of background or continuous motion is higher than that of the change histories of sudden changes. By selecting a threshold equal to 0.2 we can eliminate pixels of continuous motions and noise flickerings too.
Instead of using the offline evaluation of entropy in (27), which would need a lot of memory and computations for histogram estimation of the input $\underline{c}_{ij}(t)$, a different, quicker calculus can also be applied (28). Here the probability function on which the entropy is calculated is not the frequency function of change values, but the change value itself, as a measure of probable motion at a given time proportional to the value of change, see (28).

$$entropy^* = -\frac{1}{\log T}\sum_{t=1}^{T} \hat{\underline{c}}_{ij}(t)\log \hat{\underline{c}}_{ij}(t) \tag{28}$$

where $\hat{\underline{c}}_{ij} = \frac{\underline{c}_{ij}}{\sum_t \underline{c}_{ij}(t)}$.

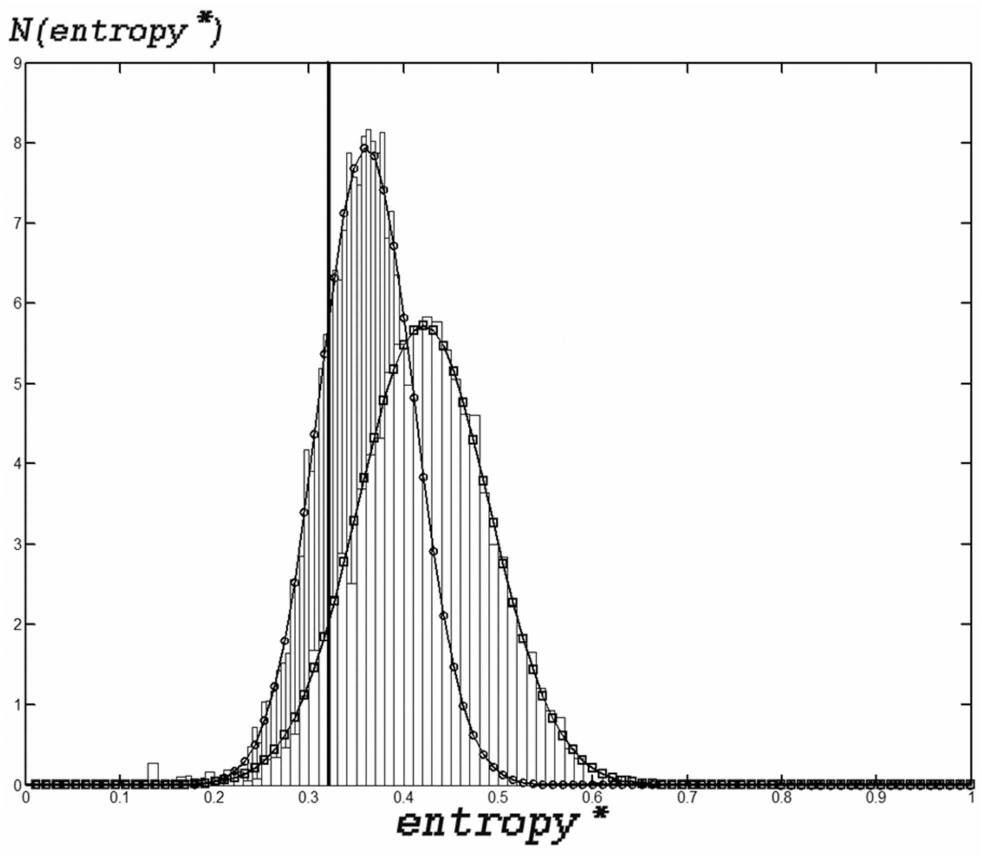


Figure 2-11. The distribution of *entropy (25) of noisy flickering (marked with boxes) and deterministic motions (marked with circles) for real life videos. The black line stands for the selected threshold. *N(.)* means Gaussian distribution.**

Noise flickering causes randomness in the motion history, resulting in a relatively high *entropy** value. In case of characteristic sudden changes at given times, it expropriates an important portion of the function's area, decreasing the *entropy** due to the less uniform distribution. Figure 2-11 shows the distribution of *entropy** of noise (flickerings) and deterministic motions, and Table 2-2 shows the parameters of the *entropy** distribution for the different test videos.

Table 2-2. Parameters of the Gaussian distribution of *entropy for different test videos. Last column shows the proportion of noise flickerings if the threshold value is 0.32.**

	Real motions		Noise flickerings		Proportion of noise flickerings
	exp. value	variance	exp. value	variance	
Video 1	0.37	0.05	0.45	0.043	16.6%
Video 2	0.33	0.06	0.43	0.05	12.7%
Video 3	0.30	0.04	0.41	0.06	17.6%
Video 4	0.35	0.054	0.39	0.051	9.0%

In a similar manner to the entropy calculated the first way (27), we show a threshold value for *entropy** where the proportion of noise signals is around 15% (in this case the threshold value is 0.32). The main advantage of formula (28) against formula (28) is that it can be calculated very rapidly, in fact on-line, from one frame to the next. The output of the entropy-based preselection of feature-points is a set of pixels with their change-history vectors for each input image.

2.5.3 Bayesian model for the extraction of the Overlapping Field of View

As the result of previous entropy based filtering, a set of image pixels with corresponding change histories are filtered in this step to get the probable points – the potential corresponding points – in the OFV area.

Consider two cameras with an overlapping field of view. Let $P(m_{ij})$ denote the probability that image pixel $m_{ij} \in OFV_i$. Let $P(m_{1i} \mid m_{2k})$ denote the probability of $m_{1i} \in OFV_1$ given the pixel m_{2k}. If $m_{1i} \in OFV_1$ then a m_{2k} must exist, such that m_{1i} and m_{2k} are corresponding points, so that at the same time changes were or were not detected. The conditional probability $P(m_{1i} \mid m_{2k})$ expresses the frequency of change detection at m_{1i} when change is detected at m_{2k} and can be calculated by the related formula (29).

$$P(m_{1i} \mid m_{2k}) = \frac{1}{\sum_{t=1}^{T} \underline{b}_{2k}(t)} \sum_{t=1}^{T} \underline{b}_{1i}(t) \underline{b}_{2k}(t), \tag{29}$$

where T is the length of the input video-image sequence; 300 frames in our experiments. Conditional probabilities $P(m_{1i} \mid m_{2k})$ are normalized such that $\sum_k P(m_{1i} \mid m_{2k}) = 1$, and $\underline{b}_{ij}(t)$

$$\underline{b}_{1i} = \begin{cases} 1, \underline{c}_{1i}(t) > K \\ 0, \quad \text{else} \end{cases}, i = 1,2,\ldots,N_1,$$

$$\underline{b}_{2k} = \begin{cases} 1, \underline{c}_{2k}(t) > K \\ 0, \quad \text{else} \end{cases}, k = 1,2,\ldots,N_2 \tag{29a}$$

are the binarized change-history vectors and K is as follows.

The role of K is to distinguish between significant changes and noise flickerings in change histories $\underline{c}_{ij}(t)$. The definition of significant changes could be different for the different views and positions in 3 dimensions, considering that projections of motion, lighting and occlusion may affect the results. For thresholding the changes we can find several semi-automatic or ad-hoc methods [45], but the adaptive thresholding in [47] may fit our case the best. It considers the thresholding of the change history as two classes of events (noise flickerings and significant changes) with each class characterised by a pdf. The method then maximizes the sum of the entropy (27) of the two pdfs to converge to a single threshold value. In our video samples above, this optimization method resulted in threshold values of about 3-20% of the range values in $\underline{c}_{ij}(t), t = 1,2,\ldots,T$. We have selected the above threshold K equal to 8% as an average value of the estimated thresholds.

If $P(m_{1i} \mid m_{2k})$ and $P(m_{2k} \mid m_{1i})$ are high then m_{1i} and m_{2k} may be corresponding points. This correspondence is not exact and unique because of the size of moving objects. A single pixel will correlate with a set of pixels from the other image. The correlation values of such set of concurrently changing pixels follow Gaussian distribution around true corresponding points, usually (but not necessarily) having maximal correlation for true corresponding points [35].

Applying Bayes' theorem we get:

$$P(m_{1i} \mid m_{2k}) = \frac{P(m_{2k} \mid m_{1i})P(m_{1i})}{\sum_j P(m_{2k} \mid m_{1j})P(m_{1j})} \tag{30}$$

Considering all the m_{1i} and their dependence on all m_{2k} we can write:

$$P(m_{1i}) = \sum_k P(m_{1i}m_{2k}) = \sum_k P(m_{1i} \mid m_{2k})P(m_{2k}), \tag{31}$$

Writing the same for $P(m_{2k})$ we have:

$$P(m_{2k}) = \sum_i P(m_{1i} m_{2k}) = \sum_i P(m_{2k} \mid m_{1i}) P(m_{1i}), \tag{32}$$

(31) and (32) is a system of linear equations with unknowns $P(m_{1i})$ and $P(m_{2k})$ and with coefficients calculated according to (29). Writing this system in matrix form

$$\left(\underline{p}_1 \quad \underline{p}_2\right) = \left(\underline{p}_1 \quad \underline{p}_2\right) \underline{\underline{\Pi}}, \tag{33}$$

where $\underline{p}_1$ and $\underline{p}_2$ are row vectors constructed from probabilities $P(m_{1i})$ and $P(m_{2k})$, respectively; $\underline{\underline{\Pi}} = \begin{pmatrix} 0 & P_{21} \\ P_{12} & 0 \end{pmatrix}$ is constructed from matrices $P_{12}(k,i) = P(m_{1i} \mid m_{2k})$ and $P_{21}(i,k) = P(m_{2k} \mid m_{1i})$. Let us consider the following matrix (34) constructed from $\underline{\underline{\Pi}}$:

$$\hat{\underline{\underline{\Pi}}} = \frac{1}{2}(I + \Pi), \tag{34}$$

Then $\hat{\underline{\underline{\Pi}}}$ can be considered as a transition matrix of an ergodic regular Markov chain with states $\{\{m_{1i}\}, \{m_{2k}\}\}$ [32]. Then, according to the Frobenius-Perron theorem [32] such a Markov chain has a unique stationary distribution π, which can be calculated by the equation:

$$\pi = \hat{\underline{\underline{\Pi}}} \pi \tag{35}$$

It is easy to check that the solutions of (35) and (33) are the same. Thus the derived system of linear equations (33) for the calculation of $P(m_{1i})$ and $P(m_{2k})$ has a unique solution.

In the following, we derive fixed point iteration for (33). Substituting (30) into (31) gives:

$$P(m_{1i}) = \sum_k \frac{P(m_{2k} \mid m_{1i}) P(m_{1i}) P(m_{2k})}{\sum_j P(m_{2k} \mid m_{1j}) P(m_{1j})}, \tag{36}$$

which formula results in the following iteration scheme:

$$P(m_{1i})_{r+1} = P(m_{1i})_r \sum_k \frac{P(m_{2k} \mid m_{1i}) P(m_{2k})_r}{\sum_j P(m_{2k} \mid m_{1j}) P(m_{1j})_r} \tag{37}$$

Due to the symmetry of the above considerations with respect to the $P(m_{2k})$, iteration (37) can be extended with the estimation of $P(m_{2k})$:

$$P(m_{2k})_{r+1} = P(m_{2k})_r \sum_i \frac{P(m_{1i} \mid m_{2k}) P(m_{1i})_r}{\sum_j P(m_{1i} \mid m_{2j}) P(m_{2j})_r} \tag{38}$$

In the iteration (37) and (38) initial values of $P(m_{1i})$ and $P(m_{2k})$ are set to $1/N_1$ and $1/N_2$, where N_1 and N_2 are the numbers of preselected feature points in images I_1 and I_2 after the entropy-based processing. We need to select those features which are in OFV_1 and OFV_2, for which $P(m_{1i})$ and $P(m_{2k})$ are increasing. The sum of probabilities $P(m_{1i})$ and $P(m_{2k})$ remains *1.0* during the iterations. This means that if the probabilities of some pixels are increasing then the probability of some other pixels must decrease. Due to noisy change detection, finally, only one point pair will "survive" the Bayesian iterations (37) and (38). In our case this Bayesian iteration converges to a nearly corresponding point-pair as a unique solution of (33), but this correspondence is by no means exact. The obtained (potential) point correspondences are correspondences only in a statistical sense (according to the defined conditional probabilities in (29)), but they are not correspondences according to a model that relates the two images geometrically, as the model information is not included in (29). Running the procedure again, excluding the previous pairs, we can get more correspondences. Examples of such extracted correspondences are illustrated in Figure 2-12.

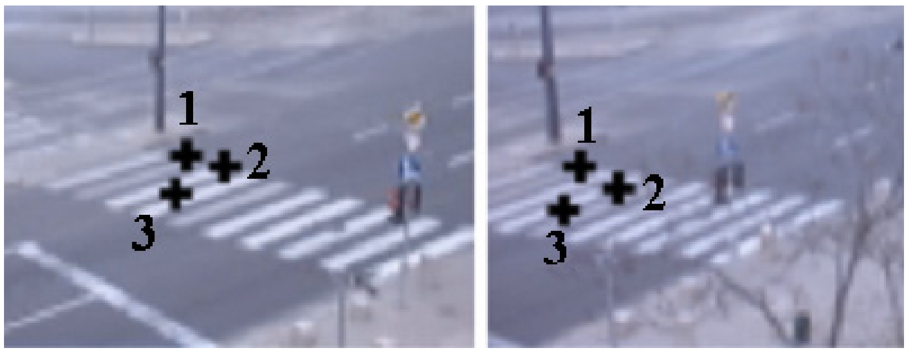


Figure 2-12. Sample point pairs obtained by Bayesian iteration (15) and (16). The nearly corresponding points are numbered with the same number.

However, this Bayesian process is very time consuming and the set of resulting point-pairs might need a further selection through optimization. Instead of a fully converging Bayesian iteration series, we can run the series of double iterations only once up to a limited number of iterations. A similar iteration-cutoff is also applied for blind deconvolution [38]. After four of the double iteration steps the algorithm is stopped and those feature-points are selected for which $P(m_{1i})$ and $P(m_{2k})$ are greater than $1/N_1$ and $1/N_2$ respectively.

Table 2-3. Reduction of ROI after feature extraction steps

Feature extraction step	Size of ROI in pixels	
	I_1	I_2
Input	19200	19200
Entropy based preselection	2311	3253
Bayesian iteration	636	671

This set of points in the image of each view well defines the projected common viewing area. In Table 2-3 it can be seen that the regions of interesting points (ROI) are significantly reduced and the obtained two set of points are in the projections (images) of the cameras' OFV, as shown in Figure 2-13. This truncated double-iteration method does not supply us with selected point-pairs, but it does give a good estimate of the OFV.

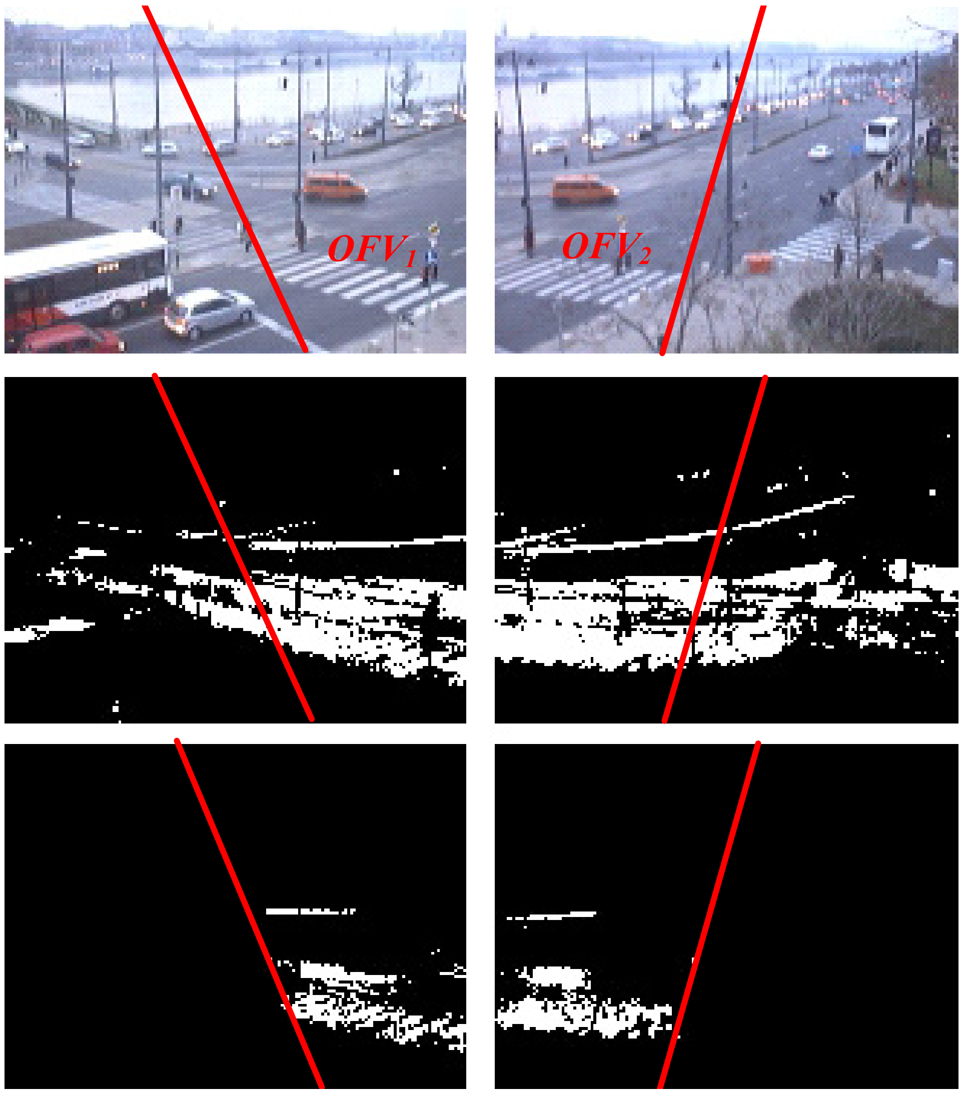


Figure 2-13. Images show the resulting point sets of the feature extraction. First two images are samples from input videos. Below them there are the results of the *entropy (6) based preselection of feature points. Last two images show the result of Bayesian estimation of OFV. Red lines are the borderlines of real OFVs.**

The advantage of this Bayesian iteration can be shown when it is compared to the pure statistical information of the conditional probabilities. Here the correlated areas are estimated from the conditional probabilities (29) as a correlation measure of change histories of two points and the resulted $P(m_{ij})$ probabilities.

We compare the relative impact of the conditional probabilities $\frac{\sum_{m_{1i} \in OFV_1} \max_k P(m_{1i} \mid m_{2k})}{\sum_i \max_k P(m_{1i} \mid m_{2k})}$

against the relative impact of feature points after the Bayes iterations $\frac{\sum_{m_{1i} \in OFV_1} P(m_{1i})}{\sum_i P(m_{1i})}$. The relative impact qualifies the distribution of extracted feature points: the higher the value of the impact the more of the feature points are from the true OFV. The change of the impact of feature points from common area can be seen in Table 2-4.

Table 2-4. The change of the relative impact (in %) of feature points within the estimated OFV areas relative to the whole image before and after Bayesian iteration.

	Impact of feature points from OFV (%)	
	Before	After
video1	51	81
video2	59	78
video3	37	70
video4	32	75

Table 2-5. The proportion of estimated OFV points to the real OFV points after correlation based selection and Bayesian iteration.

	Correlation	Bayesian
video1	71%	98%
video2	68%	99%
video3	73%	93%
video4	72%	90%

We have also compared the number of true OFV points after Bayesian estimation of common areas and after selecting the first N_f most significant feature points according to their "correlation" (29) value, where N_f is the number of extracted feature points by the Bayesian iteration. Denoting the number of real OFV points among them by N_{OFV} the precision of the estimation can be estimated as $\frac{N_{OFV}}{N_f}*100$. In the correlation based method the N_f most significant feature points were selected according to eq. 7. Table 2-5 shows the change of this ratio after correlation based selection and Bayesian iteration. Table 2-4 and Table 2-5 clearly show that after the iterations we can take a useful estimation about the OFV.

The above Bayesian approach gives a good estimation for the OFVs without any further processing. However, to fit the different views exact point-pairs are needed.

2.5.4 Extraction of candidate corresponding points

Having detected the feature-points in both views, for the extraction of candidate corresponding points the feature-points of the different views must be compared. For the comparison of feature-points (the corresponding motion-history vectors in our case), we have used Kulback-Leibler divergence (KLD) (39). For recognition, KL distance proved to be a usable measure in similar recognition cases [51]. When KLD (39) is minimized over b (40) the result of the estimation is equivalent to that of the maximum-likelihood method [27].

$$KLD(a,b)=\sum_i a_i \log(\frac{a_i}{b_i}) , \tag{39}$$

where *a* and *b* are the compared distributions. Thus, a feature point m_{2k} is a matching point for m_{1i} if (40) holds:

$$k=\arg\min_j KLD(\underline{\hat{c}}_{1i},\underline{\hat{c}}_{2j}) , \tag{40}$$

It is obvious that if all of our candidate corresponding points are from the close neighborhood of the input images, and thus are close to each other, then small errors in the coordinates of the points (which arise from the change detection, which is of course not perfect) will result in large errors in the final alignment of the whole images. To mitigate this problem, we force points to be better distributed in the region by introducing some structural constraints: images are divided into blocks of $n \times n$ pixels (*n* depends on the resolution) and for each block the algorithm searches for only one candidate point-pair with the minimum divergence. In our case n takes a value between *4* and *9*: for greater windows feature extraction could result in undersampling for model fitting.

2.5.5 Estimation of models and rejection of outliers

For planar scenes the homography H is a projective transformation while in case of 3D motions the fundamental matrix F relates the input images. The rejection of outliers with respect to the models and robust estimation of the homography and fundamental matrix was performed by the RANSAC algorithm [31].

2.5.6 Shadow based registration of indoor images

As it was mentioned before in introduction (Chapter 2.1) the existing algorithms can be used only in restricted situations. Usually, existing methods focus on the solution of the view-registration problem in respect of outdoor scenes for a given plane where points are assigned, and neglect the additional difficulties, which tend to arise for indoor scenes. These methods assume that the objects of interest are moving on the ground plane and that the cameras are far distant from the scene, so that the height of the moving objects is small enough for them to appear in the recorded videos as “moving blobs on the ground”, as seen in Figure 2-14 (right). In case of indoor cameras, see Figure 2-14 , due to the larger size of the moving objects, the existing motion-based methods will also fail as the observed motions are not necessarily on the ground-plane.

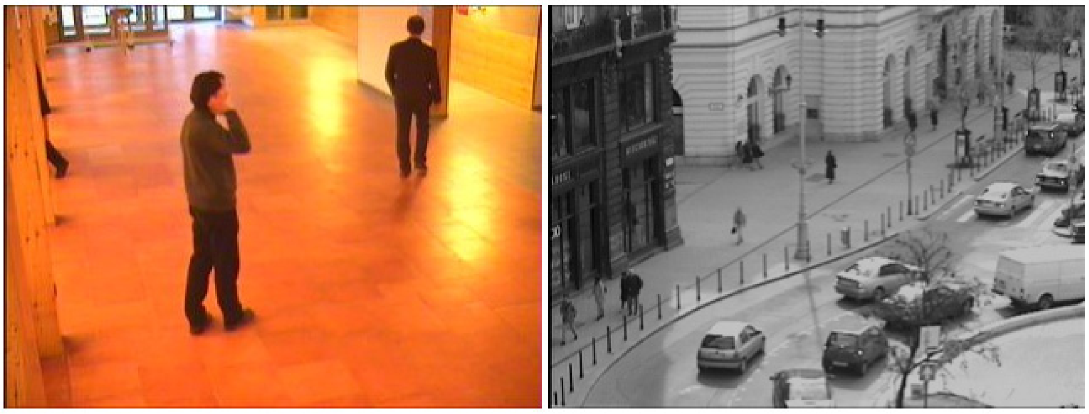


Figure 2-14. Examples of indoor (left image) and outdoor (right image) scenes

Thus, in indoor situations detecting objects’ motion is not useful for ground plane registration. We are only interested in changes on the ground. Shadows are usually on the ground plane, thus they are excellent features for our algorithm to analyse. Here the input is the output of a shadow detector instead of a change detector.

As shadow detector we used a modified ‘Sakbot’ method [28][24]. It works in the HSV color space, which corresponds closely to the human perception of color. The method exploits that the occlusion of cast shadow darkens the background pixel and does not change significantly its colour. As it can be observed in Figure 2-15 c) and d), the shadow pixels were found well on the images. However there were false positive shadow occurrences especially around the body, so the shadow mask used in section 4.3 was filtered by an erosion step, see Figure 2-15 e).

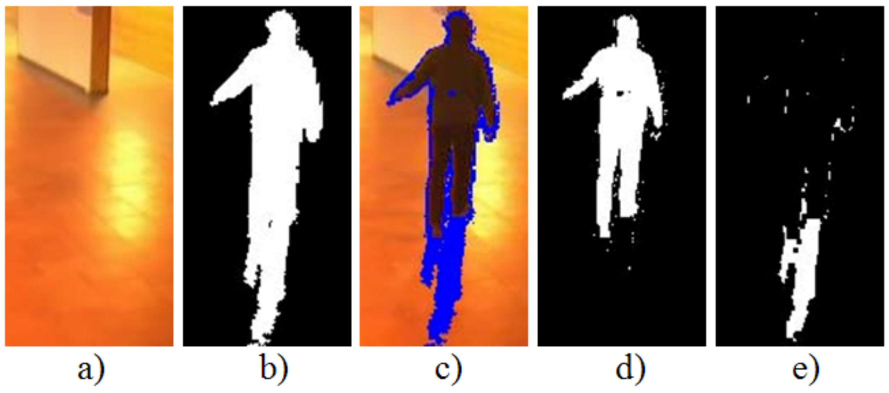


Figure 2-15. a) Background image and b) Silhouette image c) Result of shadow detection (blue color marks shadow points) d) Object silhouette after removing the shadows e) Filtered shadow mask used for camera registration

Further steps of the shadow based registration method are the same as the steps of the motion based method of registration. The block diagram of the method is shown in Figure 2-16.

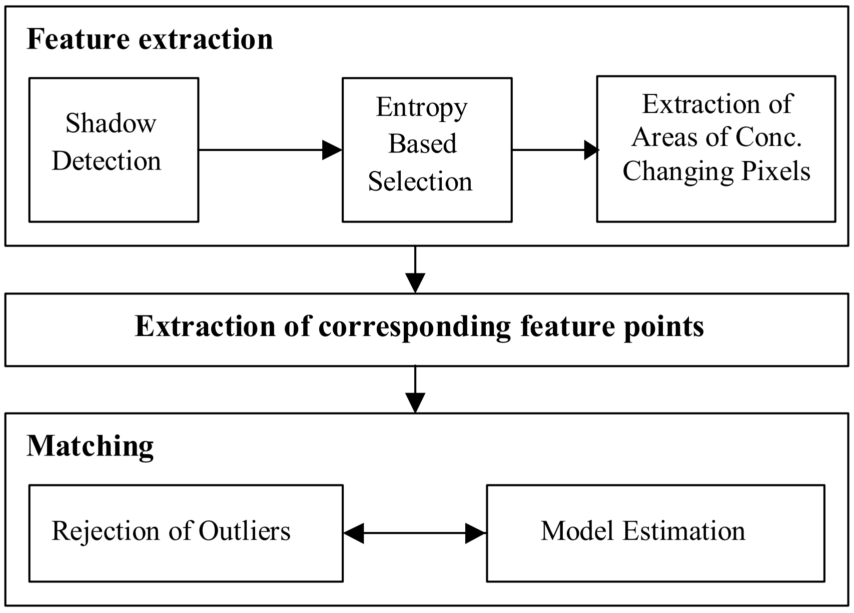


Figure 2-16. Shadow based registration of images

2.5.7 Experimental results

Initially, the aim of the proposed method was to register two overlapping views based on the evaluation of observed motions. So, the algorithm is unable to register image regions where no motion was observed. Usually the observed objects (people, car etc.) are moving on the ground plane, when homography was used to register the two views (see section V/A and V/B). Also, if the observed motions are in 3D, the proposed algorithm still performs well by estimating the fundamental matrix for the registration of the two views.

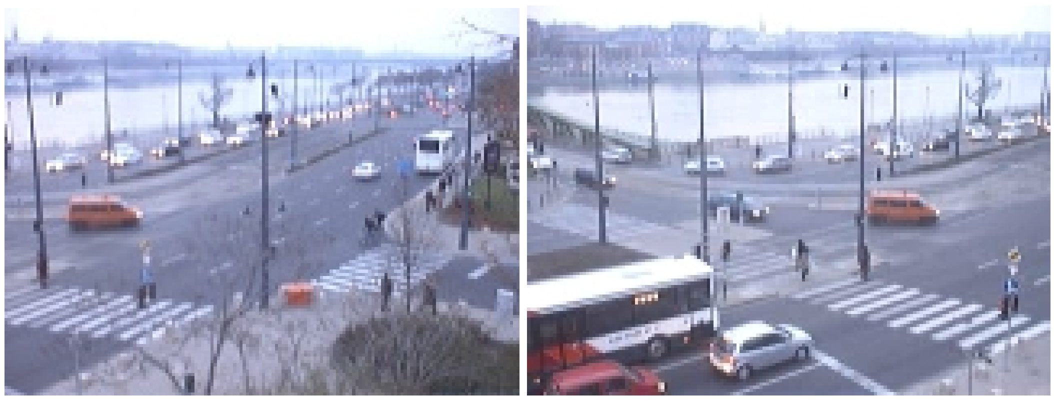
In our experiments we have used the following algorithm:

1. Detection of changes;
2. On-line entropy based preselection of features;
3. Bayesian estimation of *OFV;*
4. Finding point correspondences according to (40);
5. Robust model fitting and rejection of outliers by using the RANSAC algorithm.

For testing of our method's performance we have captured several videos with different cameras and at different conditions. The videos were captured at public places and at the university campus of Peter Pazmany Catholic University.

2.5.7.1 Experiments with outdoor videos

The above-described algorithms were tested and compared on videos captured by two cameras, having partially overlapping views, at Gellert square (GELLERT videos) in Budapest. The GELLERT videos are captured at resolution 160×120, at same zoom level and with identical cameras.



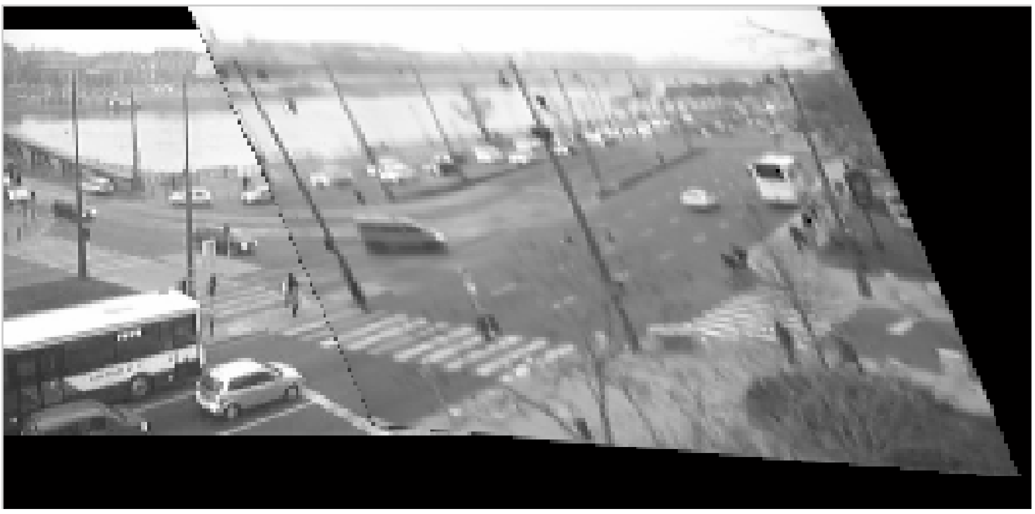
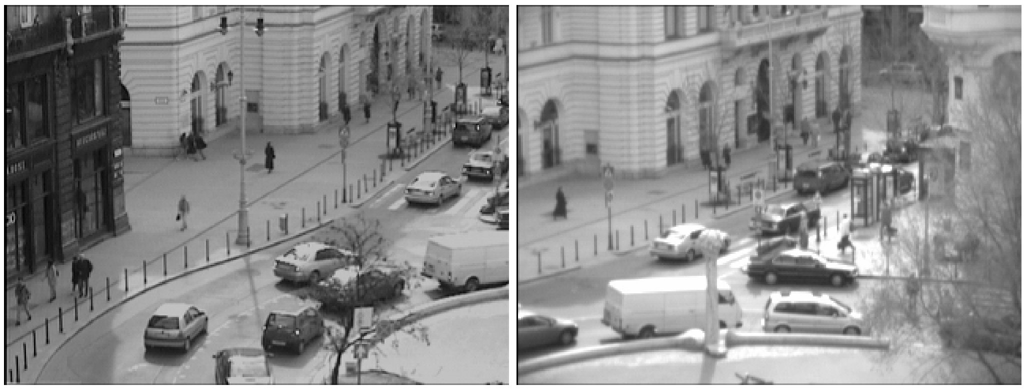


Figure 2-17. Final alignment of two views with transformation T2 for the GELLERT videos.

The result of final alignment is shown in Figure 2-17. Figure 2-18 shows the results of final alignment for the FERENCIEK videos. FERENCIEK videos are captured at Ferenciek square in Budapest at resolution 320×240, at different zoom levels and with different cameras.



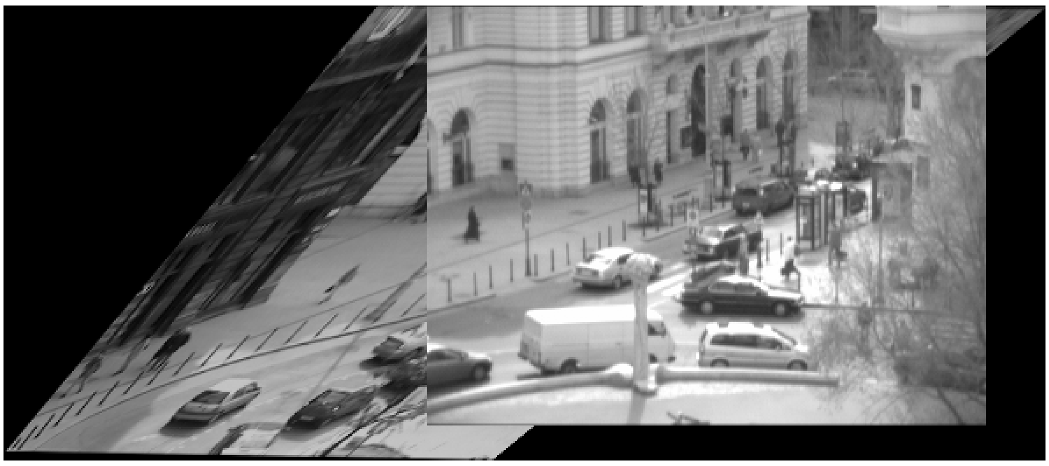


Figure 2-18. Final alignment of two views for the FERENCIEK videos.

We have also tested our algorithms on videos from PETS2001 dataset. The results of alignment can be seen in Figure 2-19.

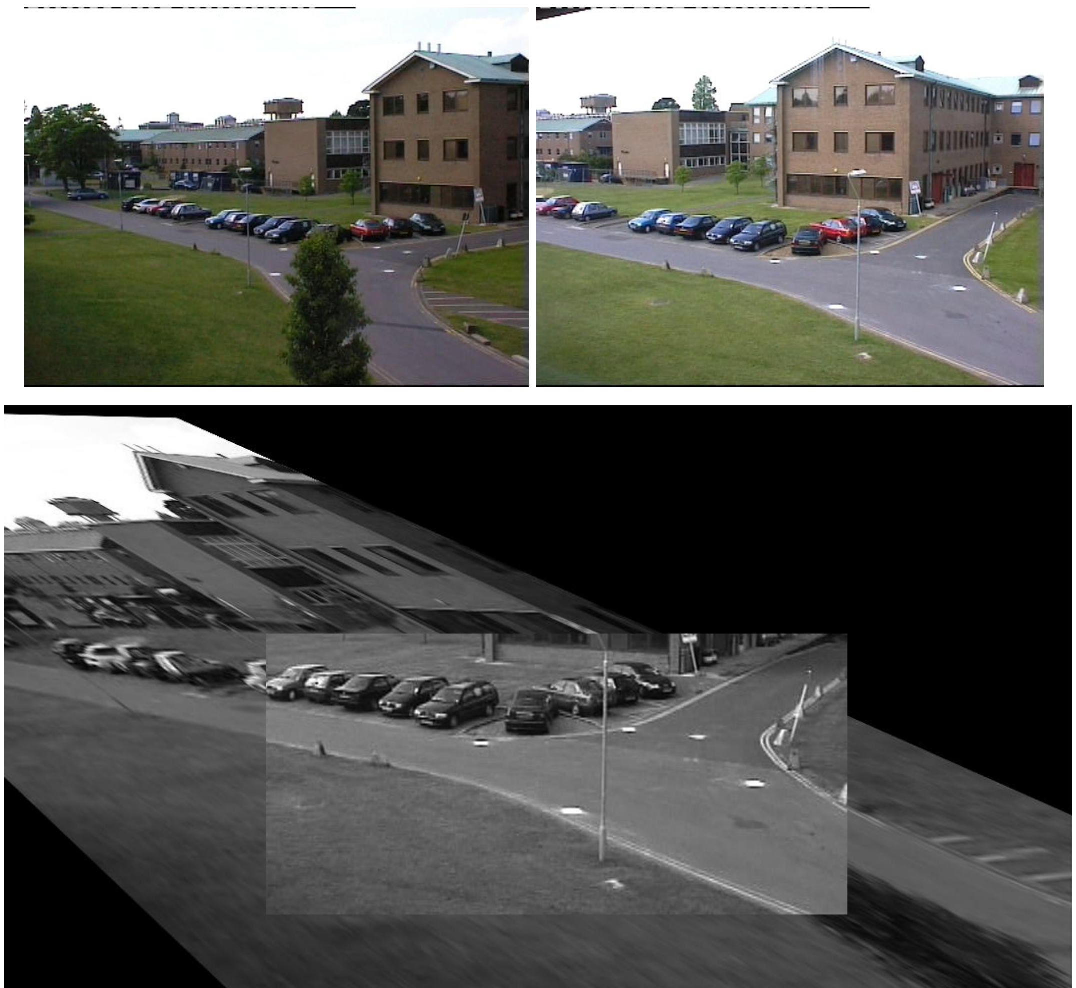


Figure 2-19. Final alignment of views from PETS2001 dataset.

2.5.7.2 Experiments with indoor videos

For the testing the proposed shadow based registration method we used indoor videos captured at the university campus of Peter Pazmany Catholic University. Figure 2-20 shows the results of alignment.

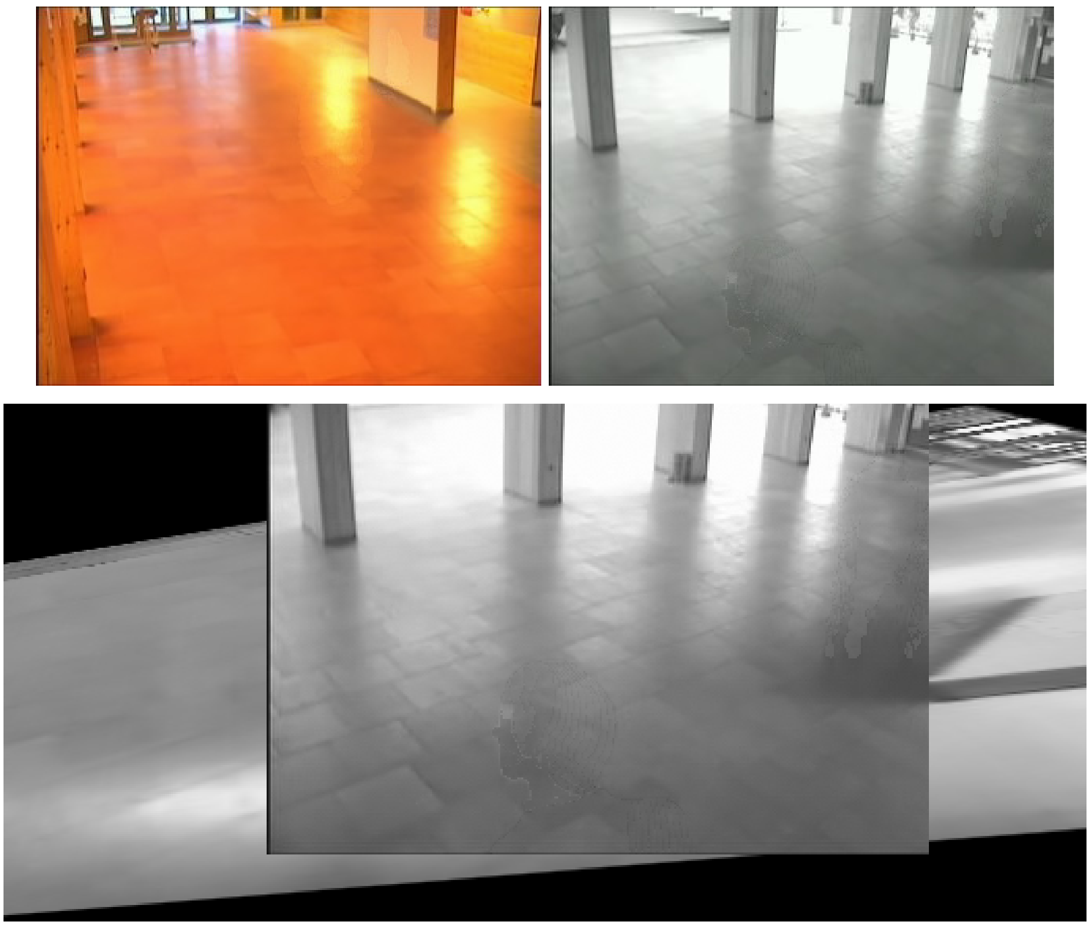


Figure 2-20. Alignment of views based on the detection of concurrently moving shadows.

Finally, in order to demonstrate that unpredictable random 3D motion can be used for extraction of point correspondences, we set up the following experiment. Small trees were blown with an artificial wind emanating from a ventilator fan which performed a periodical scanning movement (back and forth), and the generated motion of the leaves was recorded with two cameras at resolution of 320×240, at the same zoom setting (video 4: LAB video). The estimated epipolar geometry can be seen in Figure 2-21.

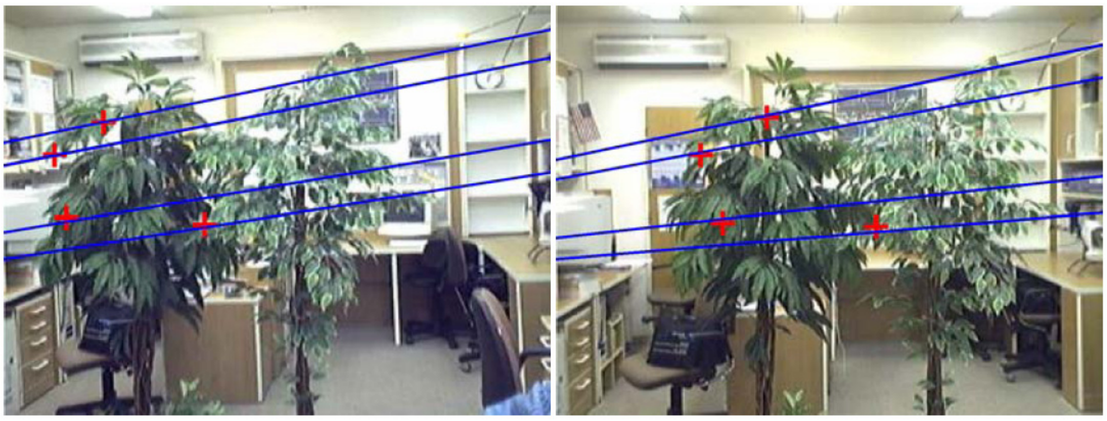


Figure 2-21. Sample epipolar lines obtained for the LAB videos.

2.5.8 Evaluation of results

Table 2-6 summarizes the numerical results of model fitting for different video pairs. The error of alignment was calculated according to formulas (9) and (10) on manually selected control points. The minimum error owes to regions where corresponding points were detected. It can be clearly seen that the minimum error is in the subpixel range for both models. Then, model selection in a practical application can be performed as follows. First, the algorithm tries to relate the two input images with a homography. If the minimum error is significantly larger than one pixel that means that the observed motions are in 3D and the model that relates the two images should be the fundamental matrix. Fitting a homography to the images of LAB videos, the minimum error of alignment was 12.3 pixel, so the algorithm switched to calculate a fundamental matrix which was fitted with minimum error 0.16 and average error 6.15 pixels.

Table 2-6. Numerical results of model fitting for different experiments

Experiment	Average Error	Min Error	Model
Gellert	5.40	0.16	H
Ferenciek	6.54	0.44	H
PETS2001	4.18	0.88	H
Indoor	9.00	0.63	H
LAB	6.15	0.16	F
LAB	21.22	12.30	H

The obtained results of Table 2-6 look promising, especially if we take into account that the proposed algorithm does not use any *a priori* information (shape, motion etc.)

about observed objects. It can hardly be compared to other motion based methods described in [25][36][37][39][42] as they are aligning tracks but not views. Qualitative results shown in Figure 2-17- Figure 2-19 and quantitative results of Table 2-6 prove the feasibility of our approach with respect to other methods. But, our method also can be used in cases when no tracking is possible because of the complexity of observed motions (see Figure 2-20 and Figure 2-21).

3 Facial feature extraction by using CNN

An essential step for face detection, face recognition, facial expression recognition is the extraction of facial features. This process is the basis of feature based face processing algorithms and usually involves the detection of facial features (eyes, nose, mouth), their relative position.

In this chapter we describe analogic algorithms for the detection of facial features. The algorithmic framework is the CNN Universal Machine. CNN is a natural tool for image processing tasks because of its regular two-dimensional structure and fast execution.

3.1 Introduction

Reliable face analysis is very important for many applications such as human-computer interfacing [64] or recognition of face and facial expression either from still images [72][76][19] or image sequences [80][85]. Several face detection and feature extraction methods were proposed in the field of face processing in the past years [58][62][84][83]. The methods may be categorized into two classes: methods based on template-matching [84], and those based on geometrical analysis methods [56][58][71][85]. An interesting comparison between the two techniques is given in [57]. Geometric or feature-based matching algorithms aim to find structural features that exist even when the pose, viewpoint, or lighting conditions vary, and then to use these to locate a face or faces. The idea is to extract relative position and other parameters of distinctive features such as eyes, mouth, nose etc. In the simplest version of template matching, the image, which is represented as a 2-dimensional array of intensity values, is compared using a suitable metric with a single template representing the whole face. Patterns of facial features also can be stored as templates and the correlations between an input image and the stored patterns are computed for detection.

Relatively few results have been published on face analysis from video images [80][85]. The reliability (recognition rate) in the existing approaches is very high – in [85] it is nearly 100%; but a common disadvantage of them is the time-consuming nature of the algorithms: 16s per frame (frame size 350×240 pixels) for a face-recognition task in [85], and in another case [80], 24s per proportion of the face in the image (image size 352×288 pixels) for a face-detection task. This slow response means that such methods are impractical for real-time processing of faces, for example in a visual surveillance system. Many results are known for face detection and recognition in still images. Rowley's neural network based algorithm for face detection requires just half second for 320*240 images [75]. A more recent result is Viola and Jones's algorithm for face detection, which can process 15 images (image size 384×288 pixels) per second in average (the performance depends on the number of features) [81].

The algorithms we propose were developed for use with a Cellular Visual Microprocessor (CVM), which process images in a pixel parallel way. The CVM [60][66] has proved to be an appropriate tool for accomplishing real-time image processing [59], as well as for real-time classification [77] and segmentation [78].
The input of our system is a complete face contained within the image-frame. We assume the face to have been already detected by using skin-color based segmentation [62][83]. The morphological approach for facial-part detection is not an established procedure; however, here we attempt to explore and demonstrate its possible merits. Morphological operations have previously been applied for face shape extraction [82] and for extraction of darker regions in the face image [61]. Previous papers have also demonstrated how to use morphological operations for the pre-processing of images, for detecting features contained within them [61][77][82].
In the proposed algorithm, first a geometrical face model is defined. Based on this, the structure of the face is analyzed by binary morphology. Later we demonstrate that by means of this essential information (i.e. the *x, y* coordinates of the nose and eyes), primary feature extraction is possible. The algorithms were tested using input images from the face-image database developed by UMIST (the University of Manchester Institute of Science and Technology) [65].

3.2 Cellular neural networks

Since their invention in 1988 by Leon O. Chua Cellular Neural Networks (CNN) has become a widely used popular tool for solving different problems. Examples are simulation of partial differential equations, simulation of artificial immune systems, modelling biological systems. The most popular application of the CNNs is the processing of images. Because of the CNN's regular structure an image easily can be mapped onto CNN and then the processing of the image is just "playing" with CNN's control parameters.

3.2.1 The CNN Paradigm

A CNN is a collection of cells defined by:

- The dynamics of each cell;
- The local couplings among cells within a prescribed sphere of influence $N_{r(i,j)}$ of radius r centered at (i,j):

  $N_r(i,j) = \{C_{kl} \mid \max_{1\le k\le M, 1\le l\le N} \{|k-i|, |l-j|\} \le r\}$, where C_{kl} is the k-th cell in the l-th row.

Figure 3-1 shows a 2D rectangular CNN composed of cells that are connected to their nearest neighbors.

A regular cell is a cell whose r-neighborhood does not consist of boundary pixels. Boundary conditions are very important in the design of CNNs. A few standard boundary conditions are listed as follows:

- Fixed (Dirichlet) Boundary Conditions. The input and output of a boundary cell are fixed to constant values.
- Zero-Flux (Neumann) Boundary Conditions. A boundary cell and its symmetric (with respect to the boundary) regular counterpart share the same input and output.
- Periodic (Toroidal) Boundary Conditions. By sticking one pair of boundaries side by side and then sticking the two opening ends of the resulting cylinder end to end, we get a torus on whose surface the cells are relocated.

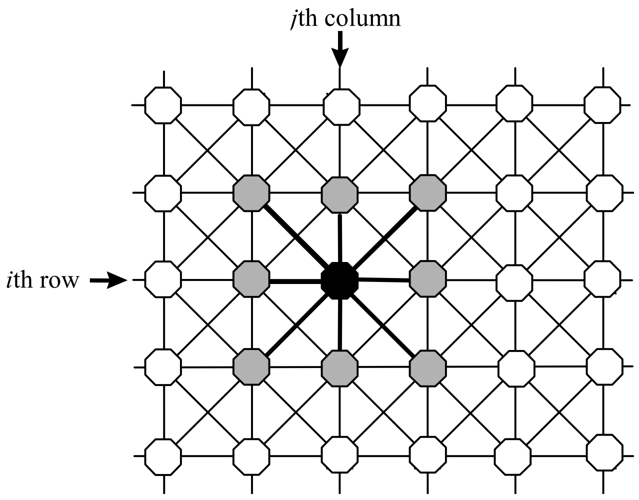


Figure 3-1. A 2-dimensional CNN defined on a square grid. The *ij*-th cell of the array is colored by black, cells that fall within the sphere of influence of neighborhood radius *r* = 1 (the nearest neighbors) by gray.

Due to its symmetry, regular structure and simplicity this type of arrangement (a 2D rectangular grid) is primarily considered in all implementations.

3.2.2 Mathematical formulation

As the basic framework for the experiments in this chapter, let us consider a two dimensional (*MxN*) standard CNN array in which the cell dynamics is described by the following nonlinear ordinary differential equation with linear and nonlinear terms (the extension to three dimensions is straightforward allowing similar interlayer interactions):

$$C\frac{d}{dt}x_{ij}(t) = -R^{-1}x_{ij}(t) + \sum_{kl\in N_r} A_{ij,kl}\,y_{kl}(t) + \sum_{kl\in N_r} B_{ij,kl}\,u_{kl} + z_{ij}(t) \qquad (41)$$

$$y_{ij}(t) = f(x_{ij}(t)) = 0.5\,(|x_{ij}(t)+1| - |x_{ij}(t)-1|)$$

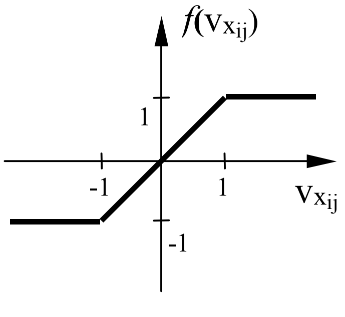


Figure 3-2. The output function of a CNN cell.

where x_{ij}, u_{ij}, y_{ij} are the state, input and output of the specified CNN cell, respectively. The state and output vary in time, the input is static (time independent), ij refers to a grid point associated with a cell on the 2D grid, and $kl \in N_r$ is a grid point in the neighborhood within the radius r of the cell ij. Term $A_{ij,kl}$ represents the linear feedback, $B_{ij,kl}$ the linear control, while z_{ij} is the cell current (also referred to as bias or threshold) which could be space and time variant. The output characteristic f is a sigmoid-type piecewise-linear function. The time constant of a CNN cell is determined by the linear capacitor (C) and the linear resistor (R) and it can be expressed as $\tau=RC$. A CNN cloning template, the program of the CNN array, is given with the linear terms $A_{ij,kl}$ and $B_{ij,kl}$ completed by the cell current z_{ij}.

3.2.3 Image processing with CNN

During the last decade numerous templates have been proposed for various image-processing tasks: filtering, feature extraction, segmentation etc. [59][63][77][78][79]. Here however we describe only those image-processing template operations that are used in our proposed algorithms. Some of the template operations are direction-dependent; the direction of the operations is "encoded" in the template values; and by "rotating" the template the operation will be performed in different directions. The Connected Component Detector detects the number of connected components in a given direction [67]. Also direction-dependent are the Shadow operation [68], and the Local Eastern Element Detector template operation [63]. A very interesting and important class of CNNs is the propagating CNNs. In these CNNs the information, like a wave, is propagating from cell to cell along the whole grid or until a given stop criterion. Examples of such CNNs are the Connected Component Detector, Shadow and Recall CNNs. In many image-processing tasks it is very useful to be able to reconstruct objects in the image, and this can be performed by the Recall template operation [63]. Logical operations can be performed also [70], as well as morphological operations [86]. The values of the templates are in Table 3-1 and the corresponding inputs and outputs of the template operations are shown in Figure 3-3 to Figure 3-6.

Table 3-1. The values of feedback, input synaptic operators and bias for different templates

Name					A									B					z
	a_{-1-1}	a_{-10}	a_{-11}	a_{0-1}	a_{00}	a_{01}	a_{1-1}	a_{10}	a_{11}	b_{-1-1}	b_{-10}	b_{-11}	b_{0-1}	b_{00}	b_{01}	b_{1-1}	b_{10}	b_{11}	
CCD	1	0	0	0	2	0	0	0	-1	0	0	0	0	0	0	0	0	0	0
Local E.-elt. Det.	0	0	0	0	1	0	0	0	0	0	0	0	1	1	-1	0	0	0	-2
Shadow	0	0	0	0	2	2	0	0	0	0	0	0	0	2	0	0	0	0	0
Recall	.5	.5	.5	.5	4	.5	.5	.5	.5	0	0	0	0	4	0	0	0	0	3
Logic AND	0	0	0	0	2	0	0	0	0	0	0	0	0	1	0	0	0	0	-1
Erosion	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	-2
Dilation	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	2

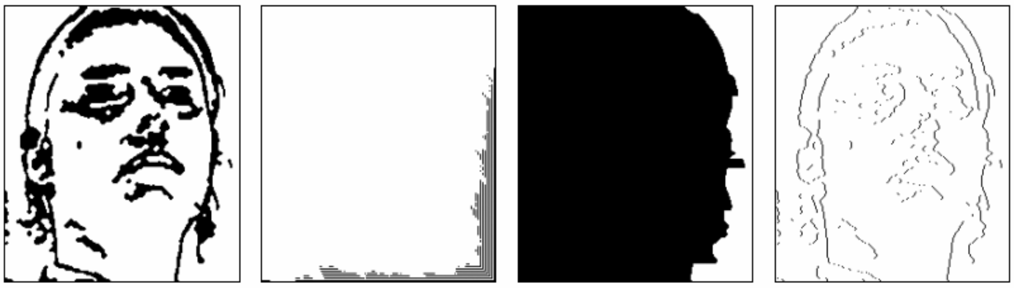


Figure 3-3. The input and outputs of the diagonal Connected Component Detector (CCD), Shadow, and Local Eastern-element Detector (Local E.-elt. Det.) template operations respectively

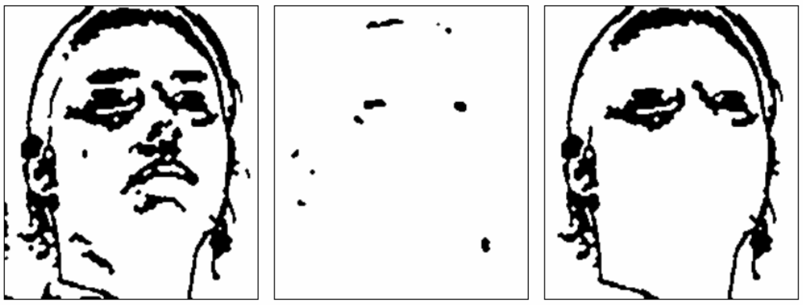


Figure 3-4. The input, initial state and output of the CNN for the Recall/Reconstruction operation

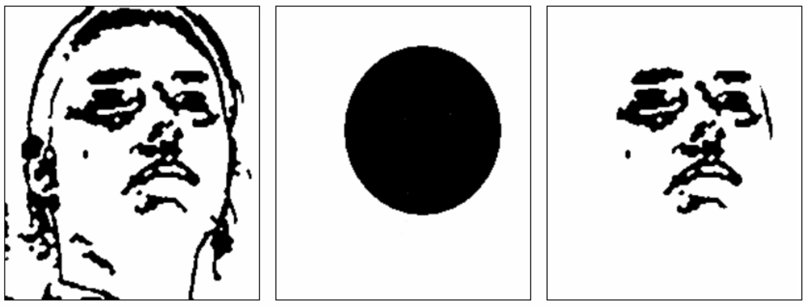


Figure 3-5. The input, initial state and output of the CNN Logic AND template operation

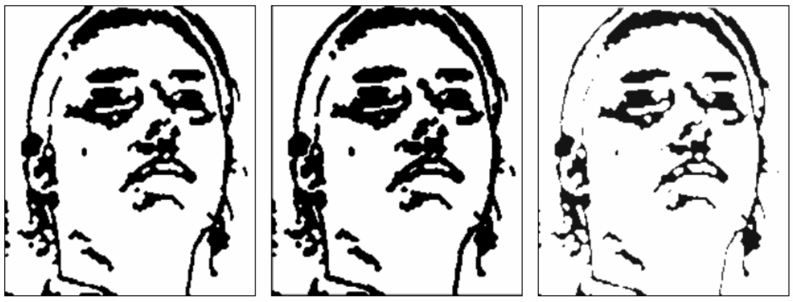


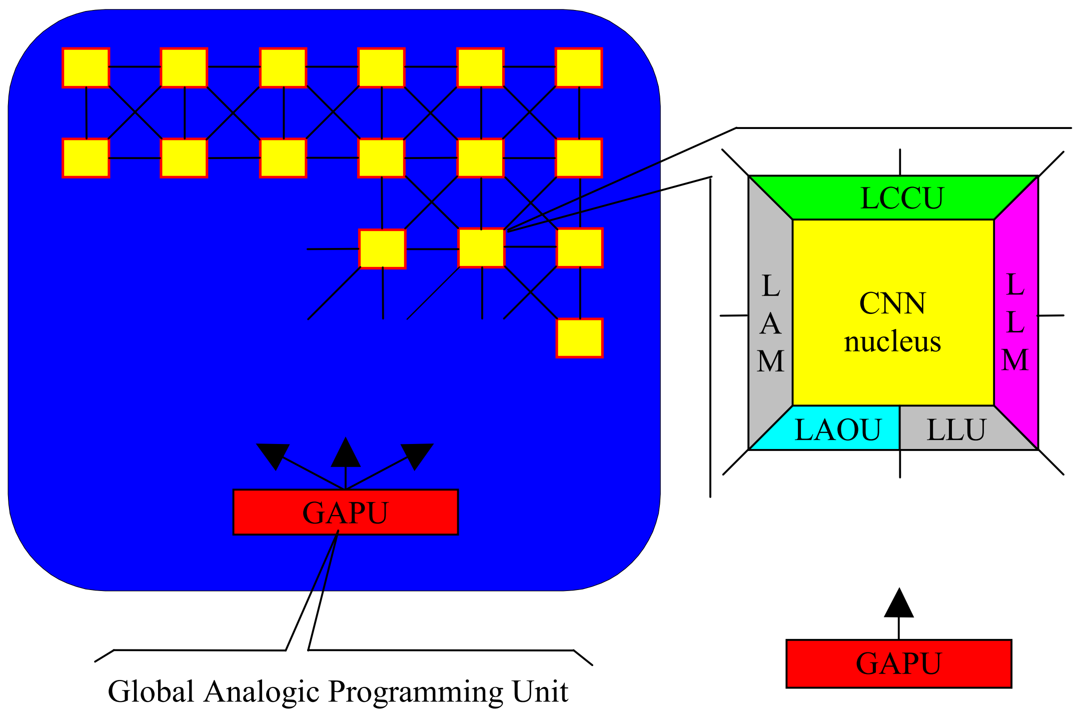
Figure 3-6. The input and outputs of Dilation and Erosion template operations

Generally, the detection of objects in an image necessitates a great deal of computation. A natural way to reduce the number of operations is to execute them in parallel, or to design parallel algorithms. Image-processing parallel algorithms can be interpreted as the simultaneous processing of each pixel and its neighborhood. The type of algorithm, which is provided by a CVM, a parallel operation on the whole image, is typically executed in a few microseconds. By using templates, different image-processing, morphological and wave-metric operations can be implemented [59][63]. Sophisticated complex tasks, such as texture segmentation or image segmentation employing a CNN, are described in [78] and [79].

3.2.4 The CNN Universal Machine

The design of the CNN Universal Machine (CNN-UM, [59]) was motivated by the need of a reprogrammable CNN device. All earlier chip realizations of CNN allow a single instruction only and the programming of algorithms at chip level was impossible.

CNN-UM, the stored program nonlinear array computer, is able to combine analog array instructions with local logic operations, it is capable of executing complex analogic (analog and logic) algorithms.



APR	global Analog Program Register
LPR	global Logic Program Register
SCR	Switch Configuration Register
GACU	Global Analogic Control Unit

LCCU	Local Communication and Control Unit
LAOU	Local Analog Output Unit
LLU	Local Logic Unit
LAM	Local Analog Memory
LLM	Local Logic Memory

Figure 3-7. The CNN-UM.

As illustrated in Figure 3-7 the CNN-UM was built around the dynamic computing core of simple CNN. Local memories store analog (LAM: Local Analog Memory) and logic (LLM: Local Logic Memory) values in each cell. A Local Analog Output unit and a Local Logic Unit perform cell-wise analog and logic instructions on stored values. The output is always transferred to one of the local memories. The Local Communication and Control Unit (LCCU) provides for communication between the extended cell and the central programming unit of the machine, the Global Analogic

Programming Unit (GAPU). The GAPU has four functional blocks. The Analog Program Register (APR) stores the analog program instructions, the CNN templates. In case of linear templates, for a connectivity $r = 1$ a set of 19 real numbers have to be stored (this is even less for both linear and nonlinear templates assuming spatial symmetry and isotropy). All other units within the GAPU are logic registers containing the control codes for operating the cell array. The Local Program Register (LPR) contains control sequences for the individual cell's LLU, the Switch Configuration Register (SCR) stores the codes to initiate the different switch configurations when accessing the different functional units (e.g. whether to run a linear or nonlinear template). The Global Analogic Control Unit (GACU) stores the instruction sequence of the main (analogic) program. The GACU also controls the timing, sequence of instructions and data transfers on the chip and synchronizes the communication with any external controlling device.

Synthesizing an analogic algorithm running on the CNN-UM the designer should decompose the solution in a sequence of analog and logical operations. A limited number of intermediate results can be locally stored and combined. Some of these outputs can be used as a bias map (space variant current) or fixed-state map (space-variant mask) in the next operation adding spatial adaptivity to the algorithms without introducing complicated inter-cell couplings. Analog operations are defined by either a linear or a nonlinear template. The output can be defined both in fixed and non-fixed state of the network (equilibrium and non-equilibrium computing) depending on the control of the transient length. It can be assumed that elementary logical (NOT, AND, OR, *etc.*) and arithmetical (ADD, SUB) operations are implemented and can be used on the cell level between LLM and LAM locations, respectively. In addition data transfer and conversion can be performed between LAMs and LLMs.

3.2.5 Cellular Visual Microprocessors and the Aladdin System

The Aladdin system is the high-performance, professional computational environment of the ACE4k chip [60]. The block diagram, which shows the internal logic arrangement and the interconnection system of the Aladdin system, can be seen in Figure 3-8. The DSP module is connected to the motherboard of the host PC via the PCI bus. The PCI bus bridges the Aladdin system to standard frame-grabbers. The output images (if there is any) leave the system also through the PCI bus and can be displayed on the display of the PC. In many cases, only decisions or a couple of identified events are read out from the system, or stored in the hard drive, and there is no need to send or store original or processed images.

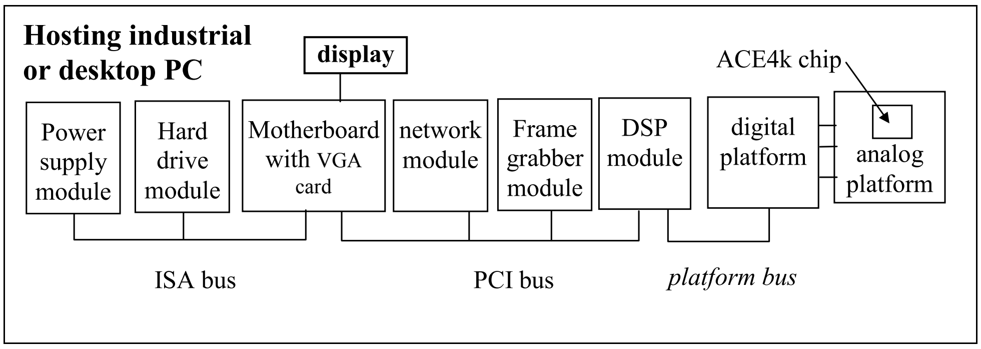


Figure 3-8. The block diagram of the Aladdin visual computer

Either a desktop or an industrial PC may host the DSP module. The operating system on the PC can be either Windows NT or 2000. Since the DSP module uses a number of resources of the PC, a hosting program is needed (called CNNRUN) to provide the appropriate services. On the DSP module, another program, called CNN Operating System (COS), runs. This program drives the platforms and the ACE4k chip. The data transfer speeds and some general processing speed figures can be found in Table 3-2.

Table 3-2. The measured operation speeds of the system

Function	*speed*
64×64 grayscale image transfer	2,300 fps
64×64 binary image transfer	22 000 fps
Pixel-by-pixel logic operation on a 64×64 binary image	3.8 μ
Template operation on a 64×64 binary or grayscale image	3-15 μs
Processing of 64×64 sized images (grayscale input, binary output)	1500 fps
Processing of 64×64 sized images (binary input, binary output)	5 000 fps

At our laboratory an image processing library was developed for ACE4K visual microprocessor [60]. User can call them from the native AMC language [12] of the Aladdin Visual Computer and from C language also. The template values, transient time settings, reference values, etc. are all hidden from the users. Users have to parameterize the functions in the top level only, like setting threshold levels, describing the structuring element set of a morphological operation, etc. The following list contains all image processing functions in the library (DSP functions are in italic type set and ACE4K operations are in regular):

1. Basic data movement routines:
    a. *Image transfer and type conversion from/to/inside the chip (e.g. threshold)*
    b. *Full PC frame grabber card support*
2. Pixel-wise image handling:
    a. *Logic operations*
    b. *Addition, subtraction*, multiplication, division
    c. *Binary and gray-scale translation*
3. General statistics:
    a. *Histogram calculation*
    b. *Mean, variance, median, global maximum, minimum*
4. Image features, extraction:
    a. *Mathematical morphology (dilation, erosion, skeleton, prune, thicken, SKIZ, fill, shrink, pattern matching, single pixel removal, reconstruction, ...)*
    b. *Classic functions (edge, shadow, whole filler, patch maker, concave place detection, perimeter structural operator, center-of-mass, majority, ...)*
    c. *Connected component labeling (object information: area, solidity, bounding box, orientation, convex area, ordered contour, ...)*
5. Image filtering:
    a. *Contrast stretching,* histogram equalization
    b. *Linear and adaptive edge enhancement, Laplacian, Gaussian, Sobel, Correlation, Adaptive diffusion, Gradient, Thresholded gradient*
6. Transformations:
    a. *Forward DCT, inverse DCT, wavelet processing, FFT-1, FFT-2, Haar, Hadamard, and Radon, JPEG compression, matrix quantization with rounding*
7. Image synthesis:

   a. *Patterns (e.g. checker, stripes), noise, gradients*

8. Signal processing:
   a. *FIR, IIR filters, Autocorrelation, weighted vector sum, matrix operations*

The library contains over 40 optimized binary image processing functions. The implementation of the operators are based on reprogrammed switch configurations. Moreover, the self and the spatial feedback loops are fully neglected resulting in higher robustness and significantly reduced settling time in binary output generation [14]. Therefore, the propagation type operations are implemented iteratively.

Table 3-3 summarizes the execution times of the classic (non-decomposed) CNN solution, the library functions and the DSP solution as a comparison. As it can be seen, the library functions are not faster all the time, however they provide error-free results all the time. If we compare the execution times of the library function to those of a state-of-the-art DSP, it turns out that the ACE4K chip has almost an order of magnitude speed advantage. We have to mention here, that the DSP functions are optimized assembler functions provided by the manufacturer.

Table 3-3. Running time comparison for some typical operations between the non-decomposed CNN-type single template executions and the library solution in case of chip-sized images. As a comparison, we also show the execution times with a state-of-the-art DSP with its optimized assembly code. All images are 64x64 sized. The displayed times include on-chip data transfer and processing only, which means that all the images are stored in internal memories of the ACE4K and the DSP.

Operator	Non-decomposed CNN-solution (not error free)	Library functions (error-free)	DSP solution (Texas C6202 @ 250MHz) (error-free)
Binary edge detection of 4-nbr connectivity	7 μs	4.1 μs	31μs
Binary edge detection of 8-nbr connectivity	7 μs	6.7 μs	58μs
Reconstruction using 4- nbr connectivity	Initialization: 3 μs Propagation time per pixel: 1.1 μs	Initialization: 3 μs Propagation time per pixel: 0.65 μs	Propagation time per pixel: 31μs
Reconstruction using 8- nbr connectivity	Initialization: 3 μs Propagation time per pixel: 1.2 μs	Initialization: 3 μs Propagation time per pixel: 1.5 μs	Propagation time per pixel: 58μs
Skeletonization of 8-nbr connectivity	171 μs	61.7 μs	464 μs

All the algorithms presented later are developed by using equivalent operations of the Aladdin System's image processing library to the template operations that are collected in [63]. So, we are not using templates but the image processing library for ACE4K in which the template values, transient time settings, reference values, etc. are all hidden from the users. In the description of the algorithms the name of the used templates are only references to the equivalent image processing library operation because the same operations have same names.

3.3 Facial feature extraction with CVM based operations

We now present our main contribution, the implementation of an efficient algorithm on real analog hardware. Facial features (as well as other features) could in principle be detected by calculating the vertical and horizontal intensity-gradients of the input image [55]. However we chose to design an algorithm employing binary-output operations, because these are more stable on analog hardware such as the ACE4K. Consequently, in our facial feature extraction system the input is the edge map of a face image, obtained by applying the CNN equivalent of the Sobel operator [63]. By analyzing the face image, a geometric face model can be defined [56].

3.3.1 The geometric face model

The extraction of primary facial features such as eyes, nose and mouth relies on the fact that the distance between the eyes is proportional to the distances between the facial features. Consequently, if one of the features is detected, the approximate position of the other features will be known and appropriate masks can be created to extract the features from the face image. Features extracted are the face symmetry axis, face width, nose, eyes and mouth. The geometric face model and the relationship between these measures are defined [56] as follows (see also Figure 3-9):

1. Let the distance between two eyes be D;
2. The vertical distance between the eyes and the center of the nostrils is $0.6\times D$;
3. The approximate face width is calculated as $1.8\times D$;
4. The width of the nose is about $0.6\times D$.

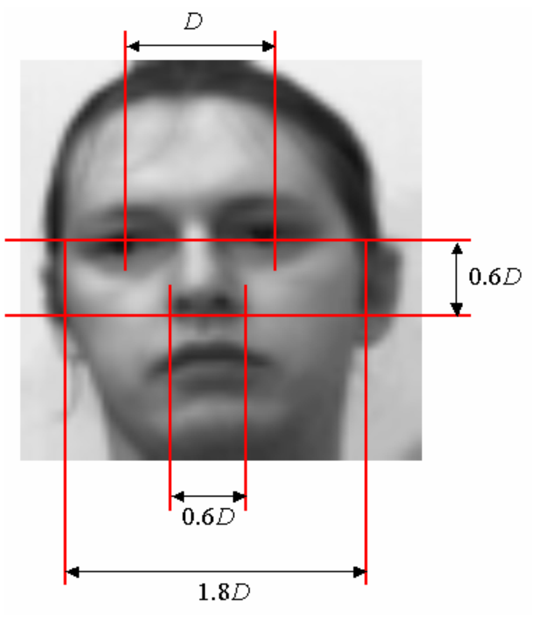


Figure 3-9. The defined geometry of the face model

3.3.2 Extracting facial features

The procedure for locating the facial features using the above-defined geometric face model is described below. First, the input image is filtered by a Gaussian filter template operation [63] to reduce the noise, and then edge detection is applied to the face image. Following this the face symmetry axis and the face width are detected as in [19] and then, based on the defined face model (Figure 3-9), the nose is extracted. By utilizing the geometrical relationship among the features, the eyes, and then finally the mouth, are extracted.

In the work described here we rely on CNN-friendly operations such as the projected histogram, the CCD operation of about 15μsec duration [67], which results in a histogram-like image of its input in either the X or Y direction (see Figure 3-10). The "peak" of the "histogram" in the resulting image corresponds to the most detailed region of the input image in the direction the operation is applied. The whole of this work is based on the principle that peaks on the resulting feature-images (i.e. the heights of the columns) refer to the respective vertical and horizontal positions of the specified facial features. Specifically, the region of the nose is the most detailed vertical feature; and the region of the eyes is the most detailed horizontal feature in the edge-map of the face image (see Figure 3-10).

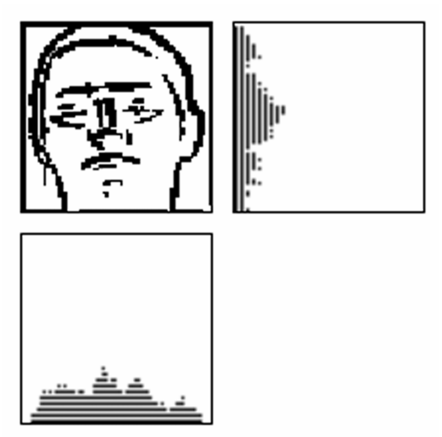


Figure 3-10. Application of CCD operation in directions West and South to the edge map of face image. The peaks in the resulted histogram like images refer to vertical and horizontal positions of the nose and the eyes.

To find the face symmetry axis, corresponding to the peak of the histogram-like image resulting from the CCD template operation in Figure 3-10, some typical template operations for binary images [19] are used (see Figure 3-11).

In [74] a valuable analytic tool which we make use of was introduced, namely UMF (Universal Machine Flows) diagrams, a description language for complex analogic algorithms. The basic principles of UMF diagrams are described below, in the Appendix. UMF diagrams clearly show how the template operations are combined in an analogic algorithm.

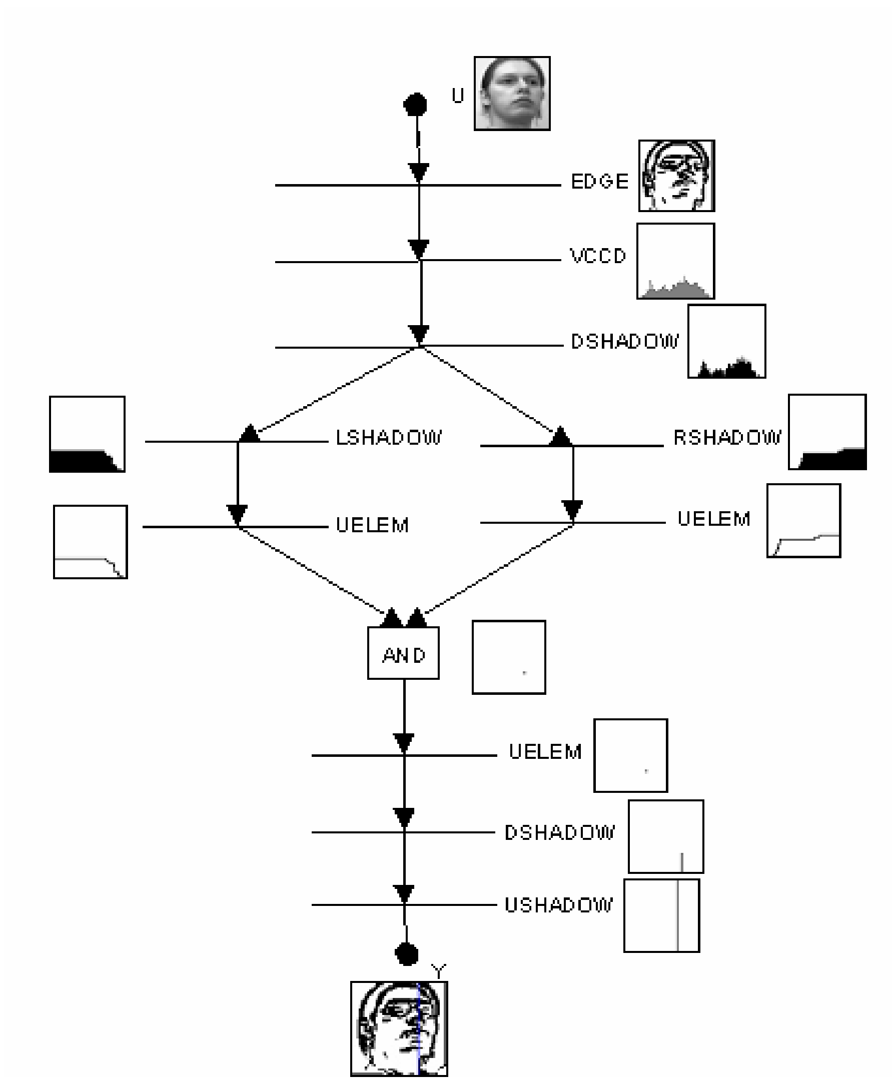


Figure 3-11. The UMF diagram of face symmetry axis extraction

To detect facial features in the face image, we make use of the information that there is a correspondence between the geometric positions of the features and the overall size of the face. We apply binary dilations [86] to create masks; and by using them, the facial features can be extracted (see Figure 3-12). From the relations defined in the geometric face model (Figure 3-9), a parameter can readily be calculated to control the dilation time.

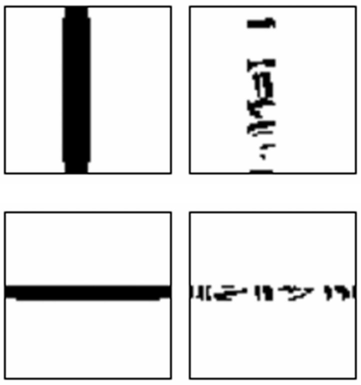


Figure 3-12. The dilated axes: the face symmetry-axis (the horizontal position of the nose; see Figure 3-13) and the axis of the eyes (which defines their vertical position; see Figure 3-13). The created masks are on the left, and the extracted features are shown next to them.

Thus, the extension of the face symmetry-axis detection algorithm described above is the detection of the nose, see Figure 3-13. However, besides the sought-for nose feature, some image-noise is extracted as well; some parts of the eyes, eyebrows, mouth, and the top of the head are also detected. To improve the detection, further analysis steps are needed. The parts of the top of the head can be removed from the image by using Figure Delete template [63]; and if the eyes have been detected (Figure 3-14 and Figure 3-15) the parts of the eyes and eyebrows can also be removed from the noisy nose-image. Even then, in the resulting image the mouth and some further noise below the nose are present; but if the top patch is chosen as the tip of nose and then used as a mask for a recall operation [63], the nose can be "cleanly" extracted from the original edge-map (see Figure 3-13).

After the nose is detected, the next step is to find the position of the mouth, the next "patch" below the nose. Deleting the nose, eyes and eyebrows from Figure 3-12 and applying the recall template operation, we can extract the mouth from the edge-map of the region of the nose (Figure 3-13).

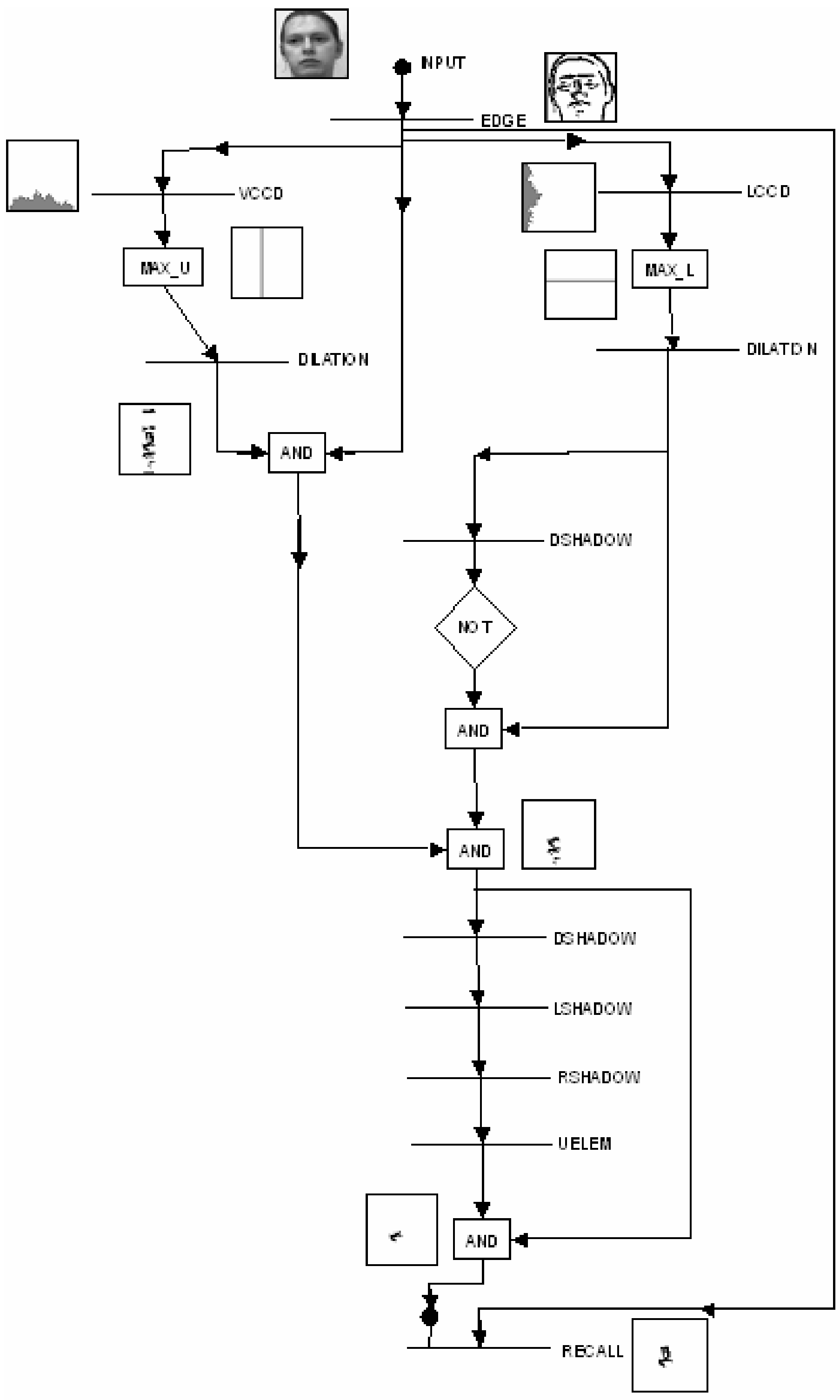


Figure 3-13. UMF diagram of the nose detection analogic algorithm

In the "eyes extraction" process, first their vertical position is detected and the two eyes are extracted from the image by using a mask (see Figure 3-12) which is created according to the relations defined in Figure 3-9. Then the nose is deleted from the image. After this, the image is divided into two parts along the symmetry-axis; and then the horizontal position of the eyes is found see Figure 3-14.

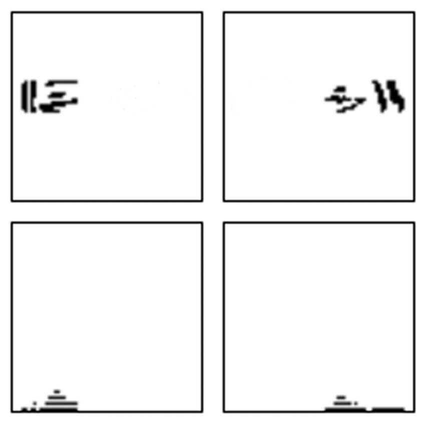


Figure 3-14. Detection of the horizontal position of the eyes. The noisy images of the eyes are in the upper images; below them are the results of the applied CCD operation in the Southern direction. The methodology is the same as that for the detection of the horizontal position of the nose.

Figure 3-15 shows the UMF diagram of the eyes-detection algorithm.

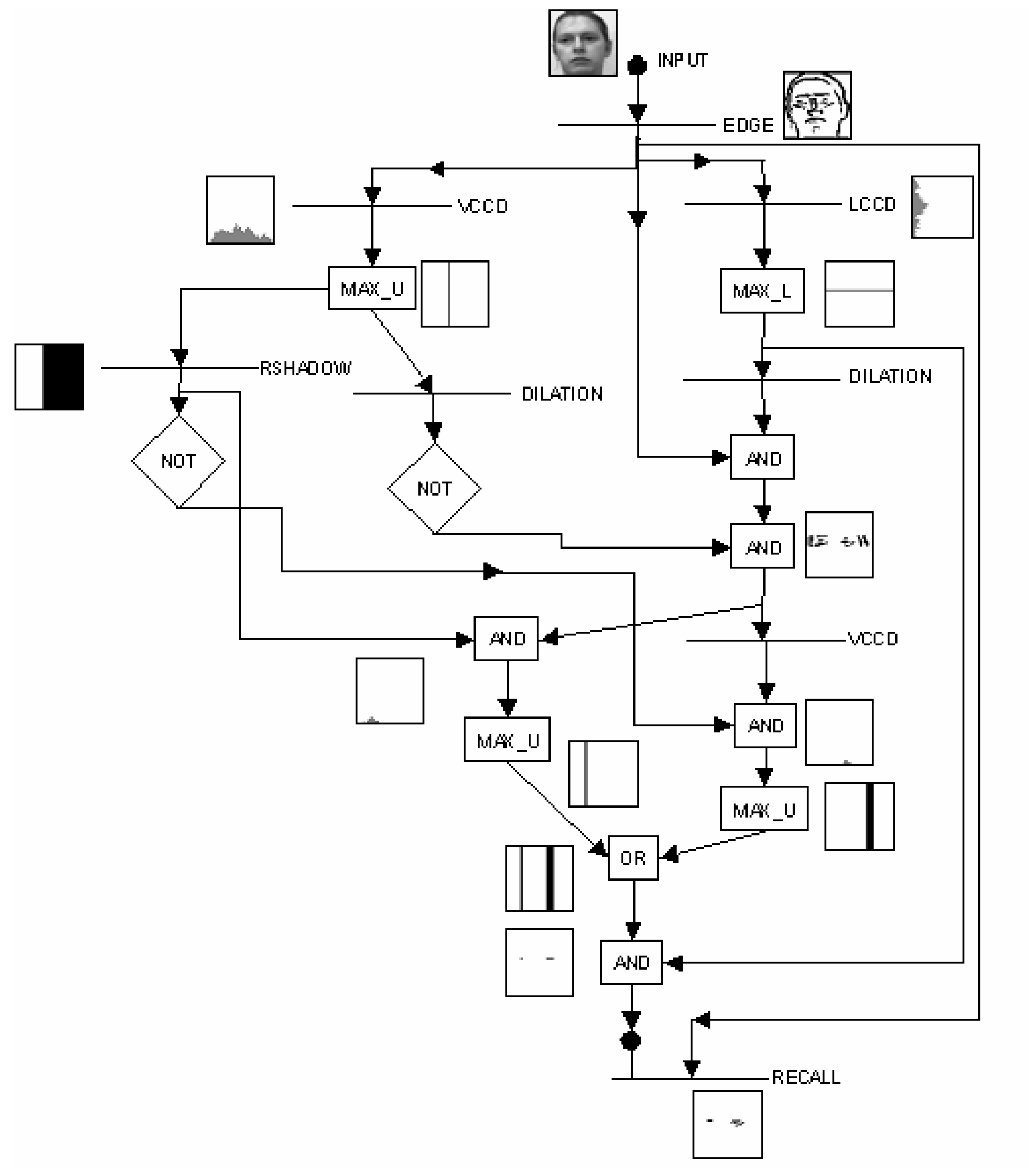


Figure 3-15. UMF diagram of the eyes-detection analogic algorithm

The outputs of the above-described algorithms can be fed up into the Decision Table circuit of [77] to achieve a face detector/recognizer. Bad detections can be filtered out in the final decision making process. If the *a posteriori* rate of the final Bayesian decision is above a threshold then the decision is accepted if below then not, as in [22].

3.3.3 Experimental results

The algorithms were tested on the UMIST face database [65]. This is widely used for testing face processing algorithms, and is freely available for download from the Internet [88]. It consists of 493 images of 20 persons, both male and female (some of the subjects wear glasses and some have facial hair). We chose this database since it contains, for each person, a range of poses from profile to frontal views, presented as a video image sequence. Images from the database are of size approximately 220×220 pixels, in 256 shades of grey. For our tests however the images were transformed to a size of 64×64 pixels, since the size of the ACE4K is 64×64 cells. This lower resolution is still sufficient for use in real applications; in surveillance systems, the resolution of faces typically varies from 20×20 to 50×50 pixels. As an example, Figure 3-16 shows a sequence of images (available at [33]) from subject 1a.

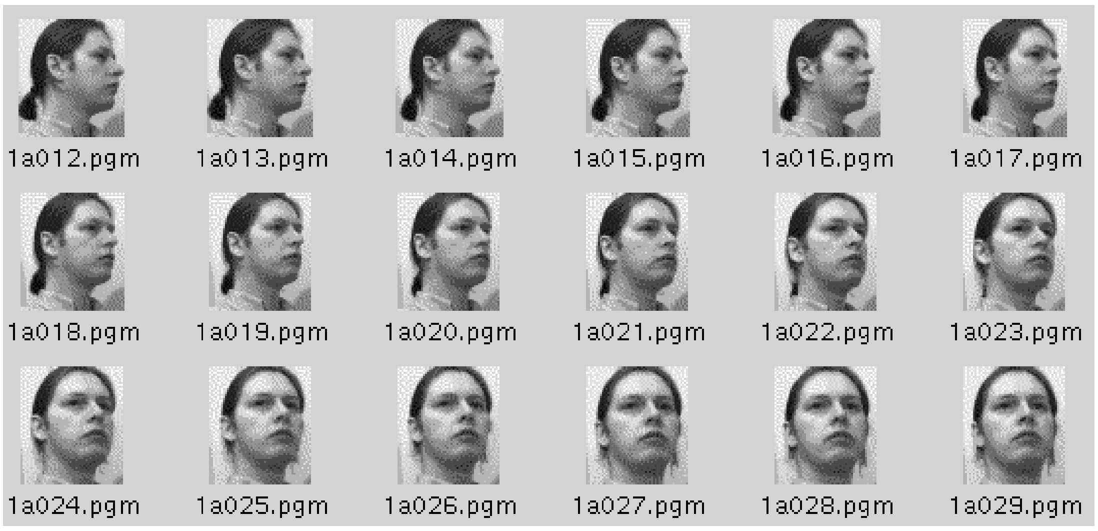


Figure 3-16. Images from subject 1a in the face-database

As can be seen in the Figure 3-16, most of the images are profile images of the subject, or nearly so. The algorithms were tested on a subset of 70 images; we selected from the database the face images on which the eyes can be seen.

The images in Figure 3-17 show some results of the face symmetry axis detection algorithm described above. Cases sometimes occur where the axis is marked by a band and not by a single line. However, the centre of this band can be found by applying a bipolar wave template operation [73], and the resulting axis then will be a single line.

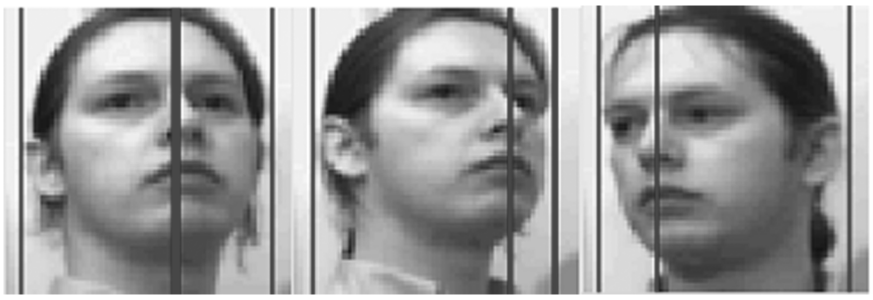


Figure 3-17. Results of ACE4K experiments of the face symmetry axis detection algorithm for faces in different poses

Figure 3-18 and Figure 3-19 show some results of the nose-, eyes- and mouth-detection algorithms implemented on the ACE4K device.

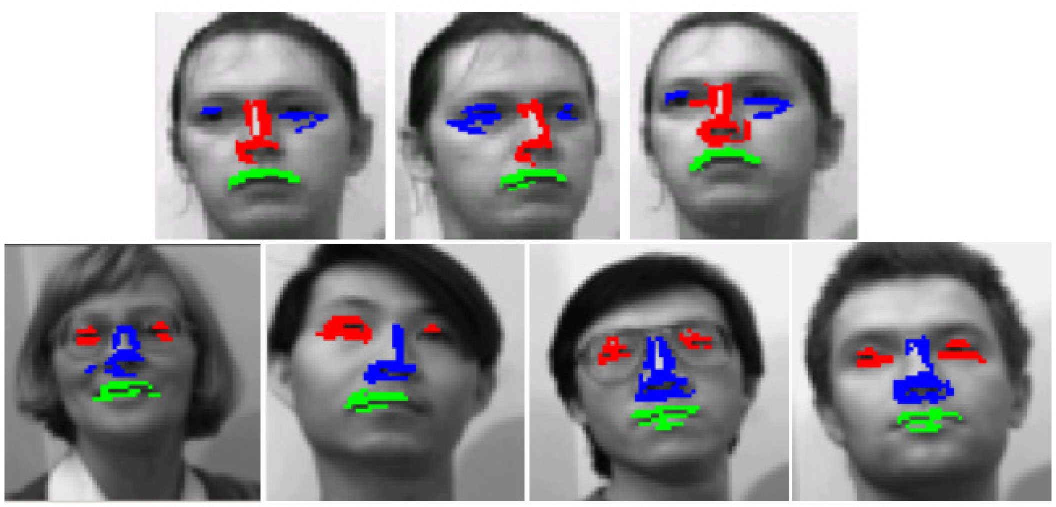


Figure 3-18. Results of experiments on nose, eyes and mouth detection (ACE4K)

The correct extraction of the features depends on the quality of the derived edge-map of the face image. If the detection of the edges is not precise, there may not be enough information to extract the facial features. If one or more of these features is wrongly detected, other features may also be detected incorrectly, as illustrated in Figure 3-19.

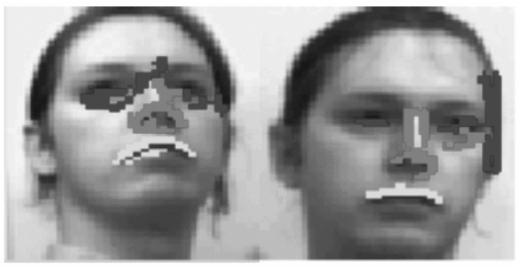


Figure 3-19. Typical errors caused by erroneous edge-detection

The accuracy of the algorithms used was verified by visual checking of the outputs by human observers. In the face images of size 64×64 pixels, the size of eyes and the width of nose-tip varies from 10 to 15 pixels, while the width of the mouth is larger than this. In the verification process, if the detected feature was within a circle of radius 5 pixels located on the centre of the true feature, then the detection result was counted as correct. The percentage of the correct detection of facial features is given in Table 3-4.

Table 3-4. Detection percentage for the extraction of primary facial features (for the database of 70 face images).

FEATURE	CORRECTLY DETECTED (%)
Nose	97
Eyes	89
Mouth	92

These results were judged to be acceptable [71], considering the fact that the input images were not ideal: shots were in most cases not frontal, and the lighting conditions did not have a uniform effect on all parts of the image.

Below we give computation complexity estimation for the proposed algorithm. Table 3-5 shows the number of different instructions used in the implemented algorithm.

The measured speed of the overall algorithm for the extraction of facial features, including capturing, downloading to the chip and the binarization of the image, is 20.3 ms per frame; which means processing of 50 images per second and it is feasible for real time processing. In paper [85] the achieved speed is 0.8 sec per frame; and in

[80] the speed is 0.96 sec per frame, if we recalculate the speed of the algorithms for a frame of size 64×64 pixels.

Table 3-5. Execution time estimation for the proposed algorithm for the extraction of facial features. The execution time refers to the processing of a 64×64 image.

Operation		Number of instructions	Estimated time (ms)
Memory transfer:	DSP to Chip	18	4.5
	Chip to DSP	16	2.0
	Chip to Chip	16	0.3
Logic operations:		24	0.1
Template operations:		48	11.0
Total (Σ)			18.9

3.4 Identification of rotated faces by using CNN

Based on the above described facial feature extraction algorithm a face identification analogic algorithm was introduced. Most of the face recognition algorithms focus on frontal facial views. However, pose changes often lead to large nonlinear variation in facial appearance due to self-occlusion and self-shading. Some algorithms have been proposed, which can recognize faces at a variety of poses. However most of these algorithms require gallery images at every pose [69]. These techniques are often too slow to be used for tracking purposes.

In our approach we assume that the human head is detected and localized. The task is divided into the following sub-tasks. First some features are extracted to estimate the pose of the head. Then the head is transformed to create canonical images, in which the face is filtered, rotated and scaled to a standard size. At last, searching the most similar face in a database makes the identification of face.

First, the algorithm extracts the face symmetry axis by using algorithm in Figure 3-11. The sides of the face are detected by shadowing the edge image of the face and then extracting the left and right side of the image by using hit and miss operators. Reading out the positions of the face symmetry axis and the sides of the face we have three parameters, c, r, l are the position of face symmetry axis, right side and left side, respectively, that describes the horizontal rotation of the face. If $r - c = c - l$ then we have a frontal face, otherwise we have a face rotated by $|r - 2c + l|$ pixels to the left if $r - 2c + l > 0$ and to the right if $r - 2c + l < 0$. Then applying a shift template to the rotated face image the face can be shifted back to the frontal view, see Figure 3-20.

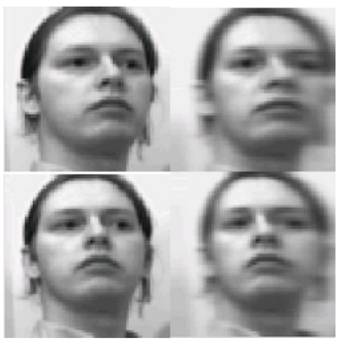
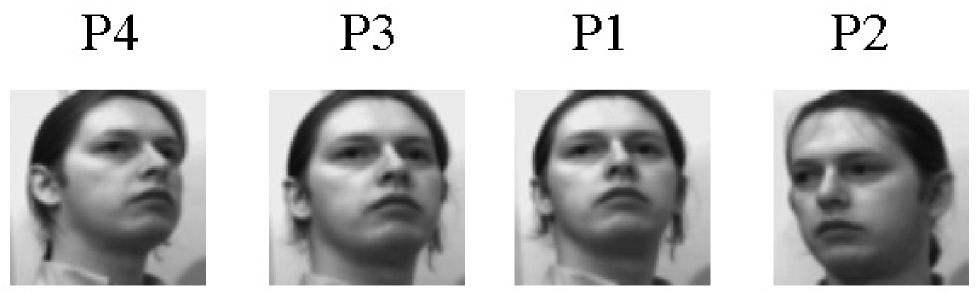


Figure 3-20. Rotation of face on CNN-UM (Simulation results). The left is the input image the right is the rotated.

For the comparison of different face images we apply a simple technique based on the use of whole image gray-level templates. In the template matching the images are represented as two-dimensional arrays of intensity values representing the whole face. The Euclidean distance is calculated between two templates (images) to compare the test images. When attempting recognition, the unclassified image is compared in turns to all of the database images, returning a score, the distance to an image in database. Then the unknown person is classified as the one giving the smallest score.

To test the algorithm's performance we have divided the face database into four classes according to the amount of rotation of the input face. Examples of each class can be seen in Figure 3-21. We have performed four experiments. In each of four experiments one of the classes is the training class for which the algorithm calculates the average rotation. All the other classes are the test classes and the algorithm transforms the input images to the training class and tries to find the closest image from the training class to the input test image. Figure 3-21 summarizes the results of the experiment.



P4	89%	80%	70%
92%	**P3**	83%	67%
84%	86%	**P1**	88%
72%	81%	90%	**P2**

Figure 3-21. Results of identification of rotated faces. The rows are the training classes and the columns are the test classes in the table. Example images from each class are also shown.

4 Summary

The main results of research on registration of images were presented in Chapter 2.5 and 2.6. Co-motion statistics were introduced in Chapter 2.5. It was shown that the principle of concurrent motions efficiently can be used for registration of cameras' overlapping fields of view. Chapter 2.6 presents an enhanced algorithm for the estimation of overlapping fields of view of two cameras based on the detection of concurrently changing pixels. A Markov chain based model describes the concurrent motions in two cameras. Experimental tests showed the advantages of the proposed approach.

In Chapter 3 an analogic algorithm was presented to detect facial features in images of human faces. The designed framework and algorithm is based on the fact that the geometry of faces can be described by only one parameter. A set of rules was defined to perform facial feature extraction by using CNN operations. Successful tests on a public face database prove the feasibility of the proposed methods.

4.1 Methods used in experiments

In the course of my work, theorems and assertions from the field of ordinary and partial differential equations, mathematical statistics, numerical geometry, optimization, reported results of image and video processing were explored.

The experiments for camera registration were performed by using the MDICAM multi-camera software system that was designed in the Analogical and Neural Computing Laboratory. For unique experiments I have also designed simulation systems in Matlab. Testing of the proposed algorithms was performed on various video sequences from personal experiments and from publicly available video databases.

For the design and testing of analogic algorithms I have used a software-hardware system developed in the Analogical and Neural Computing Laboratory. Designed CNN templates and algorithms were tested on software simulators, such as Aladdin System and on Cellular Visual Microprocessors: ACE4k and ACE16k. The simulating system was developed in the Analogical and Neural Computing Laboratory. The implementation of different methods was completed in different CNN languages (Alpha, AMC, UMF) ensuring their applicability on different platforms. The proposed methods were tested on images from a publicly available image database.

4.2 New scientific results

The First Thesis summarizes the results that related to the use of accumulated motion statistics for camera registration, the Second Thesis is about a Bayesian method for camera registration. The Third Thesis presents results that related to feature extraction on wave computers.

4.2.1 First Thesis

Accumulated motion statistics for automatic camera registration.

a. I gave a new camera registration method based on the use of co-motion statistics.

I defined the notion of co-motion statistics that is based on the accumulation of motion statistics for different camera views. I have shown experimentally that co-motion statistics can be used for camera registration without any *a priori* knowledge about the objects' appearance or motion and without human interaction. The method's accuracy is in the subpixel range.
Described in Chapter 2.4
Published in [2][6][16]

b. I showed that temporal alignment of image sequences in the case of co-motion statistics based camera registration is possible through the minimization of the estimated alignment error.

In camera registration, it is usually assumed that the cameras are synchronized. I have solved the problem of temporal alignment of image sequences during camera registration for unsynchronized cameras. The proposed method aligns the camera views for different time offsets and then searches for offset with minimum alignment error.
Described in Chapter 2.4
Published in [2][6]

4.2.2 Second Thesis

Automatic Bayesian method for camera registration without human interaction and assuming any *a priori* information about scene structure, appearance of objects or movement.

a. **I showed that the extraction of cameras' overlapping field of views can be done by a new automatic Bayesian iterative algorithm.**

   The knowledge of cameras' overlapping field of view is a prerequisite of the fast and robust matching of wide baseline stereo images. I proposed a Bayesian model for modelling concurrent changes in different camera views. I have shown that the solution of this model is equivalent to a solution of a periodic Markov chain. The developed method provides an automatic solution to the problem without any human interaction and *a priori* object model.
   Described in Chapter 2.5
   Published in [1][8]

b. **I showed that an entropy based analysis can be applied for the extraction of pixels of significant non-continuous changes.**

   I worked out an entropy based method for the classification of motion histories into two classes: (i) significant non-continuous changes; (ii) significant continuous changes or camera noise.
   Described in Chapter 2.5
   Published in [1][8]

In practice, the existing algorithms can be used only in restricted situations. The reported methods focus on the solution of the view-registration problem in respect of outdoor scenes, and neglect the additional difficulties, which tend to arise for indoor scenes. In the case of indoor cameras, the still-image based methods may fail due to the variability of conditions: occlusions, changing illumination etc. Due to the larger size of the moving objects, the cited motion-based methods will also fail; the observed motions are not necessarily on the ground-plane – while for outdoor scenes, such an assumption can safely be made. But if we could detect feature points that are on the groundplane then the motion based methods still can be applied for the matching of two views.

c. **I introduced shadow as salient features on the ground and I have shown experimentally that shadows efficiently can be used for camera registration.**

   Shadows are excellent features for indoor scenes, because they are mainly on the ground plane. I have experimentally shown that if we could detect shadows of moving objects then we have a lot of concurrently moving points in the cameras. By extracting them, the matching of the views can be done.
   Described in Chapter 2.5
   Published in [7]

4.2.3 Third Thesis

Working out robust feature extraction algorithms by using spatio-temporal dynamics.

a. **I did show that basic face features can be extracted by a set of procedures which can run on the reduced set of parallel operations in the CNN UM.**

   I have developed new analogic algorithms for the extraction of main, eyes, nose and mouth, facial features. The extraction of primary facial features such as eyes, nose and mouth relies on the fact that the distance between the eyes is proportional to the distances between the facial features [56]. All the algorithms run on the ACE4K CNN chip, and the achieved speed of the algorithms is equivalent to 50 frames per second.
   Described in Chapter3.3
   Published in [14]

b. I worked out an analogic algorithm for the identification of rotated faces by using CNN.

I have designed and implemented the algorithm in the CNN-UM framework. The algorithm is based on the estimation of face position by using analogic facial feature extraction algorithms. The steps of the algorithm are as follows: extraction of facial features, estimation of face position, transforming back the face to the frontal position and comparing to images in the database.
Described in Chapter 3.4
Published in [19]

4.3 Possible applications

All the developed methods and implementations offer solutions for real applications problems.
Methods of First and Second Theses provide general solutions for camera registration in real circumstances. They can be used in any multi-camera surveillance system where automatic functioning is needed.
Algorithms of the Third Thesis provide solutions for computer systems where real-time face analysis is needed, e.g. face detection, identification.

References

The author’s journal publications

[1] **Zoltán Szlávik**, Tamás Szirányi, “Stochastic view registration of overlapping cameras based on arbitrary motion”, IEEE Transactions on Image Processing, in print, 2006

[2] **Zoltán Szlávik**, Tamás Szirányi, László Havasi, “Video camera registration using accumulated co-motion maps”, ISPRS Journal of Photogrammetry and Remote Sensing, in print, 2006

[3] László Havasi, **Zoltán Szlávik**, Tamás Szirányi, “Detection of gait characteristics for scene registration in video surveillance system”, IEEE Transactions on Image Processing, in print, 2006

[4] László Havasi, **Zoltán Szlávik**, Tamás Szirányi, “Higher order symmetry for non-linear classification of human walk detection”, Pattern Recognition Letters, vol. 27(7), pp. 822-829, 2006

The author’s international conference publications

[5] László Havasi, **Zoltán Szlávik**, Tamás Szirányi, “Use of human motion biometrics for multiple-view registration”, Advanced Concepts for Intelligent Vision Systems, ACIVS 2005, LNCS, vol. 3708, pp. 35-44, 2005

[6] **Zoltán Szlávik**, László Havasi, Tamás Szirányi, ”Estimation of common groundplane based on co-motion statistics”, International Conference on Image Analysis and Recognition, ICIAR’04, LNCS, vol. 3211, pp. 347-353, 2004

[7] **Zoltán Szlávik**, Tamás Szirányi, László Havasi, “Analysis of dynamic scenes by using co-motion statistics”, IEEE International Workshop on Visual Surveillance, 2006

[8] **Zoltán Szlávik**, Tamás Szirányi, “Bayesian estimation of common areas in multi-camera systems”, IEEE International Conference Image Processing, ICIP, Atlanta, **accepted,** 2006

[9] Cs. Benedek, L. Havasi, T. Szirányi, **Z. Szlávik**, “Motion-based Flexible Camera Registration”, in Proc. of IEEE Advanced Video and Signal-Based Surveillance, AVSS’05, pp. 439-444, 2005

[10] **Zoltán Szlávik**, László Havasi, Tamás Szirányi, Csaba Benedek, “Random motion for camera calibration”, European Signal Processing Conference, EUSIPCO, Antalya, 2005

[11] **Zoltán Szlávik**, Tamás Szirányi, László Havasi, Csaba Benedek, “Optimizing of searching co-motion point-pairs for statistical camera calibration”, IEEE International Conference on Image Processing, ICIP, pp. 1178-1181, Genova, 2005

[12] László Havasi, **Zoltán Szlávik**, Tamás Szirányi, “Eigenwalks: walk detection and biometrics from symmetry patterns”, IEEE International Conference on Image Processing, ICIP, pp. 289-292, Genova, 2005

[13] László Havasi, **Zoltán Szlávik**, Csaba Benedek, Tamás Szirányi, “Learning human motion patterns from symmetries”, International Conference on Machine Learning Workshop on Machine Learning for Multimedia, Bonn, 2005, (on CD-ROM)

[14] **Zoltán Szlávik**, Tamás Szirányi, “Face analysis using CNN-UM”, Proc. of Cellular Neural Networks and Aapplications, CNNA’04, pp. 190-196, Budapest, 2004

[15] László Havasi, **Zoltán Szlávik**, “Symmetry feature extraction and understanding”, Proc. of Cellular Neural Networks and Aapplications, CNNA’04, pp. 255-261, Budapest, 2004

[16] **Zoltán Szlávik**, László Havasi, Tamás Szirányi, “Image matching based on co-motion statistics”, 2nd IEEE Int. Symposium on 3D Data Processing, Visualization, and Transmission, 3DPVT, 2004, (on CD-ROM)

[17] László Havasi, Csaba Benedek, **Zoltán Szlávik**, Tamás Szirányi, “Extracting structural fragments of overlapping pedestrians”, Proc. Of the 4th IASTED Int. Conference VIIP’04, pp. 943-948, Marbella, 2004

[18] László Havasi, **Zoltán Szlávik**, Tamás Szirányi, “Pedestrian Detection using derived Third-Order Symmetry of Legs”, ICCVG’04

[19] **Z. Szlávik**, T. Szirányi, “Face identification using CNN-UM”, Proceedings of European Conference on Circuit Theory and Design, ECCTD'03, vol. 2, pp. 81-85, Krakow, 2003

[20] **Z. Szlávik**, D. Bályа, T. Roska, “Properties of the adaptive integration formula to compute the CNN dynamic equation”, Proceedings of European Conference on Circuit Theory and Design, ECCTD'03, vol. 1, pp. 105-109, Krakow, 2003

Publications related to the dissertation

[21] Y. I. Abdel-Aziz, H. M. Karara, “Direct Linear Transformation from Comparator Coordinates into Object Space Coordinates in Close-range Photogrammetry”, in Proc. ASP/UI Symp. Close-Range Photogrammetry, pp. 1-18, 1971.

[22] D. H. Ballard, C. M. Brown: Computer Vision, Prentice-Hall, Englewood Cliffs NJ, pp. 539, 1982.

[23] S. T. Barnard, W. B. Thompson, “Disparity analysis of images,” IEEE Trans. PAMI, vol. 2, pp. 333-340, 1980.

[24] Cs. Benedek, T. Szirányi: A Markov Random Field Model for Foreground-Background Separation, Joint Hungarian-Austrian Conference on Image Processing and Pattern Recognition (HACIPPR), Veszprém, Hungary, May 11-13, 2005

[25] Y. Caspi, D. Simakov, and M. Irani, “Feature-based sequence-to-sequence matching,” in Proc. VAMODS (Vision and Modelling of Dynamic Scenes) workshop, with ECCV’02, Copenhagen, 2002.

[26] J. K. Cheng, T. S. Huang, “Image registration by matching relational structures,” Pattern Recog., vol. 17, pp. 149-159, 1984.

[27] I. Csiszár, “About the MDI”, in Multivariable statistical analysis (in Hungarian) edited by F. T. Móri and J. G. Székely, Műszaki Könyvkiadó, Budapest, 1986, pp. 209-232.

[28] R. Cucchiara, C. Grana, G. Neri, M. Piccardi, and A. Prati, *The Sakbot System for Moving Object Detection and Tracking*. Video-Based Surveillance Systems-Computer Vision and Distributed Processing, 2001, pp. 145—157.

[29] Faugeras, O. D., Luong, Q.-T., Maybank, S. J., 1992. Camera self-calibration: Theory and experiments. Proc. European Conference on Computer Vision, Lecture Notes in Computer Science 588, Santa Margherita Ligure, 19-22 May 1992, pp. 321-334.

[30] V. Ferrari, T. Tuytelaars, L. V. Gool "Wide-baseline muliple-view Correspondences", in Proc. of IEEE CVPR, I, pp. 718-725, 2003

[31] Fischler MA, Bolles RC. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. CACM, 1981,24(6):381-395.

[32] C. M. Grinstead, J. L. Snell, Introduction to probability: Second revised edition, AMS, 1997

[33] Hall, D., Nascimento, J., Ribeiro, P., Andrade, E., Moreno, P., Pesnel, S., List, T., Emonet, R., Fisher, R. B., Santos-Victor, J., Crowley, J. L., 2005. Comparison of target detection algorithms using adaptive background models Proc. 2nd Joint IEEE Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, Beijing, 15-16 October 2005, pp. 113-120.

[34] R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision. Cambridge, Cambridge University Press, 2003.

[35] L. Havasi, T. Szirányi, "Estimation of vanishing point in camera-mirror scenes using video", Optics Letters, vol 10, pp. 1411-1413, 2006.

[36] J. Kang, I. Cohen, G. Medioni "Persistent objects tracking across multiple non overlapping cameras," in *Proc. of WACV/MOTION'05,* 2005, vol. 2, pp. 112-119.

[37] Sohaib Khan, Mubarak Shah: Consistent Labeling of Tracked Objects in Multiple Cameras with Overlapping Fields of View. IEEE Trans. Pattern Anal. Mach. Intell. 25(10): 1355-1360 (2003)

[38] L. Kovács, T. Szirányi, "Relative focus map estimation using blind deconvolution," *Optics Letters*, vol.30, pp. 3021-3023, 2005.

[39] L. Lee, R. Romano, G. Stein, "Monitoring activities from multiple video streams: establishing a common coordinate frame," IEEE Trans. PAMI, vol. 22, 2000.

[40] Maas, H-G., 1999. Image sequence based automatic multi-camera system calibration techniques. ISPRS Journal of photogrammetry and Remote Sensing 54 (5-6), 352-359.

[41] S. J. Maybank, O. Faugeras. A theory of self-calibration of a moving camera. Int. Journal of Computer Vision, 8(2):123-152, 1992

[42] W. Nunziati, J. Alon, S. Sclaroff, A. D. Bimbo, View registration using interesting segments on planar trajectories, in Proc. of IEEE AVSS'05, pp. 75-80, 2005

[43] Pollefeys, M., Koch, R., Vergauwen, M., Van Gool, L., 2000. Automated reconstruction of 3D scenes from sequences of images. ISPRS Journal of photogrammetry and Remote Sensing 55 (4), 251-267.

[44] Press, W.H., B.P. Flannery, S.A. Teukolsky and W.T. Vetterling, 1986. Numerical Recipes: The Art of Scientific Computing. Cambridge University Press, Cambridge.

[45] R. J. Radke, S. Andra, O. Al-Kofahi, B. Roysam, "Image change detection algorithms: a systematic survey", *IEEE Trans. on Image Processing*, vol. 14(3), pp. 294-307, 2005.

[46] Richardson, W. H. "Bayesian-based iterative method of image restoration," J. Opt. Soc. Am., vol. 62, p. 55-59, 1972

[47] P. L. Rosin, E. Ioannidis, "Evaluation of global image thresholding for change detection", *Pattern Recognition Letters*, vol. 24, pp. 2345-2356, 2003.

[48] Ross, M. S., Introduction to probability models, Academic Press, 1997

[49] Scott, D. W., *"Multivariate Density Estimation: Theory, Practice, and Visualization,"* John Wiley and Sons, 1992.

[50] C. Stauffer, E. Grimson, "Learning Patterns of Activity Using Real-Time Tracking", *IEEE Trans. PAMI*, vol. 22(8), pp. 747-757, 2000.

[51] T. Szirányi, "Subpixel pattern recognition by image histograms," *Pattern Recognition*, vol. 27(8), pp. 1079-1092, 1994.

[52] Tanenbaum, A. S., van Steen, M., 2001. Distributed Systems: Principles and Paradigms. Prentice Hall.

[53] J. Weng, N. Ahuja, T. S. Huang, "Matching two perspective views," IEEE Trans. PAMI, vol. 14, pp. 806-825, 1992.

[54] Z. Zhang, R. Deriche, O. Faugeras, Q.-T. Luong, "A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry," Artificial Intelligence Journal, vol. 78, pp. 87-119, 1995.

[55] Bálya D, Roska T. Face and eye detection by CNN algorithms. J of VLSI Signal Processing Special Issue: Spatiotemporal Signal Processing with Analogic CNN Visual Microprocessors 1999; 23: 497-511.

[56] Brunelli R, Poggio T. Face recognition through geometrical features. Proceedings of ECCV, S. Margherita Ligure, 1992; 792-800.

[57] Brunelli R, Poggio T. Face recognition: features versus templates. IEEE Transactions on Pattern Analysis and Machine Intelligence 1993;15(10):1042-1052.

[58] Chellappa R, Wilson C L, Sirohey S. Human and Machine Recognition of Faces: A Survey. Proceedings of IEEE 1995; 83:705-740.

[59] Chua L O, Roska T. Cellular Neural Network and Visual Computing. Cambridge: Cambridge Univ. Press; 2002.

[60] Espejo S, Domínguez-Castro R, Liñán G, Rodriguez-Vázquez A. A 64x64 CNN universal chip with analog and digital I/O. 5th Int. Conf. Electronics, Circuits and Systems (ICECS-98), Lisbon 1998;203-206.

[61] Han C C, Liao H Y M, Yu K C, Chen L H. Fast face detection via morphology-based pre-processing. Pattern Recognition 2000; 33:1701-1712.

[62] Hjelmas E, Low B K. Face detection: A survey. Computer Vision and Image Understanding 2001:83(3):236-274.

[63] http://lab.analogic.sztaki.hu/csl/CSL

[64] Gorodnichy D. On importance of nose for face tracking. Proc. of 5th Int. Conference on Automatic Face and Gesture Recognition, Washington D. C. 2002;183-193.

[65] Graham D B, Allinson N M. Characterizing Virtual Eigensignatures for General Purpose Face Recognition. Face Recognition: From Theory to Applications, NATO ASI Series F, Computer and Systems Sciences 1998;163:446-456.

[66] Liñán G, Domínguez-Castro R, Espejo S, Rodriguez-Vázquez A. ACE16K: a programmable focal plane vision processor with 128x128 resolution. Proceedings of ECCTD'01 2001;345-348.

[67] Matsumoto T, Chua L O, Suzuki H. CNN Cloning Template: Connected component detector. IEEE Trans. on Circuits and Systems 1990;37:633-635.

[68] Matsumoto T, Chua L O, Suzuki H. CNN Cloning Template: Shadow detector. IEEE Trans. on Circuits and Systems 1990:37:1070-1073.

[69] R. Nelson, I. Green "Tracking Objects using Recognition", 16th Int. Conf. On Pattern Recognition, 2002

[70] Nemes L, Chua L O, Roska T. Implementation of Arbitrary Boolean Functions on the CNN Universal Machine. Int. J. Circuit Theory and Applications - Special Issue: Theory, Design and Applications of Cellular Neural Networks: Part I: Theory 1998:26(6):593-610.

[71] Nikolaidis A, Pitas I. Facial feature extraction and pose determination. J of The Pattern Recognition 2000;33:1783-1791.

[72] Perlibakas V. Automatical detection of face features and exact face contour. Pattern Recognition Letters 2003;24:2977-2985.

[73] Petras I, Roska T. Application of Direction Constrained and Bipolar Waves for Pattern Recognition. Proceedings of Int. Workshop on Cellular Neural Networks and Their Applications (CNNA2000), Catania 2000:3-8.

[74] Roska T. Computational and computer complexity of analogic cellular wave computers. Proceedings of CNNA, Frankfurt 2002;313-338.

[75] Rowley H A, Baluja S, Kanade T. Neural-network based face detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 1998:20(1):23-38.

[76] Sobottka K, Pitas I. A novel method for automatic face segmentation, facial feature extraction and tracking. Signal Process. Image Commun 1998;12(3):263-281.

[77] Szirányi T, Csicsvári J. High speed character recognition using a dual Cellular Neural Network architecture. IEEE Trans. Circuits and Systems 1993;40(3(II.)):223-231.

[78] Szirányi T, Csapodi M. 'Texture Classification and Segmentation by Cellular Neural Network using Genetic Learning. Computer Vision and Image Understanding 1998;71(3):255-270.

[79] Szirányi T, Zerubia J, Czúni L, Geldreich D, Kato Z. Image Segmentation Using Markov Random Field Model in Fully Parallel Cellular Network Architectures. Real-Time Imaging 2000;6(3):195-211.

[80] Verma R C, Schmid C, Mykolajczyk K. Face detection and tracking in a video by propagating detection probabilities. IEEE Transaction on Pattern Analysis and Machine Intelligence 2003;25(10):1215-1228.

[81] Viola P, Jones M J. Robust Real-Time Face Detection. International Journal of Computer Vision 2004:57(2):137-154.

[82] Wang J-G, Sung E. Frontal-view face detection and facial feature extraction using color and morphological operations. Pattern Recognition Letters 1999;20:1053-1068.

[83] Yang M-H, Kriegman D, Ahuja N. Detecting faces in images: A survey. Transactions on Pattern Analysis and Machine Intelligence 2002:24(1):34-58.

[84] Zhao W, Chellappa R, Rosenfeld R. Face recognition: a literature survey. ACM Computing Surveys 2003; 399-458.

[85] Zhou S, Krueger V, Chellappa R. Face recognition from video: a condensation approach. Proc. of 5th Int. Conference on Automatic Face and Gesture Recognition, Washington D. C. 2002;221-228.

[86] Zarándy A, Stoffels A, Roska T, Chua L O. Implementation of Binary and Gray-Scale Mathematical Morphology on the CNN Universal Machine. IEEE Trans. on Circuits and Systems I: Fundamental Theory and Applications 1998:45(2):163-168.

[87] Zarándy A, Rekeczky Cs, Földesy P, Szatmáry I. The new framework of applications – the Aladdin System. Journal of Cicuit Systems and Computers 2003;12(6):769-781.

[88] http://images.ee.umist.ac.uk/danny/database.html

Appendix

The CNN Software Library (CSL) [63] contains many templates {A, B, z} for various image-processing tasks. These implement waves of both simple and exotic form, with standard and complex templates and cells of first-order, as well as of a higher order (complex). The following symbol represents an elementary template instruction:

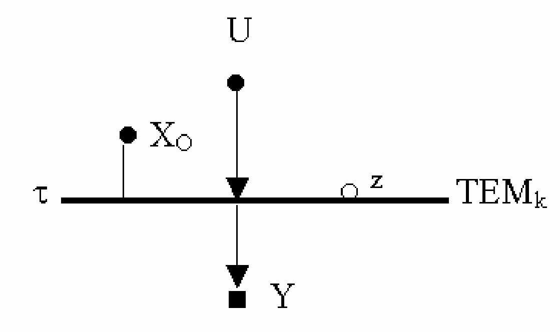


Figure A-1: The symbol of a template instruction, where U is the input and Y the output. X_0 denotes the initial state; τ the time-constant; z the bias term. TEM_k is the name of the template in CSL.

By using the above symbol of template operation as a building block, one can draw UMF diagrams to describe a complete analogic algorithm. To interconnect the steps of the algorithms, considerations similar to those in a traditional flow diagram are used. For details, see [74].