Faculteit Ingenieurswetenschappen

27 januari 2008

Verslag
Project beeldverwerking
2007-2008
A study of the 2D SIFT algorithm

Dimitri Van Cauwelaert

Prof. dr. ir. W. Philips

dr. ir. A. Pizurica

Content

1. Introduction

This report covers an initial exploration of the SIFT (Scale invariant Feature transform) algorithm[1], invented by David Lowe[2], used for image matching and object recognition. The idea is to understand how the algorithm works and how it can be improved. Image matching is a fundamental aspect of many problems in computer vision, including
object or scene recognition, solving for 3D structure from multiple images, stereo correspondence, and motion tracking, … .

The SIFT algorithm is based on a model of the behavior of complex cells in the cerebral cortex of mammalian vision. Recent research by Edelman, Intrator and Poggio – indicates that if feature position is allowed to shift over a small area, while maintaining orientation and spatial frequency, reliable matching increases significantly

The SIFT algorithm is a technique to extract highly distinctive invariant features from images, which can be used to perform a reliable matching between different views of an object or scene. In particular the features are to some extend invariant to image scaling and rotation and are shown to provide robust matching across a substantial range of affine distortion, change in 3D viewpoint, addition of noise and change in illumination. The features are highly distinctive, in the sense that a single feature can be correctly matched with high probability against a large database of features from many images. The cost of extracting these features is minimized by taking a cascade filtering approach, in which the more expensive operations are applied only at locations that pass an initial test. Following the extraction of these features we can use them for object recognition by matching individual features to a database of features from known objects using a fast nearest-neighbor algorithm, followed by a Hough transform to identify clusters belonging to a single object, and finally performing verification through least-squares solution for consistent pose parameters. This approach to recognition can robustly identify objects among clutter and occlusion while achieving near real-time performance.

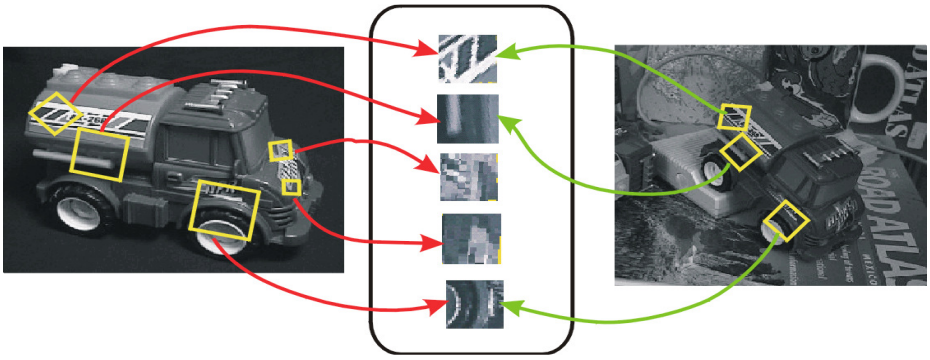

figure 1: matching of features between two similar objects in different images

[1] Patent pending

[2] University of British Colombia, Vancouver, BC, Canada

2. Detection of scale-space extrema

The first step to identify possible stable feature points, is to identify extrema in scale space.
By looking for points in scale space, we are effectively achieving invariance to scaling. Koenderink (1984) and Lindeberg (1994) showed that exploring scale space can best be done by using a Gaussian function as scale-space kernel. If we define $L(x, y, \sigma)$ as a Gaussian filtered image, $G(x, y, \sigma)$ as a Gaussian kernel and I(x,y) as an image we can write

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$$

Where * is the convolution operator.

Lowe (1999) indicates that stable keypoint locations can efficiently be detected by convolving a Difference of Gaussians (DoG) from nearby scales with the image.

$$\begin{aligned} D(x, y, \sigma) &= \big(G(x, y, k\sigma) - G(x, y, \sigma)\big) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned}$$

Where $D(x, y, \sigma)$ is a difference image. The above equations show that the difference image can be computed by subtracting two Gaussian filtered images from nearby scales separated by a factor k. Each octave (i.e. doubling of σ) is divided in s scales, resulting in

$$k = 2^{1/s}$$

For each octave S + 3 DoG images will be produced, resulting in an extrema detection in the full range of the octave. Once a complete octave has been filtered, the last Gaussian filtered image L is downsampled with a factor two in order to limit computational burden. This can be done because filtering an image with a Gaussian function, is effectively discarding the high spatial frequency content of the image, thus the image can be downsampled, while limiting the risk of aliasing. Experiments by Lowe (2004) indicate that the ideal (in terms of maximum matching of stable features) number of scales per octave is three.

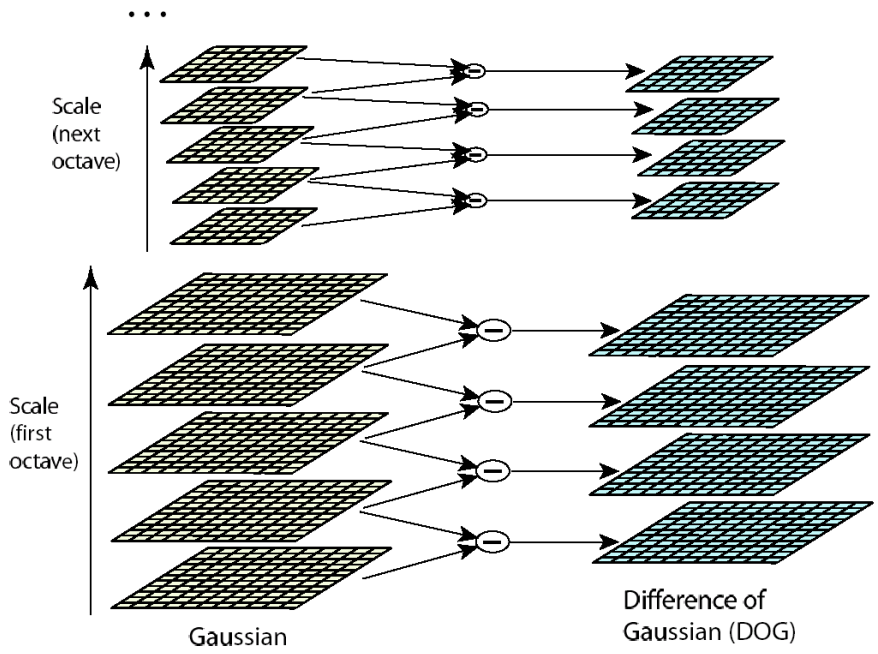

figure 2: For each octave of scale space, the initial image is repeatedly convolved with Gaussians to produce a set of scale space images shown on the left. Adjacent Gaussian images are subtracted to produce the difference –of-Gaussian images on the right. After each octave, the Gaussian images is down-sampled by a factor of 2, and the process repeated.

3. Local extrema detection

The previous paragraph described how DOG images were extracted. In each of the DOG images (except for the boundary ones) local extrema are detected, this can be done by comparing each pixel with its surrounding 8 neighbors and 9 neighbors from the DOG above and below the DOG currently under investigation. Pixels that result in a value larger or smaller than its surrounding pixels are identified as extrema.

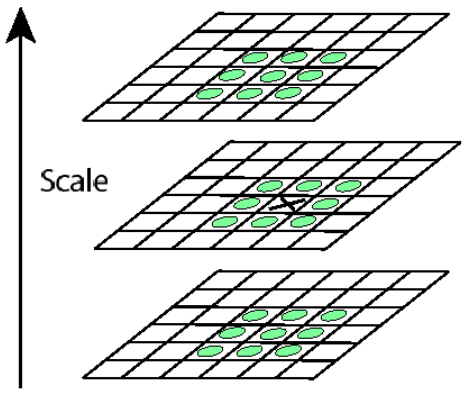

figure 3: Maxima and minima of the difference-of-Gaussian images are detected by comparing a pixel (marked with X) to its surrounding 26 neighbors in 3x3 regions at the current and adjacent scales (marked with circles)

4. Accurate keypoint localization

In order to increase the reliability of the keypoints position, we will fit a 3D quadratic function to the candidate keypoints to gain a more precision on the location and scale. We will make use of a Taylor expansion (which we will limit up to the quadratic terms) of the scale-space function, $D(x, y, \sigma)$, shifted so that the origin is at the sample point:

$$D(\mathbf{x}) = D + \frac{\partial D^T}{\partial x}\mathbf{x} + \frac{1}{2}\mathbf{x}^T\frac{\partial^2 D}{\partial x^2}\mathbf{x}$$

Where D and its derivates are evaluated at the sample point and $\mathbf{x} = (x, y, \sigma)^T$

By deriving D(**x**) with respect to **x** and setting to zero we can determine the extremum **y** of the approximation function.

$$\boldsymbol{y} = -\frac{\partial^2 D^{-1}}{\partial \boldsymbol{x}^2}\frac{\partial D}{\partial x}$$

The Hessian and the derivative of D are approximated by differences of neighboring sample points. Solving the 3x3 linear system can thus be solved with minimal computational cost.

$$D(\mathbf{y}) = D + \frac{1}{2}\frac{\partial D^T}{\partial x}\mathbf{y}$$

If **y** is larger than 0.5 in any main direction this means that the actual extremum is closer to another sample point. A new sample point (in the direction of the largest deviation from **x**)
Is chosen and the before discussed approximation repeated until a stable extremum is found.

Once a stable extremum **y** is found**,** we determine the function value (by filling in **y** in the approximation formula for **x).** The absolute value can be used to determine whether the extremum is strong valued, meaning the location and scale of the extremum is less sensitive to noise. An advised value to discard extrema is a value below 0.03

5. Eliminating Edge responses

In the previous section we described how extrema resulting from low contrast regions could be removed. Removing these points however is not sufficient. The DOG function will also have a strong response along edges, even if the location along the edge is poorly determined and therefore unstable to small amounts of noise. A poorly defined peak in the difference of Gaussian function will have a large principal curvature across the edge but a small one in the perpendicular direction. Using a 2x2 Hessian matrix, $\boldsymbol{H}$, computed at the location and scale of the keypoint, we can determine the principal curvatures.

$$\boldsymbol{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

The derivatives are estimated using differences of neighboring sample points.

The eigenvalues of H are proportional to the principal curvature of D. Harris and Stephens (1988) indicated we can avoid calculating the eigenvalues, since we are more interested in the ratio of the eigenvalues (ratio of curvature). If we assume that α is the eigenvalue with largest magnitude and β the eigenvalue with the smallest magnitude, we can compute the sum of the eigenvalues from the

trace of H and the product from its determinant

$$Tr(H) = D_{xx} + D_{yy} = \alpha + \beta$$

$$Det(H) = D_{xx} D_{yy} - (D_{xy})^2 = \alpha\beta$$

If the determinant of the Hessian turn out to be negative, this indicates is the point not an extremum, and it is discarded.

Now let be the ratio between α and β

$$\alpha = r\beta$$

Then

$$\frac{Tr(H)^2}{Det(H)} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r\beta+\beta)^2}{r\beta^2} = \frac{(r+1)^2}{r}$$

The value of the latter equation increases as the ratio of the curvatures increases.

Using the above equation we can very efficiently calculate whether the ratio of curvature is below some threshold. An advised maximum ratio of curvature is 10.

6. Orientation assignment

In the previous sections we concentrated on achieving scale invariance and invariance to noise up to some point. The next step is to assign one or more directions to the candidate keypoints. By doing so and describing the keypoints (see later sections) relative to this assigned orientation we can achieve rotational invariance. For each candidate keypoint the Gaussian blurred image L closest to the assigned scale of the keypoint is determined. Now an orientation histogram (consisting of 36 orientation bins) is formed and the neighboring sample points are used to vote for these orientation bin. For each neighboring sample point (in practice all sample points are precomputed) the orientation and strength of the discrete approximation of the gradient is determined using the following formulas.

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

Each orientation vote (using the gradient direction) from the neighboring sample points is weighted with the strength of the gradient and a circular Gaussian indicator function with σ equal to 1.5 that of the scale currently under investigation. Peaks in the histogram correspond to dominant directions of global gradients. The highest peak and any peak within 80 % of the value of the highest peak are used as assigned orientations. To each of these orientations a parabola is fit to the three closest histogram values in order to determine the orientation more precisely and thus further increasing the stability of the keypoints.

7. The local image descriptor

In the previous section we described how a scale, position and orientation to each keypoint was assigned. By describing the region around the keypoint relative to these parameters we achieve invariance to these parameters. The keypoints should be described in such a way that they are highly

distinctive yet as invariant as possible to remaining variations, such as change in illumination or 3D viewpoint.

The way the keypoints are described, is derived from a model of mammalian vision as studied by Edelman, Intrator and Poggio (1997). Following this model to describe the keypoints leads to better invariance to changes in 3D viewpoint.

In a region around the keypoint the magnitudes and orientations of the gradients are computed (as indicated before, in practice they are precomputed). In order to achieve orientation invariance, the coordinates of the descriptor and the gradient orientations are rotated relative to the keypoint orientation. A Gaussian weighting function with σ equal to one half the width of the descriptor window is used to assign a weight to the magnitude of each point (similar to the orientation assignment in the previous section). The purpose of this Gaussian window is to avoid sudden changes in the descriptor with small changes in the position of the window, and to give less emphasis to gradients that are far from the center of the descriptor, as they will be most affected by misregistration errors.
Significant shift in gradient positions is allowed by creating orientation histograms over 4x4 sample regions. The figure below shows eight directions for each orientation histogram, with the length of each arrow corresponding to the magnitude of that histogram entry. A gradient sample on the left can shift up to 4 sample positions while still contributing to the same histogram on the right, thereby achieving the objective of allowing for larger local positional shifts. It is however important to avoid all boundary affects in which the descriptor abruptly changes as a sample shifts smoothly from being within one histogram to another or from one orientation to another. Therefore, trilinear interpolation is used to distribute the value of each gradient sample into adjacent histogram bins. In other words, each entry into a bin is multiplied by a
weight of 1 - d for each dimension, where d is the distance of the sample from the central
value of the bin as measured in units of the histogram bin spacing.

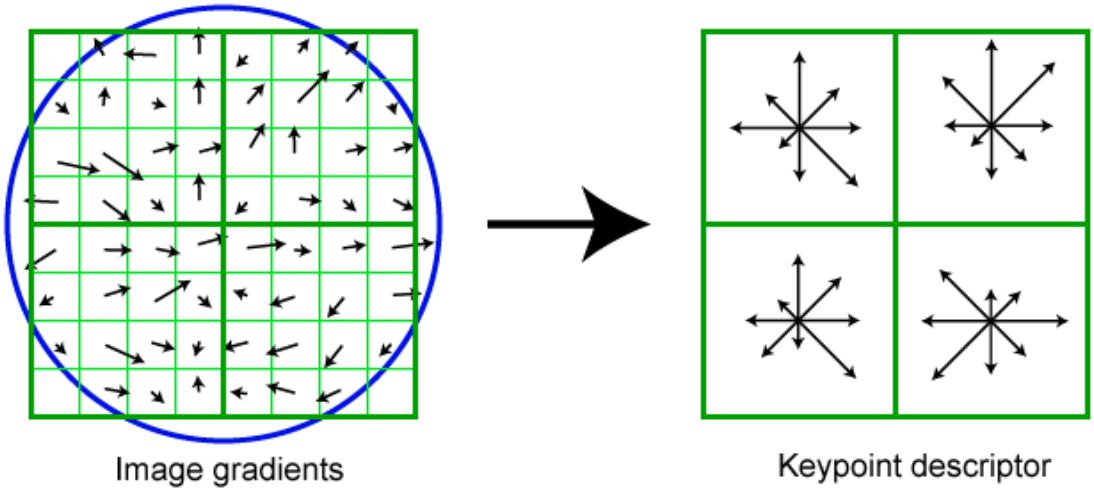

figure 4: Figure 7: A keypoint descriptor is created by first computing the gradient magnitude and orientation at each image sample point in a region around the keypoint location, as shown on the left. These are weighted by a Gaussian window, indicated by the overlaid circle. These samples are then accumulated into orientation histograms summarizing the contents over 4x4 subregions, as shown on the right, with the length of each arrow corresponding to the sum of the gradient magnitudes near that direction within the region. This figure shows a 2x2 descriptor array computed from an 8x8 set of samples, whereas the real implementation uses 4x4 descriptors computed from a 16x16 sample array.

The descriptor is formed from a vector containing the values of all the orientation histogram entries resulting in a 4x4x8 = 128 element feature vector for each keypoint. Finally, the feature vector is modified to reduce the effects of illumination change. First, the vector is normalized to unit length. A change in image contrast in which each pixel value is multiplied by a constant will multiply gradients by the same constant, so this contrast change will be canceled by vector normalization. A brightness change in which a constant is added to each image pixel will not affect the gradient values, as they are computed from pixel differences. Therefore, the descriptor is invariant to affine changes in illumination. However, non-linear illumination changes can also occur due to camera saturation or due to illumination changes that affect 3D surfaces with differing orientations by different amounts. These effects can cause a large change in relative magnitudes for some gradients, but are less likely to affect the gradient orientations. Therefore, we reduce the influence of large gradient magnitudes by thresholding the values in the unit feature vector to each be no larger than 0.2, and then renormalizing to unit length. This means that matching the magnitudes for large gradients is no longer as important, and that the distribution of orientations has greater emphasis.

In order to change the complexity of the descriptor, two parameters can be modified, the numbers of orientations , r, in the histograms, and the width, n, of the n x n array of orientation histograms. This results in a dimensionality of the descriptor vector of rn^2. As the complexity of the descriptor enhances, discrimination will rise at the penalty of more sensitivity to shape distortion and occlusion. In the implementation a 4x4x8 = 128 dimensional vector is used.

8. Matching descriptors

In order to be applicable to object recognition, features (meaning keypoints) from different images should be matched with high reliability. One way to do this, is to determine the distance between two possible matching points and then retain the two points having the smallest distance (and being below a certain threshold value) . Using a global threshold however does not seem to perform well as some descriptors are much more discriminative than others. A more robust implementation can be realized by comparing the distance to the closest match to that of the second closest match and using the ratio of these distances as criterion for a good match. Experiments by Lowe (2004) result in the following PDF for correct and incorrect matches. We can clearly see that their overlapping region (resulting in false positives, or false negatives) is rather small, indication the distance ratio is sufficiently distinctive. An advised separation level is 0.8

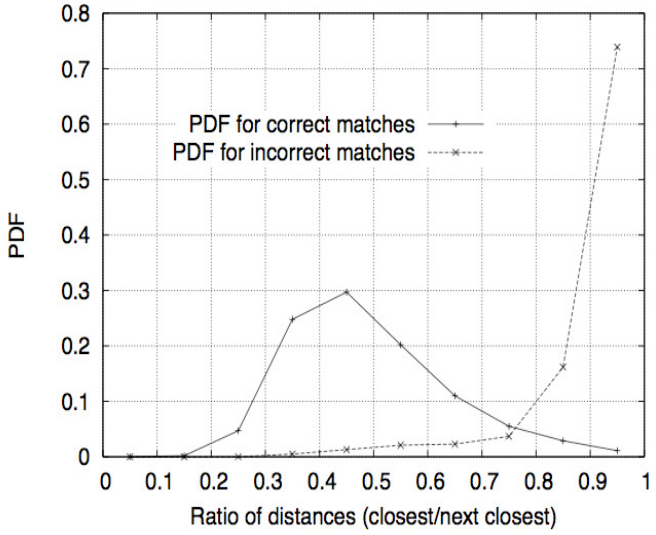

figure 5: The probability that a match is correct can be determined by taking the ratio of distance from the closest neighbor to the distance of the second closest. The solid line shows the PDF of this ratio for correct matches, while the dotted line is for matches that were incorrect. This figure was generated by matching images following random scale and orientation change, a depth rotation of 30 degrees, and addition of 2% image noise, against a database of 40,000 keypoints.

9. Efficient nearest neighbor indexing

In order to avoid a full exhaustive search during the matching process an approximation algorithm called the Best-Bin-First (BBF) algorithm is used. It is approximate in the sense that it returns the closest neighbor with high probability. Only the 200 nearest (in terms of their descriptor distance) neighbor candidates are checked. Tests indicate this only results in a slight loss of points being matched (Lowe 2004).

10. Clustering using the generalized Hough transform

Using the matching procedure described in the previous section, we are able to identify pairs of points with a reasonable chance of effectively belonging to the same object. However we shall also have points which in reality do not belong to the same object. In order to increase the reliability of the match we shall identify clusters in pose space, using the generalized Hough transform, of points agreeing on the same object . The Hough transform identifies clusters of features with a consistent interpretation by using each feature to vote for all object poses that are consistent with the feature. When clusters of features are found to vote for the same pose of an object, the probability of the interpretation being correct is much higher than for any single feature. Each keypoint specifies 4 parameters: 2D location, scale, and orientation, and each matched keypoint in the database has a record of the keypoint's parameters relative to the training image in which it was found. Therefore, we can create a Hough transform entry predicting the model location, orientation, and scale from the match hypothesis. This prediction has large error bounds, as the similarity transform implied by these 4 parameters is only an approximation to the full 6 degree-of-freedom pose space for a 3D object and also does not account for any nonrigid deformations. Therefore, we use broad bin sizes of 30 degrees for orientation, a factor of 2 for scale, and 0.25 times the maximum projected training image dimension (using the predicted scale) for location. To avoid the problem of boundary effects in bin assignment, each keypoint match votes for the 2 closest bins in each dimension, giving a total of

16 entries for each hypothesis and further broadening the pose range. In most implementations of the Hough transform, a multi-dimensional array is used to represent the bins.

11. Solution for affine parameters

Using the the hough transform we are able to determine clusters of matched points who vote for the same transformation. Although the affine transformation is a good model for 3D rotation for planar surfaces being orthogonally projected, it perform poorer for 3D rotation of non-planar object. To account for these effect we should solve the fundamental matrix (Luong and Faugeras, 1996, Hartley and Zisserman, 2000). Solving this matrix theoretically demands 7 matches having the same deformation, in reality the amount of points demanded even rises. For small object finding a large amount of correct matches might pose problems. Therefore we will approximate by using the affine transformation, requiring only three correct matches. Because of this all clusters having less than 3 points are discarded, as we cannot solve for affine parameters. The disadvantage of using the affine transformation as a model is that we have to allow for large error bounds. Mostly 0.25 percent of the maximum projected dimension of the 3D object is sufficient.

The affine transformation of a model point, which we shall denote as $\begin{bmatrix} x \\ y \end{bmatrix}$ to an image point $\begin{bmatrix} u \\ v \end{bmatrix}$ can be written as

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

In the above equation, the translation is modeled by the t_i parameters and the rotation and scaling by the m_i parameters. Using the equation we can solve the unknown transformation parameters having 3 matching points (each match solving two degrees of freedom). Additional points can be added to provide more robustness. Putting this to an equation we get

$$\boldsymbol{Ax = b}$$

In which $\boldsymbol{A}$ is a matrix indicating the model points, x the unknown transformation parameters and $\boldsymbol{b}$ the image points.

Now using the least-squares solution for the parameters $\boldsymbol{x}$

$$\boldsymbol{x = [A^T A]^{-1} A^T b}$$

Which minimizes the sum of the squares of the distances from the projected model locations to the corresponding image locations .

Checking agreement between the least-squares solution we can remove outliers to increase reliability. Each time an outlier is removed the model (least-squares solution) is recalculated until the solution converges to a stable solution with all point within the error bound of the model. When the amount of matches drops below three, we can no longer solve the equation, and all the points are discarded. A point to which we have to pay attention is the fact that possible matches might have been overlooked because of similarity between the transformation bins or other errors. Because of this a top-down matching phase is used to add any other matches that agree with the projected model position. The next step would be to use some kind of probabilistic model describing the match reliability between the training image and the image.

12. Performance analysis

In this section we will explore how well the algorithm actually performs. In the images below no clustering, nor solving for affine parameters is applied. The points indicated in the images are matched on a best bin first (BBF) matching algorithm base.

12.1. Matching of a car

In the following test sequence of a car approaching the camera, frames were captured every 400 ms for a duration of 2 s. In the last frame, a template was extracted in order to allow it to match with the same car in the previous images.

figure 6: A template of a car used to match it against other images of the same car

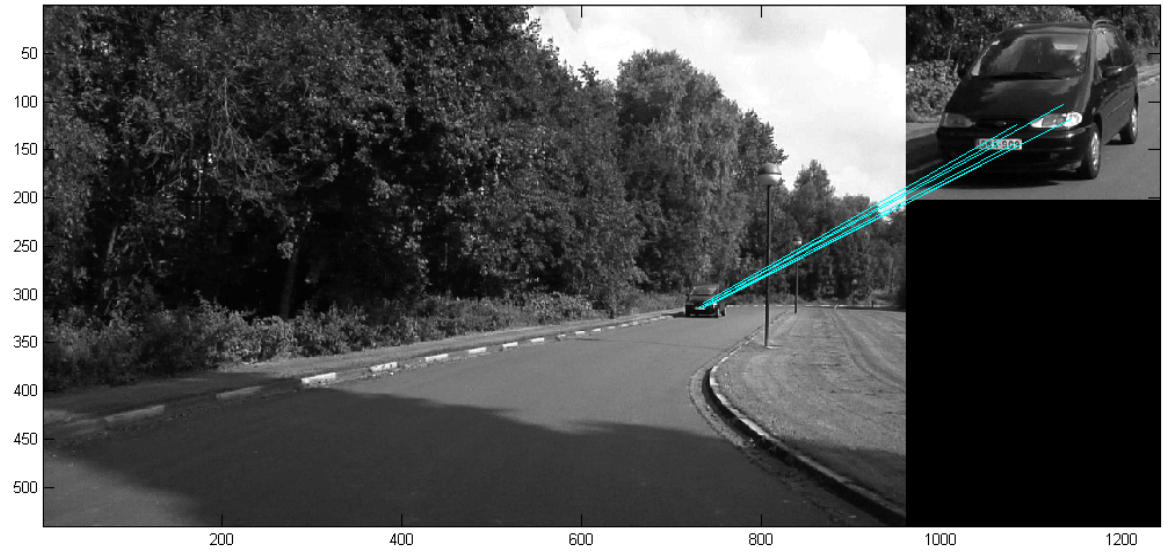

figure 7: t=0s, five points from the template are correctly identified in the scene

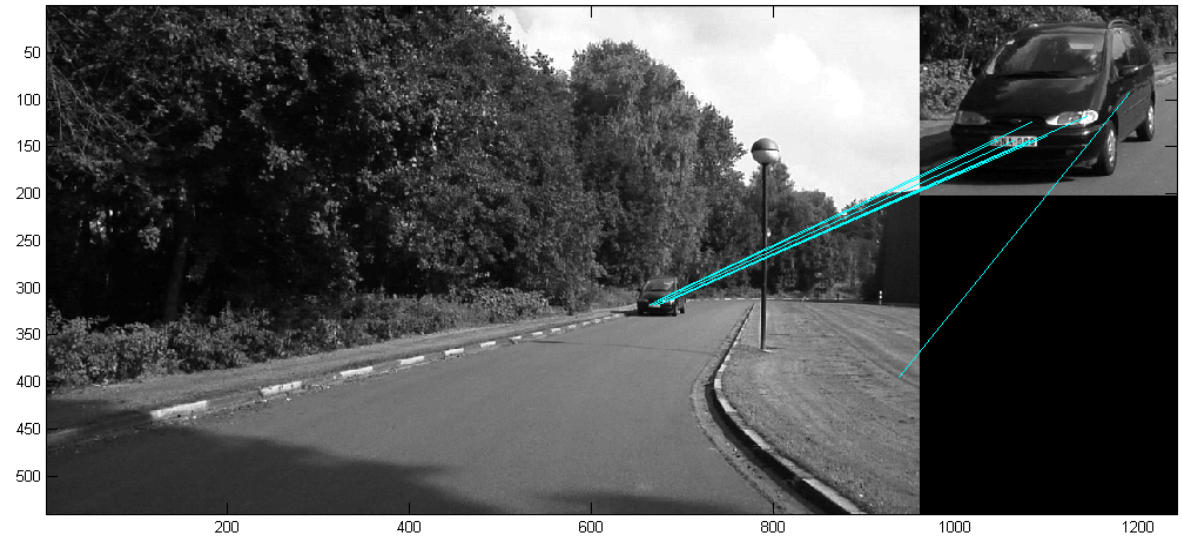

figure 8: t=400 ms, six points from the template are correctly identified in the scene, however also note the incorrect match in the right of the image

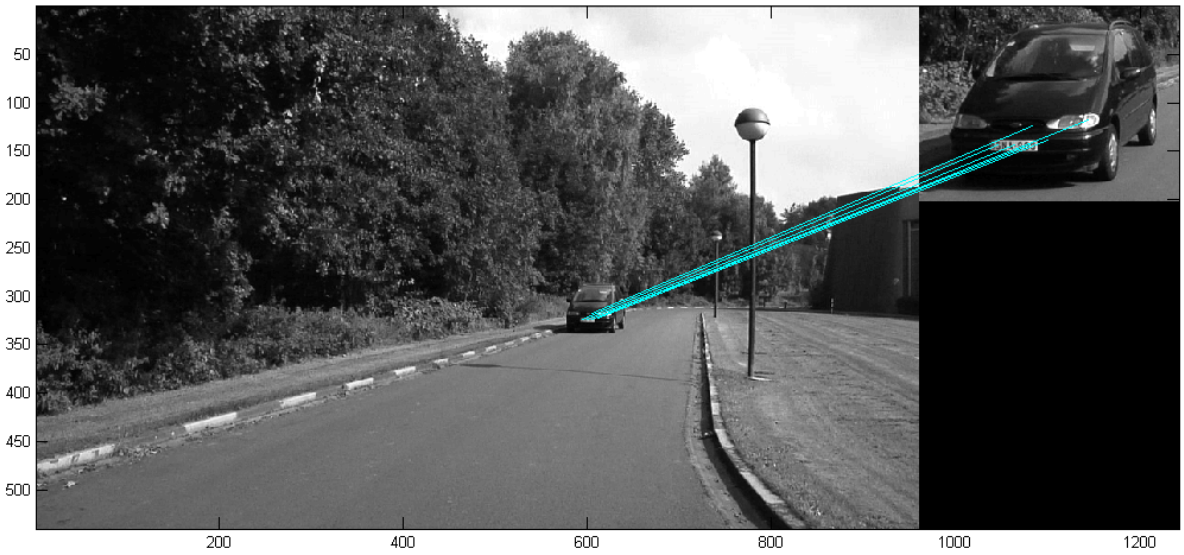

figure 9: =800ms, six points from the template are correctly identified in the scene

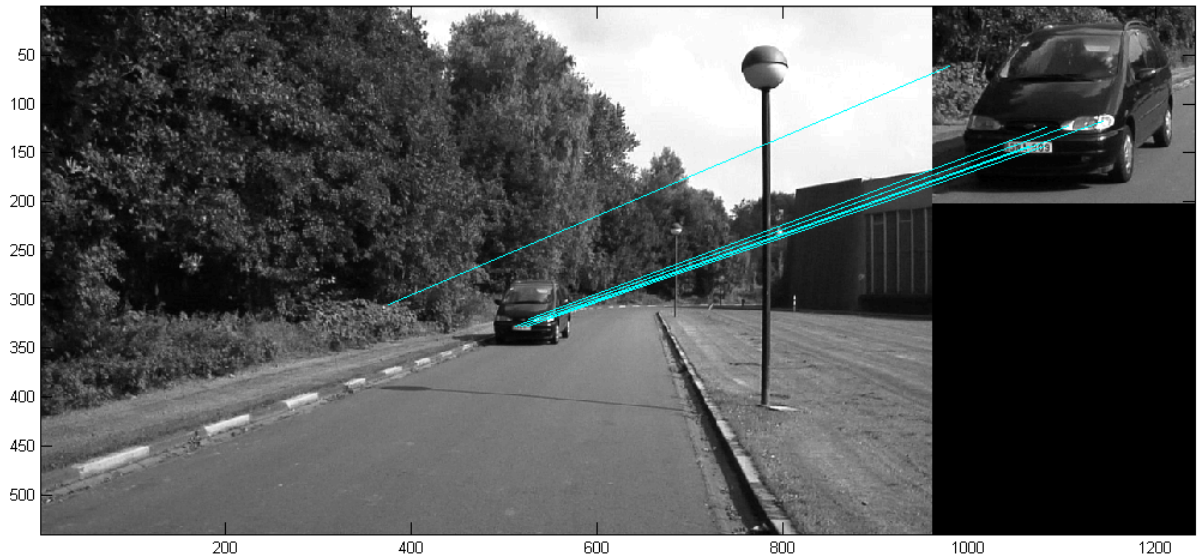

figure 10: t=1200ms, six points from the template are correctly identified in the scene (one point does not belong to the car however)

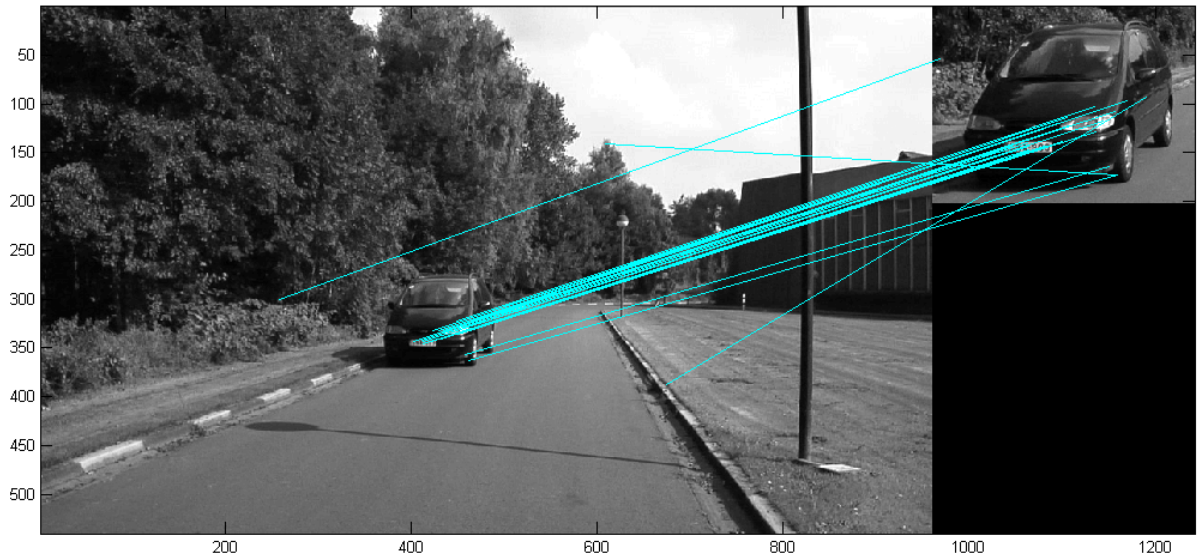

figure 11: t=1600ms, more points are being recognized, two points are wrongly matched, note that also one point from the background is correctly matched

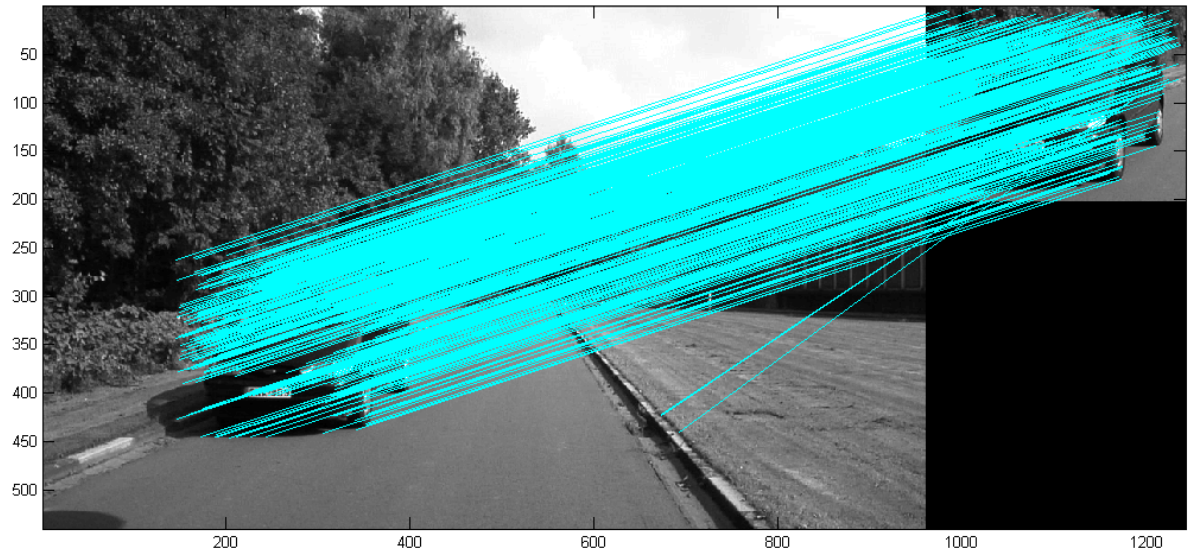

figure 12: t=2000ms, a lot of points are correctly matched (this is to be expected since the template was derived from this image). Two points are incorrectly matched

We notice that the level of errors is quite good, using the clustering techniques and the solution for affine parameters the result could be further improved. Another possibility would be to use foreground-background separation and only look within the moving region. Besides more reliability also the computational burden would be less. Although also precautions must be taken to account for foreground-background separation being in error. Other ideas to improve the reliability would be to include more points from the borders of the object. Since part (depending on the assigned scale) of the background is taken into account to describe the points at the border of the object, these points will get different descriptors as the car moves relatively to the background, thus not resulting in matches as the matching cost is too high. One way to solve this would be not to consider a Gaussian region around the point of interest, but to only take the quadrants into consideration, because in most real live situations, one quadrant is being retained as the object moves. Applied to the above images, this would mean parts of the car do not move relatively to the car. In normal situations we know this to be true. The above mentioned improvements should result in more points being matched and thus more reliably in matching objects.

We also have to note that the algorithm should be tested on far more images, while changing parameters such as, illumination, amount of cars, rotation of the car, background, … to draw more weighted conclusions. The idea behind this experiment is to serve as a proof of concept of the ability to serve for recognition of cars.

12.2. Matching of people

In the next experiment a scene of five people was constructed. The idea is to evaluate whether a person (under a variety of conditions) can be recognized with a reasonable degree of certainty. Since no clustering, nor solving for affine parameters is used in the matching process, a lot of matches will be erroneous. However, using visual inspection it is possible to get a qualitative idea of the performance of the matching process. In a first experiment we vary the angle of the person to be matched against the scene. In a second experiment we consider the influence of illumination. In the third experiment the influence of scaling is investigated. The reader should note that the parameters under investigation (i.e. viewpoint and illumination) are not completely independent in the

experiment. Independency could be improved by altering the illumination artificially using some software packet.

12.2.1. Experiment 1: the influence of viewpoint rotation of the face of the testperson on the matching process.

In this experiment, the viewpoint of the face of five test persons is observed under different angles.

Following viewpoints are considered:

1. Frontal view
2. 20 degrees right
3. 45 degrees right
4. 60 degrees right
5. 90 degrees right
6. 20 degrees left
7. 45 degrees left
8. 90 degrees left

Looking at the images we can conclude the matching process works quite well for frontal views (note the matching can even be improved using clustering techniques and solving for the affine parameters). However viewpoints of 20 degrees or more result in erroneous results. This probably is the result of too little stable keypoints. Using the quadrant technique as described in the previous section should result in more keypoints. Having more keypoints we can use a stronger criterion to allow less false positives in the matching process. Recapitulate we needs a least 3 correct matches for calculating the affine approximation of the deformation process. Looking at the images the reader may notice a discrepancy between the amount of correct matches in the left and right change of viewpoint by 20 degrees. Taking into account the limited amount of test images and the fact that the parameters influencing the matching process are not fully independent in the experiment, no valuable conclusions can be drawn considering this discrepancy.

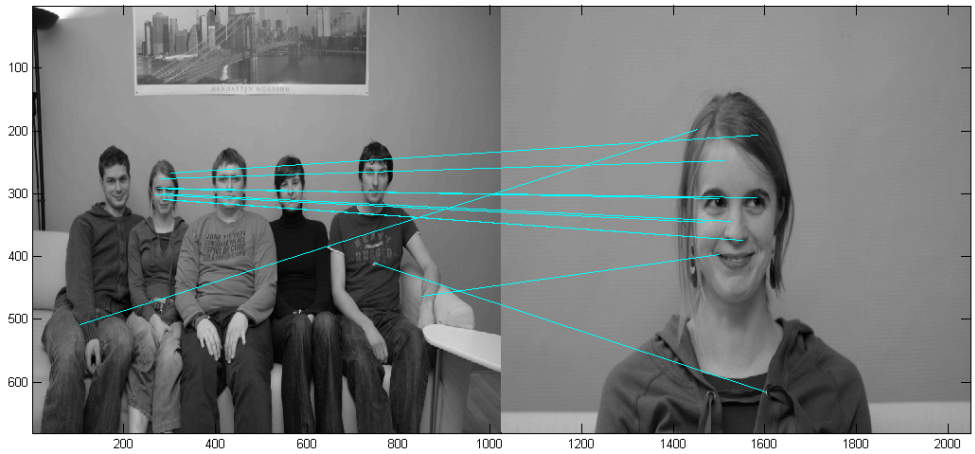

figure 13: matching result of scene and test person 1, frontal view

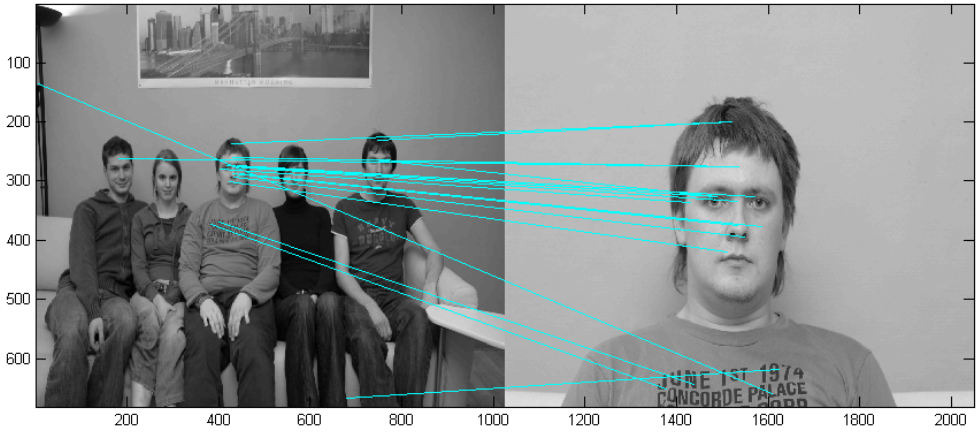

figure 14: matching result of scene and test person 2, frontal view

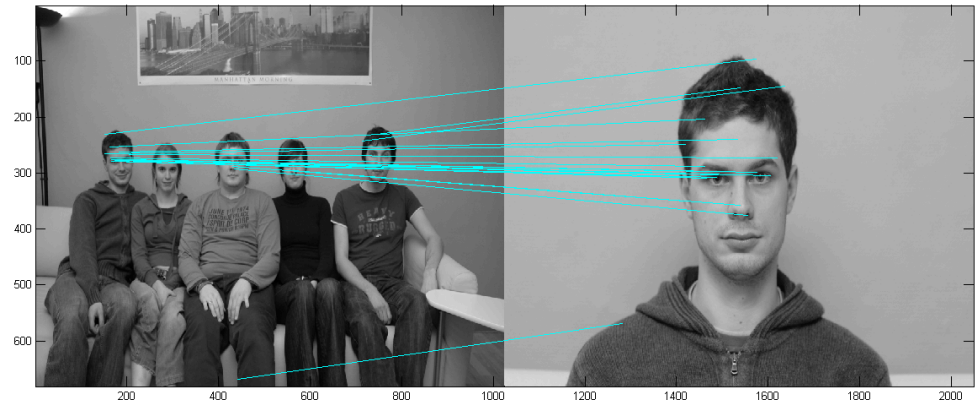

figure 15: matching result of scene and test person 3, frontal view

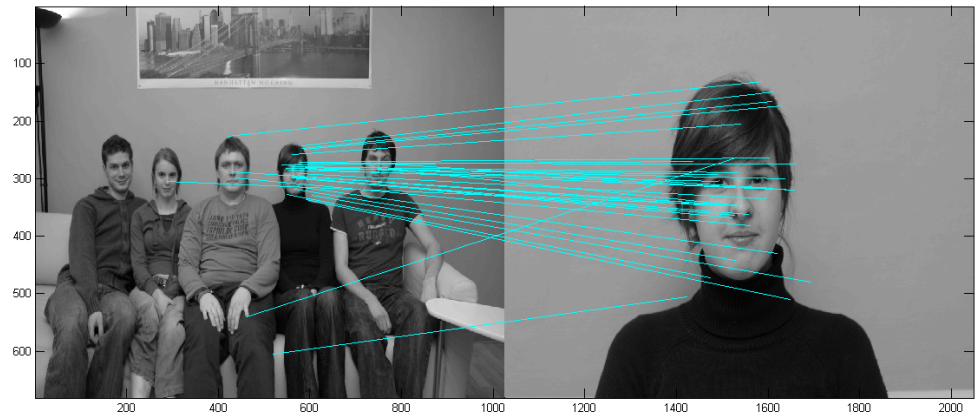

figure 16: matching result of scene and test person 4, frontal view

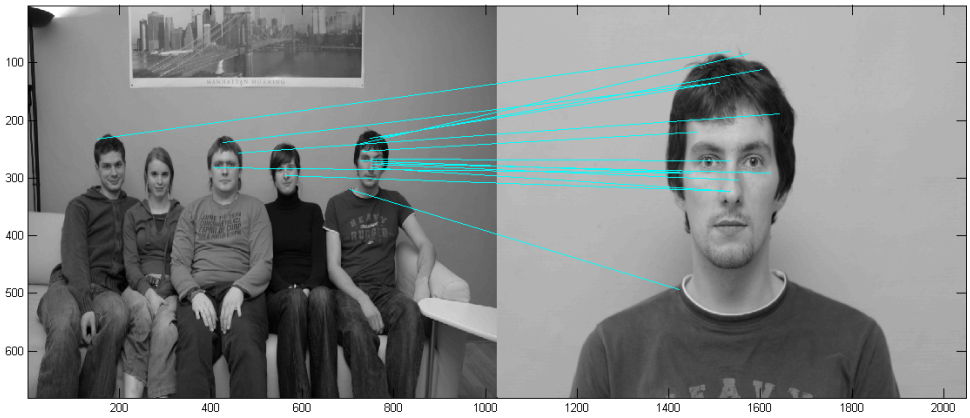

figure 17: matching result of scene and test person 5, frontal view.

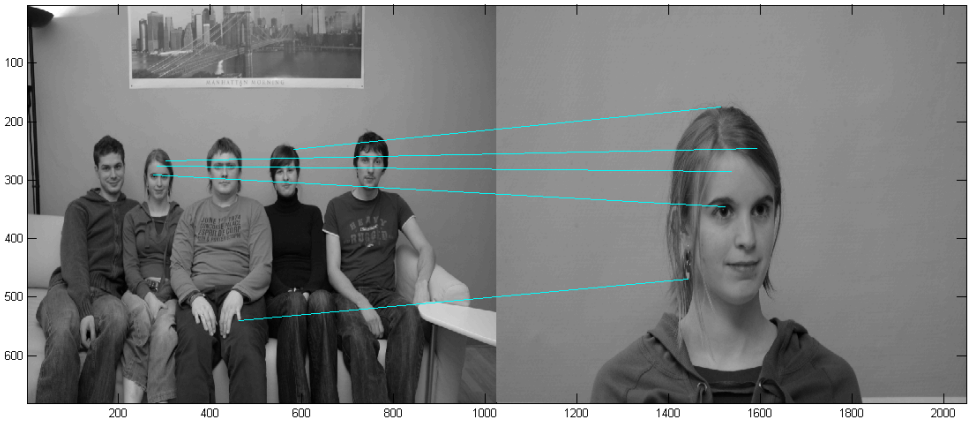

figure 18 matching result of scene and test person 1, view 20 degrees right

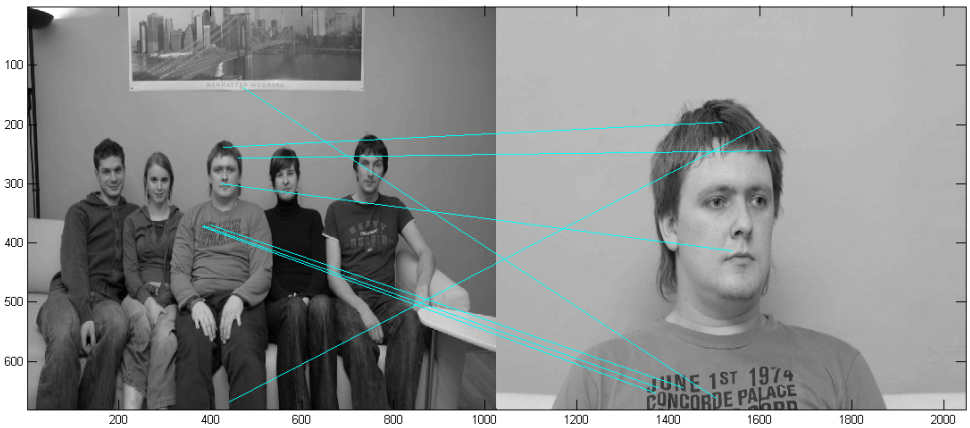

figure 19: matching result of scene and test person 2, view 20 degrees right

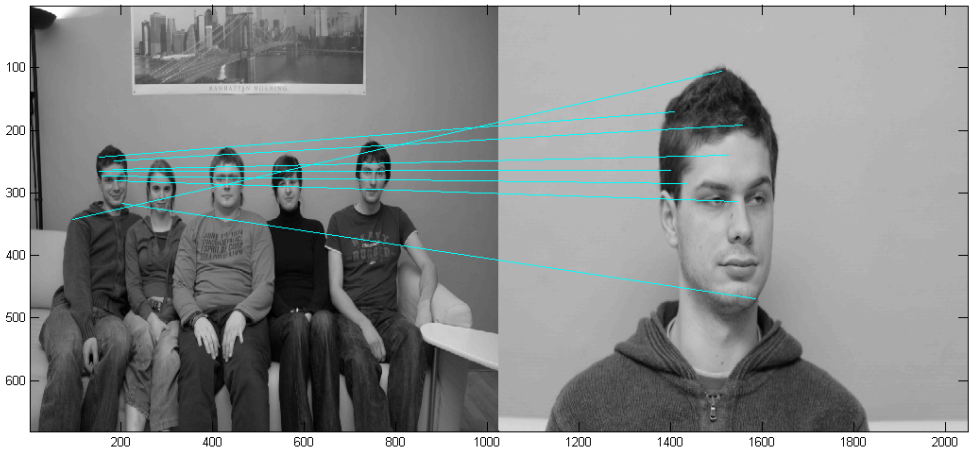

figure 20: matching result of scene and test person 3, view 20 degrees right

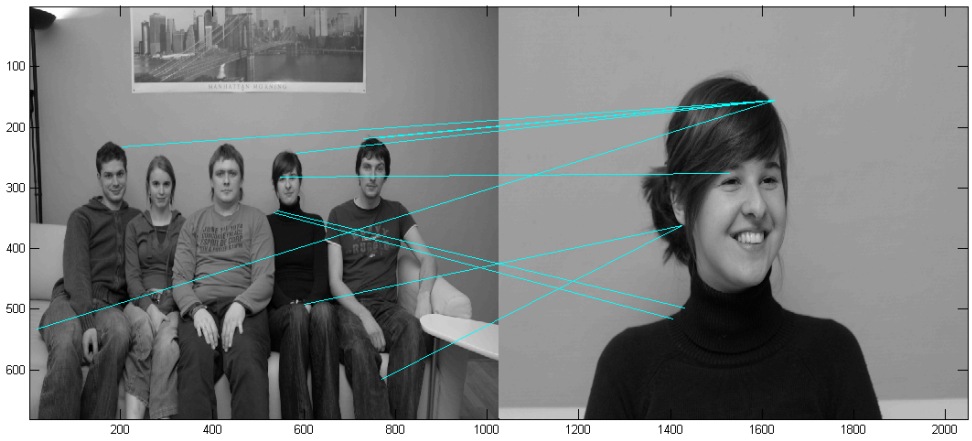

figure 21: matching result of scene and test person 4, view 20 degrees right

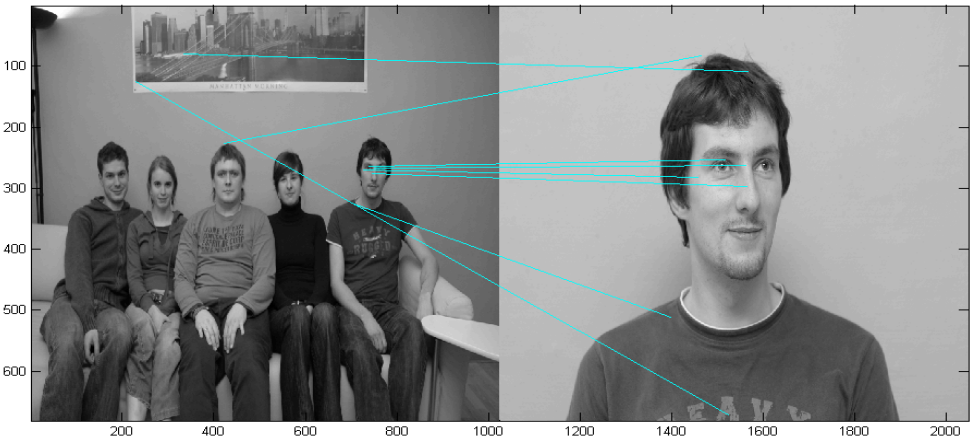

figure 22: matching result of scene and test person 5, view 20 degrees right

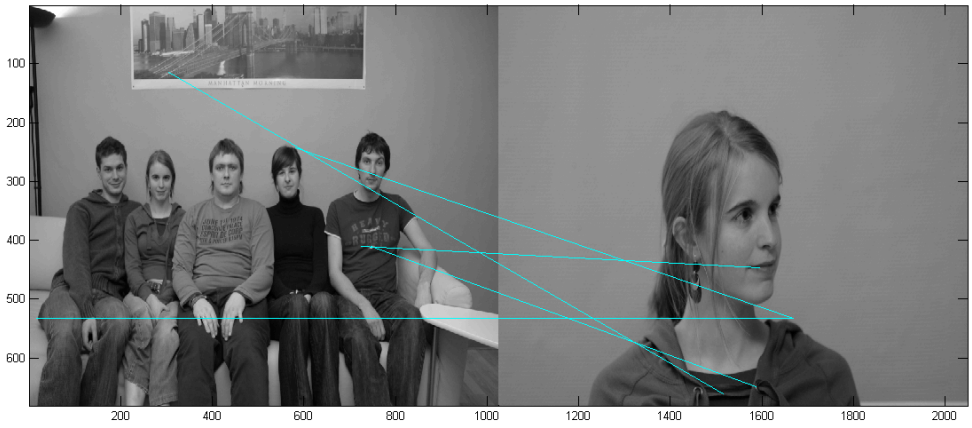

figure 23: matching result of scene and test person 1, view 45 degrees right

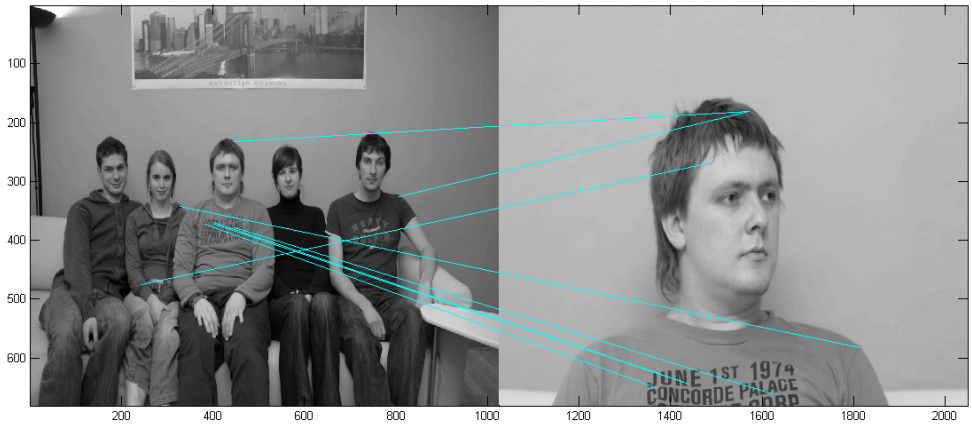

figure 24: matching result of scene and test person 2, view 45 degrees right

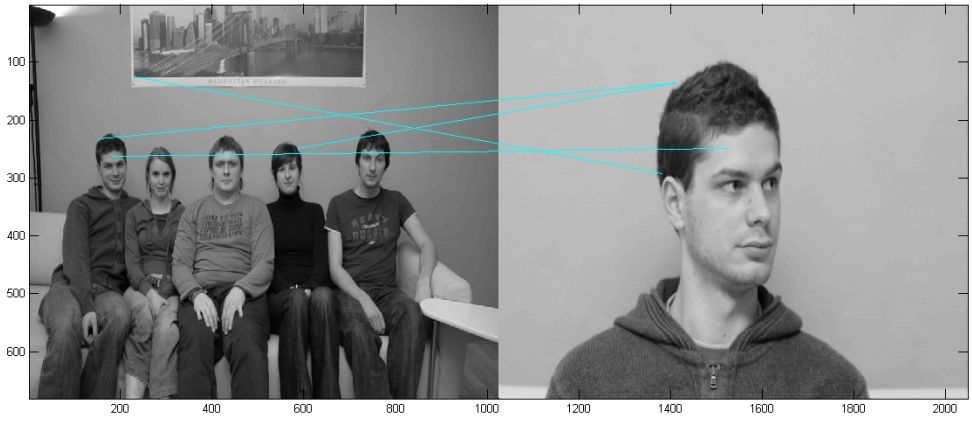

figure 25: matching result of scene and test person 3, view 45 degrees right

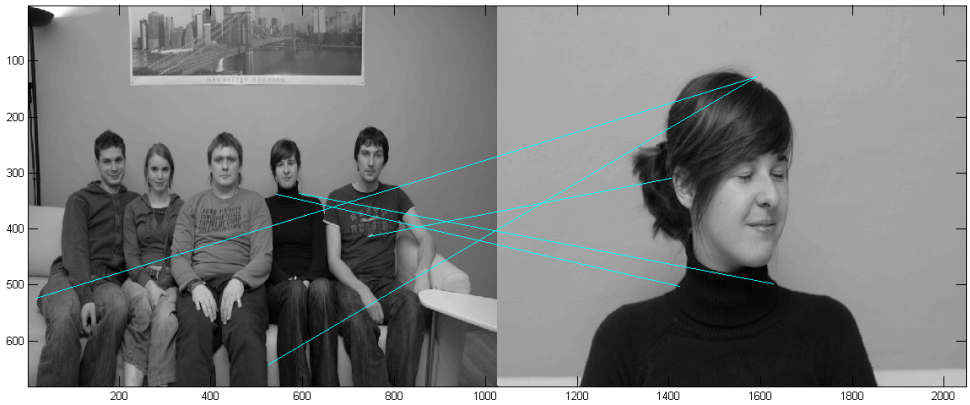

figure 26: matching result of scene and test person 4, view 45 degrees right

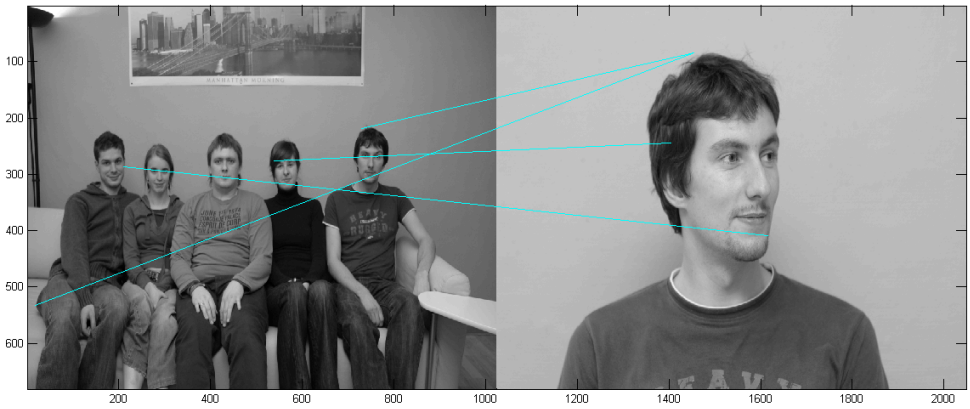

figure 27: matching result of scene and test person 5, view 45 degrees right

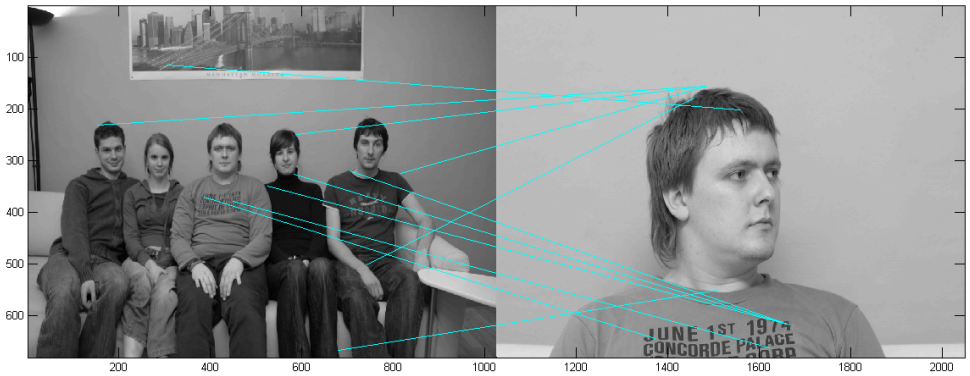

figure 28: matching result of scene and test person 2, view 60 degrees right

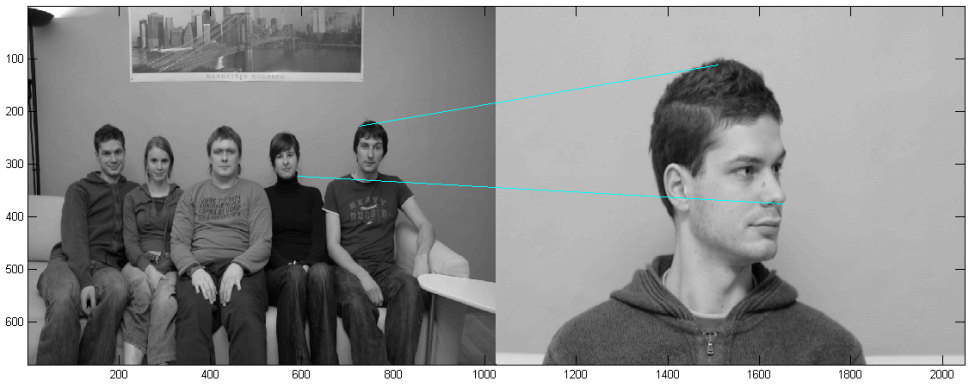

figure 29: matching result of scene and test person 3, view 60 degrees right

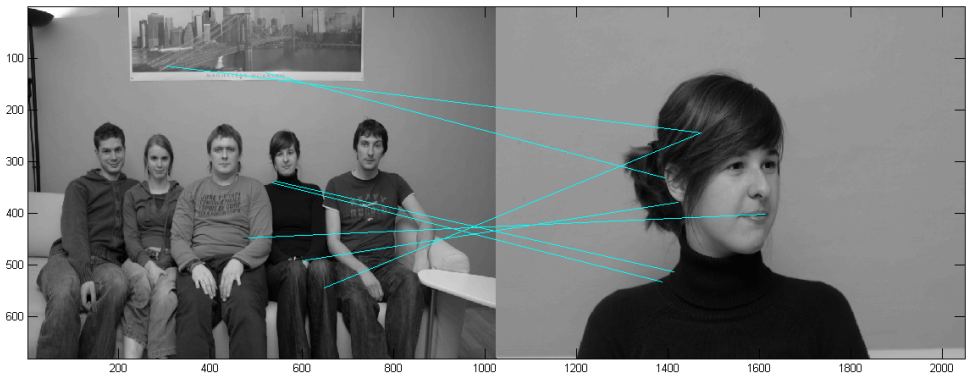

figure 30: matching result of scene and test person 4, view 60 degrees right

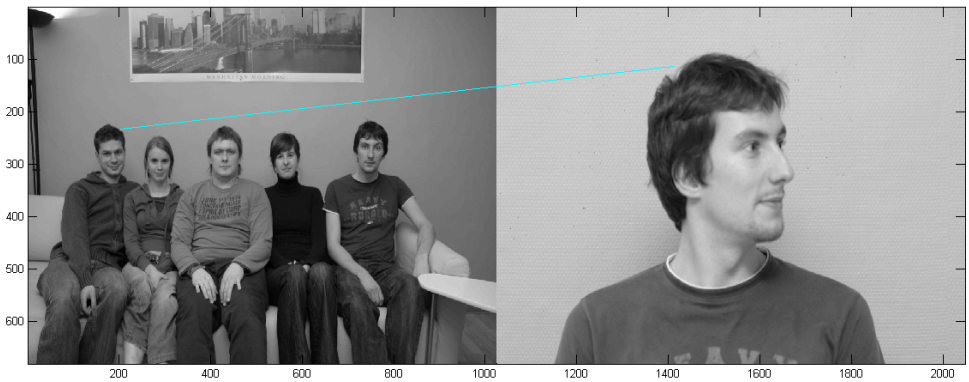

figure 31: matching result of scene and test person 5, view 60 degrees right

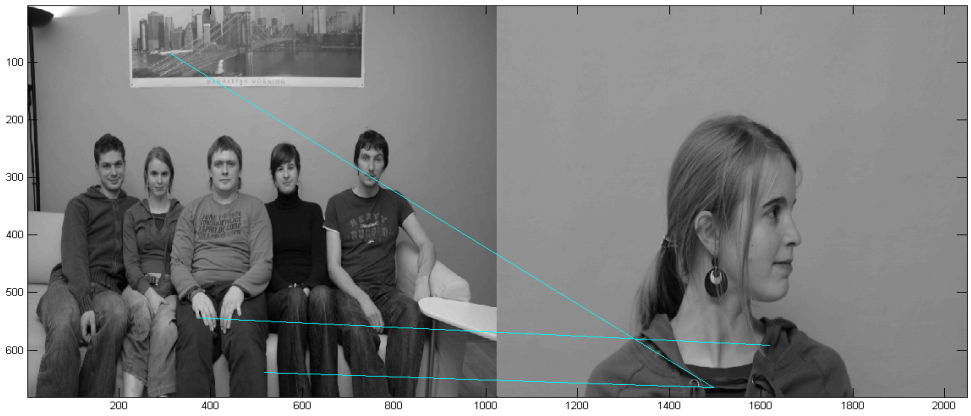

figure 32: matching result of scene and test person 1, view 90 degrees right

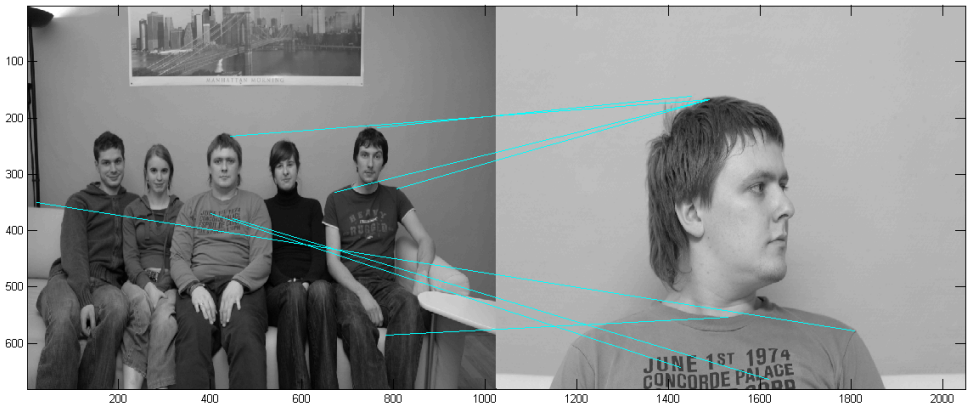

figure 33: matching result of scene and test person 2, view 90 degrees right

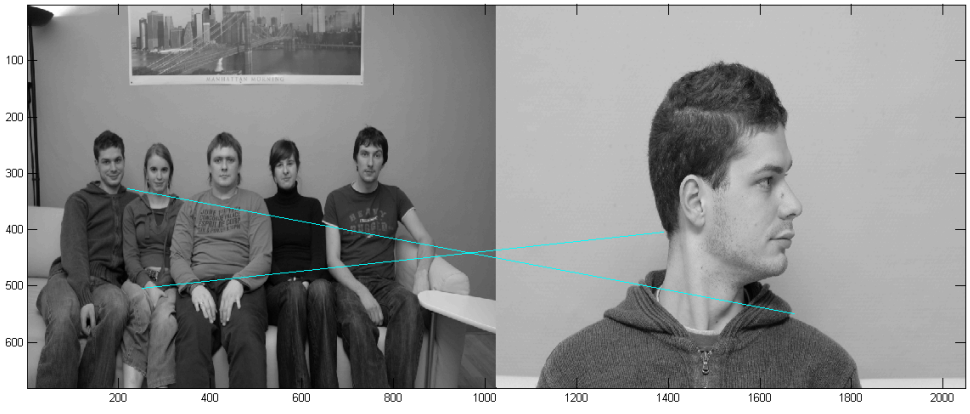

figure 34: matching result of scene and test person 3, view 90 degrees right

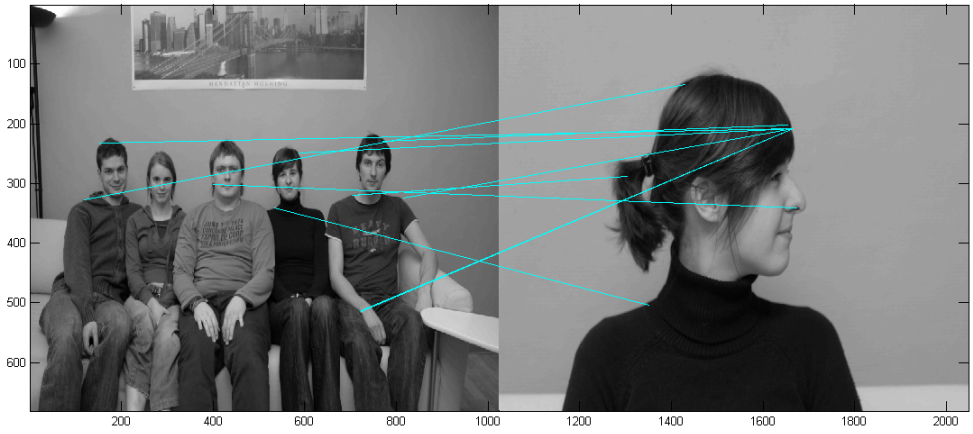

figure 35: matching result of scene and test person 4, view 90 degrees right

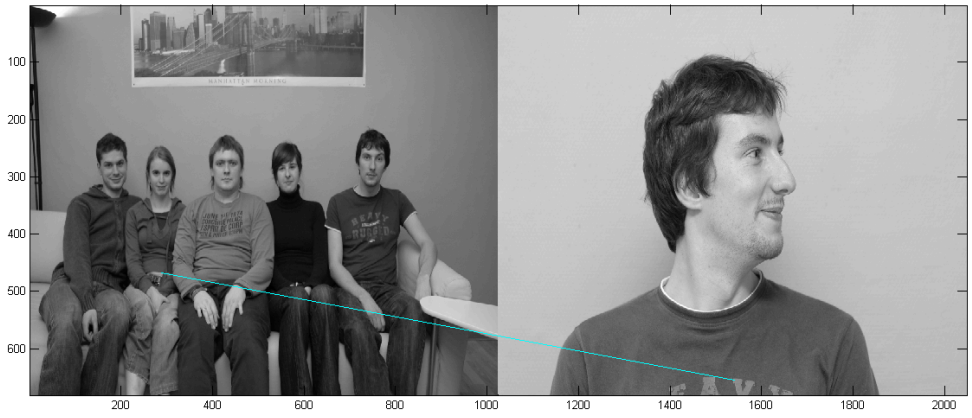

figure 36 matching result of scene and test person 5, view 90 degrees right

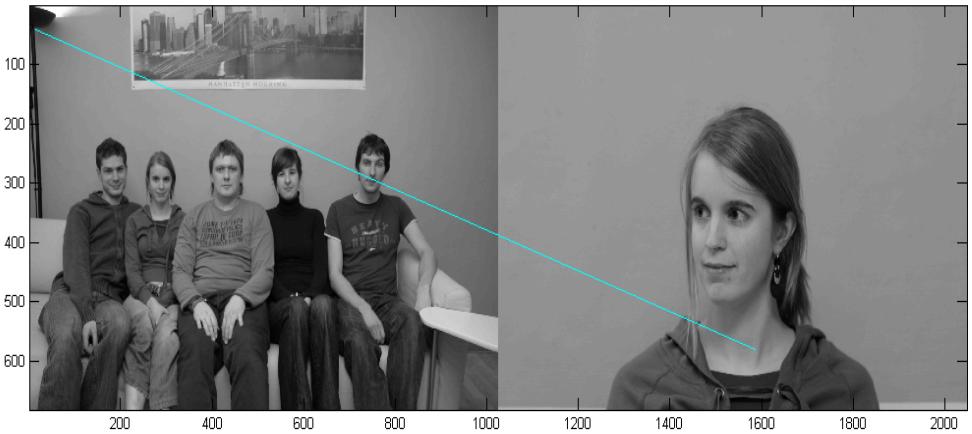

figure 37: matching result of scene and test person 1, view 20 degrees left

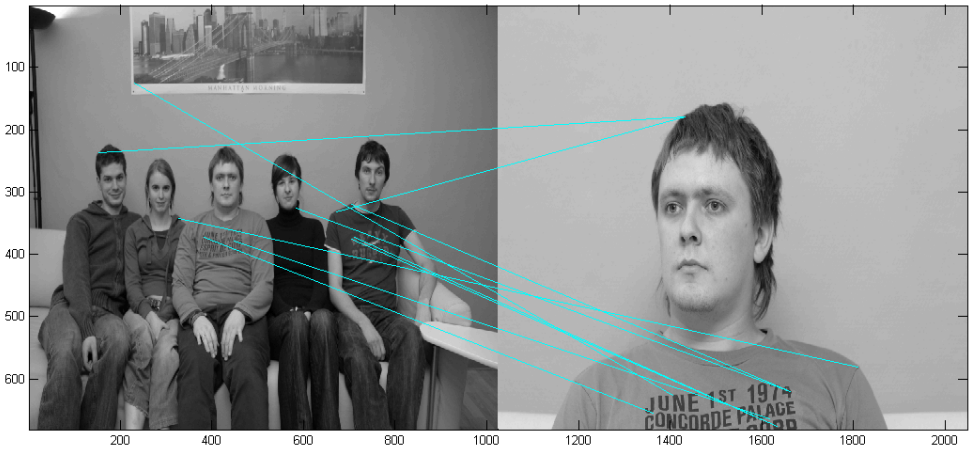

figure 38: matching result of scene and test person 2, view 20 degrees left

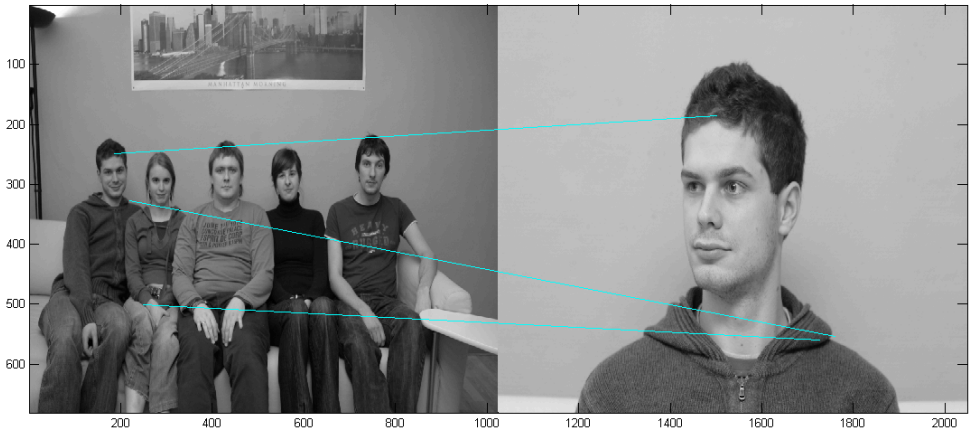

figure 39: matching result of scene and test person 3, view 20 degrees left

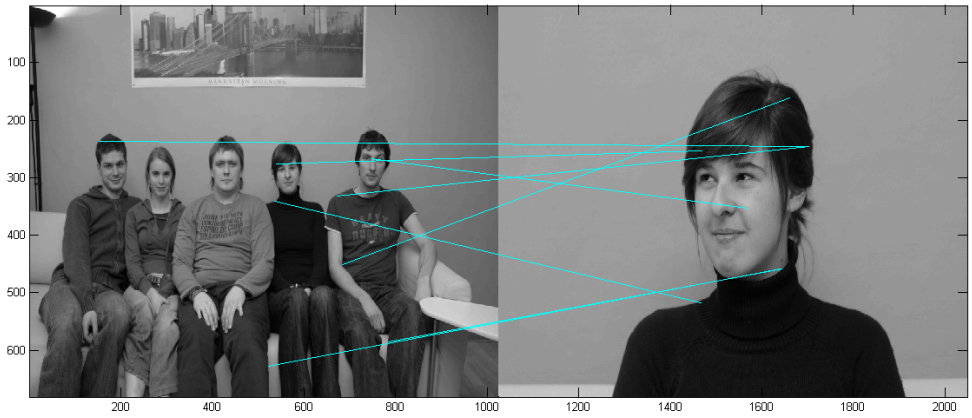

figure 40: matching result of scene and test person 4, view 20 degrees left

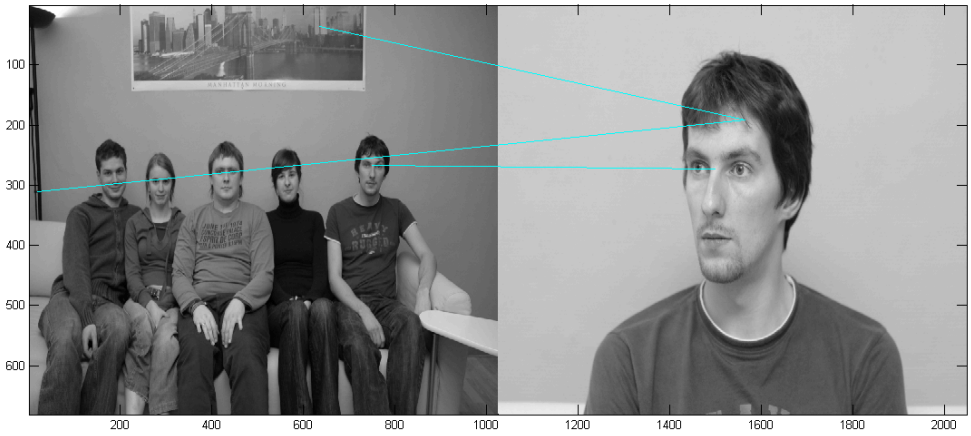

figure 41: matching result of scene and test person 5, view 20 degrees left

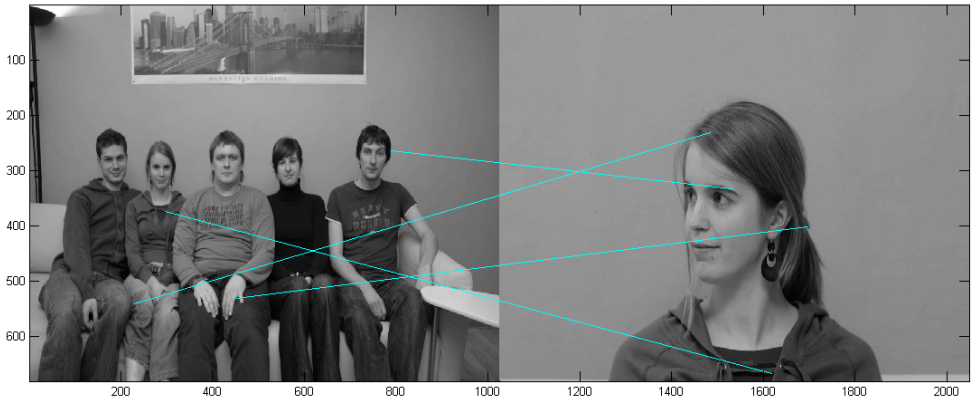

figure 42 matching result of scene and test person 1, view 45 degrees left

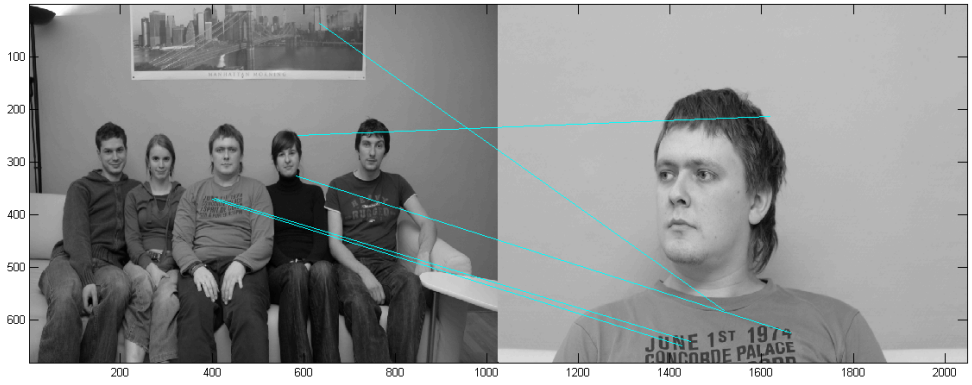

figure 43: matching result of scene and test person 2, view 45 degrees left

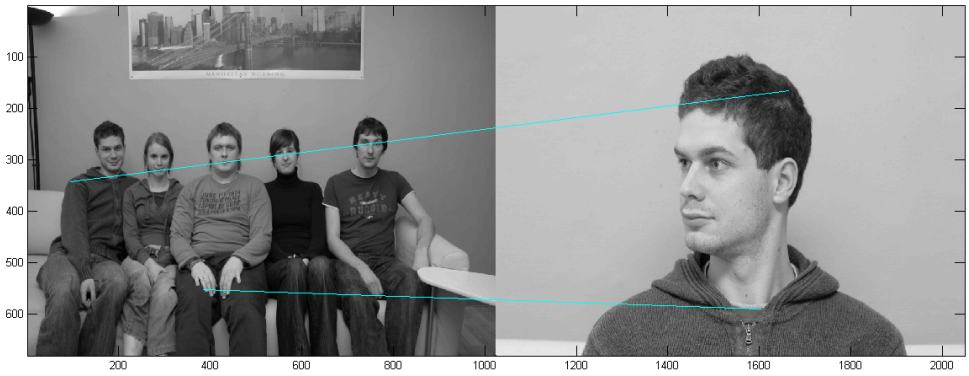

figure 44: matching result of scene and test person 3, view 45 degrees left

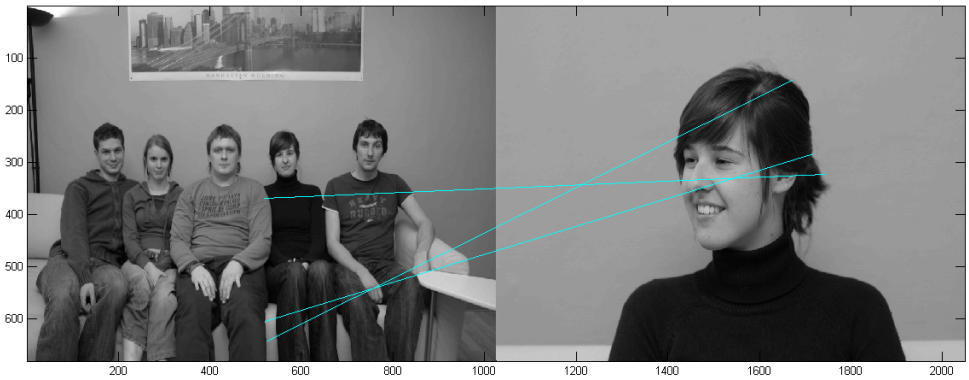

figure 45: matching result of scene and test person 4, view 45 degrees left

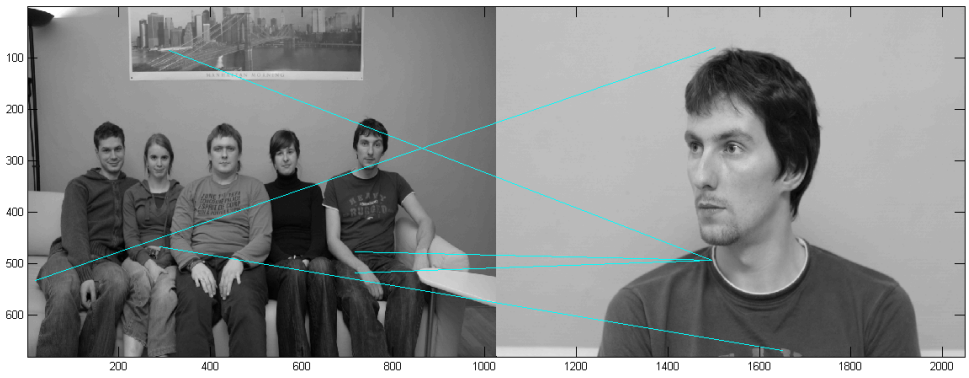

figure 46: matching result of scene and test person 5, view 45 degrees left

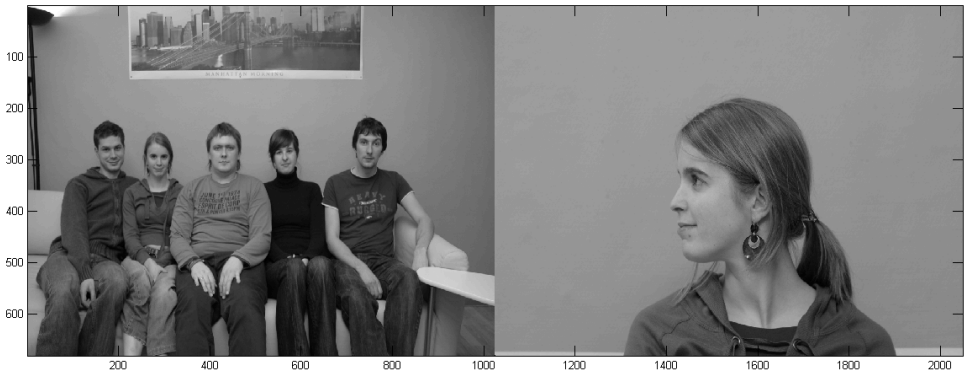

figure 47: matching result of scene and test person 1, view 90 degrees left

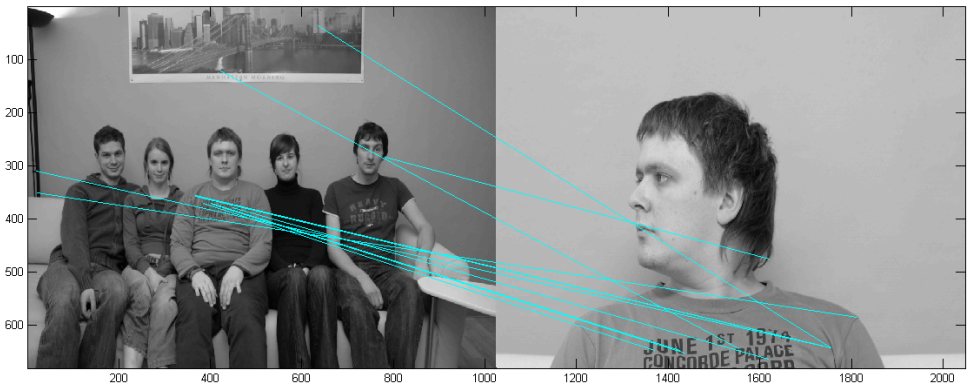

figure 48: matching result of scene and test person 2, view 90 degrees left

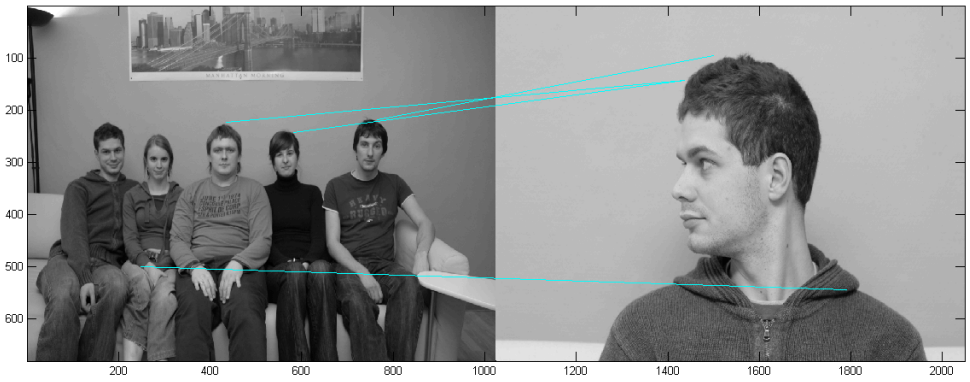

figure 49: matching result of scene and test person 3, view 90 degrees left

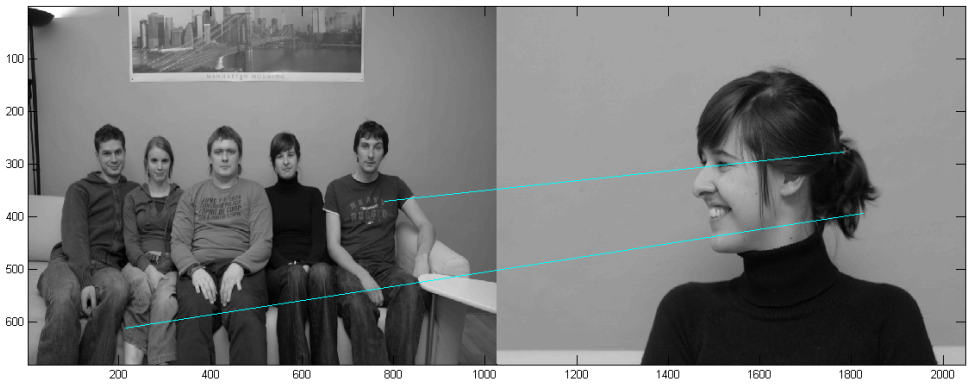

figure 50: matching result of scene and test person 4, view 90 degrees left

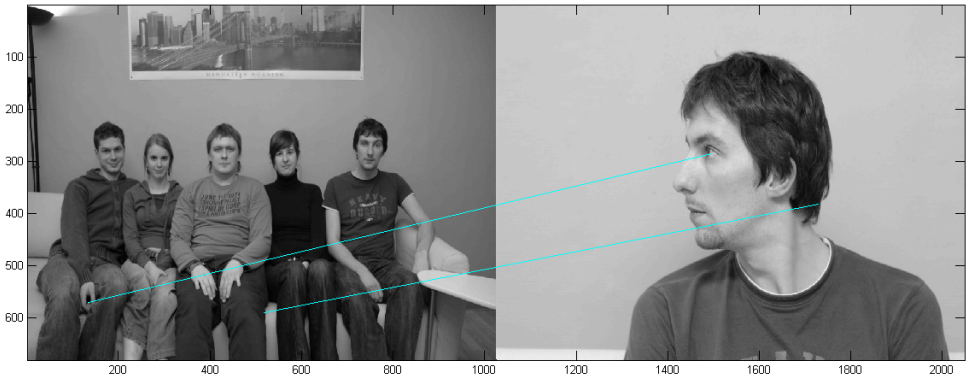

figure 51: matching result of scene and test person 5, view 90 degrees left

12.2.2. Experiment 2: the influence of change in illumination of the test person on the matching process.

In the second experiment the influence of illumination on the matching process was considered. Each test person in frontal view was photographed three times using a different ISO value, varying from normal illumination to dark illumination. Looking at the match results below, the influence of change in illumination is minimal to some point, however there seems to be a tendency that the amount of correct matches drops as the difference in illumination becomes larger.

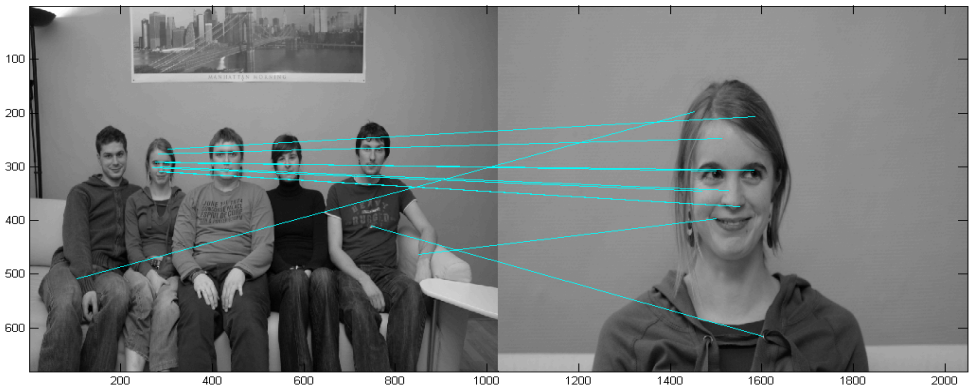

figure 52: matching result of scene and test person 1, frontal view, normal illumination

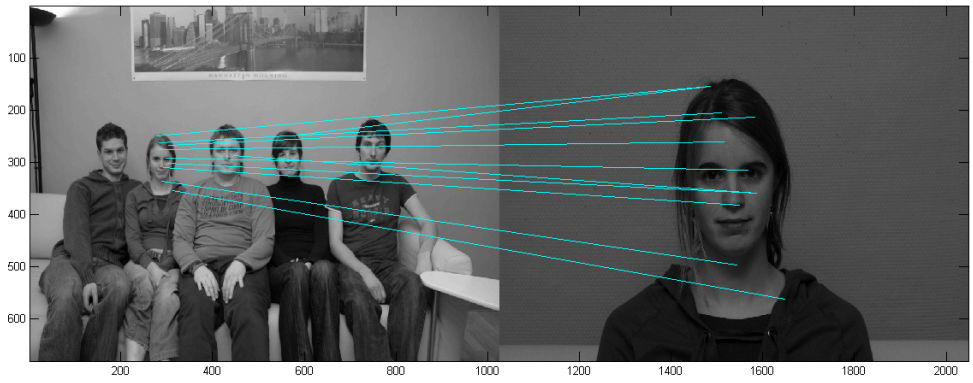

figure 53 matching result of scene and test person 1, frontal view, dark illumination

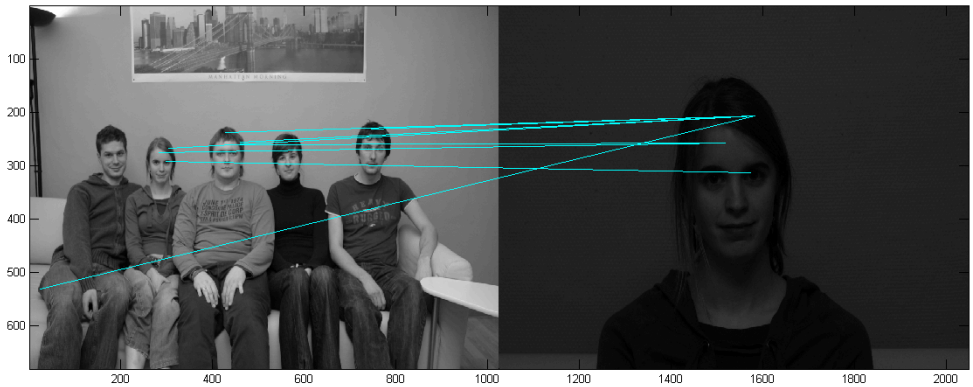

figure 54: matching result of scene and test person 1, frontal view, darker illumination

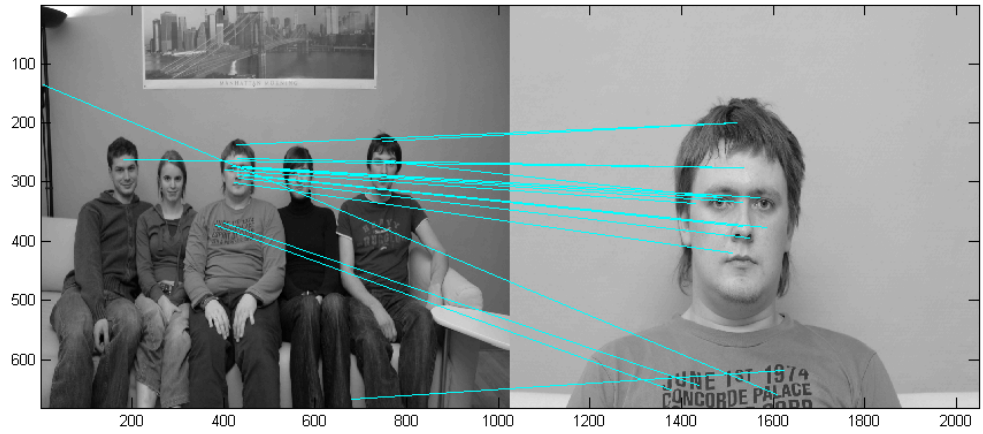

figure 55 matching result of scene and test person 2, frontal view, normal illumination

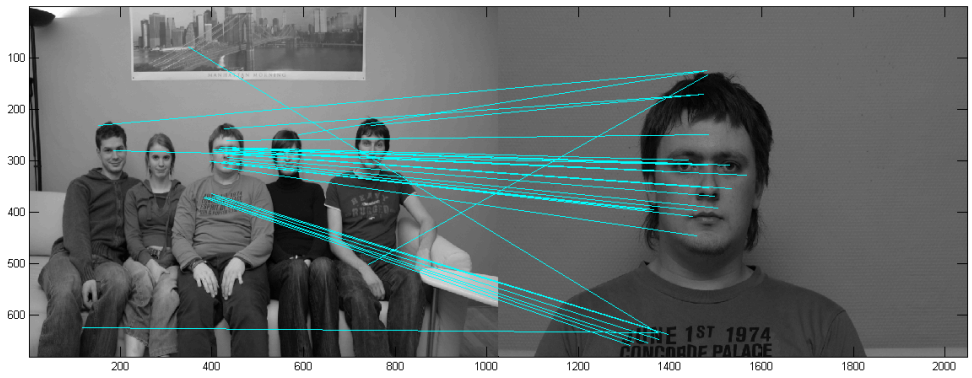

figure 56: matching result of scene and test person 2, frontal view, dark illumination

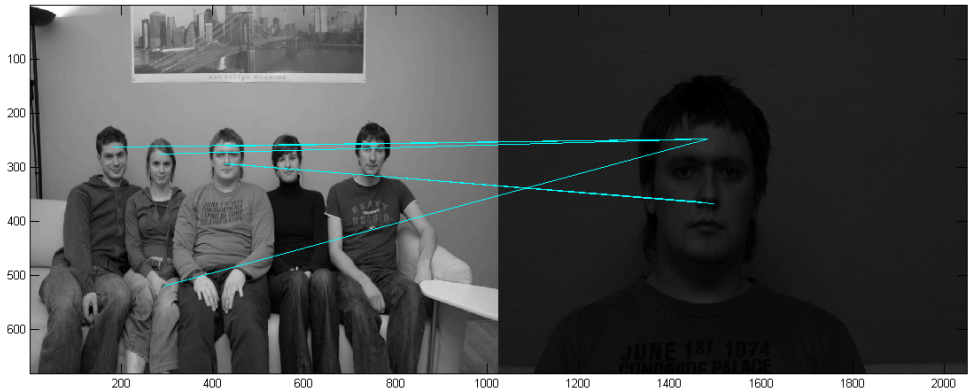

figure 57: matching result of scene and test person 2, frontal view, darker illumination

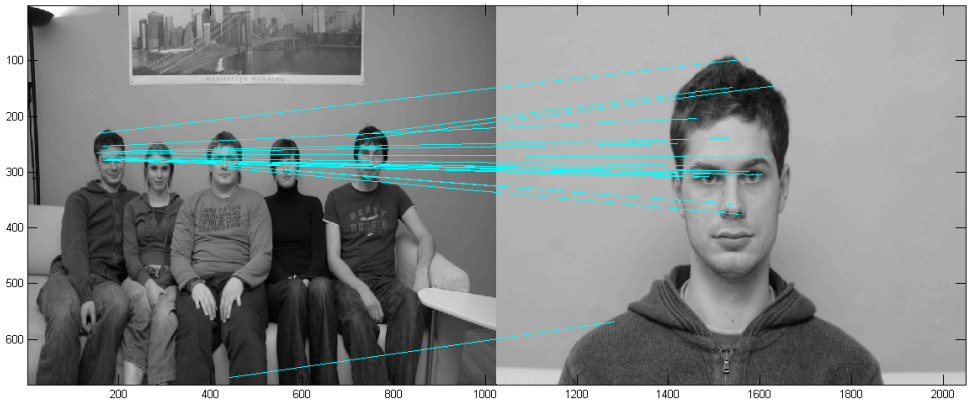

figure 58: matching result of scene and test person 3, frontal view, normal illumination

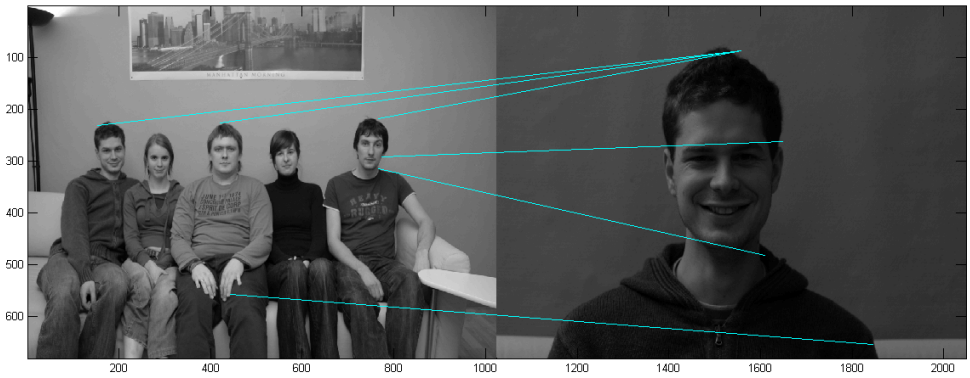

figure 59: matching result of scene and test person 3, frontal view, dark illumination

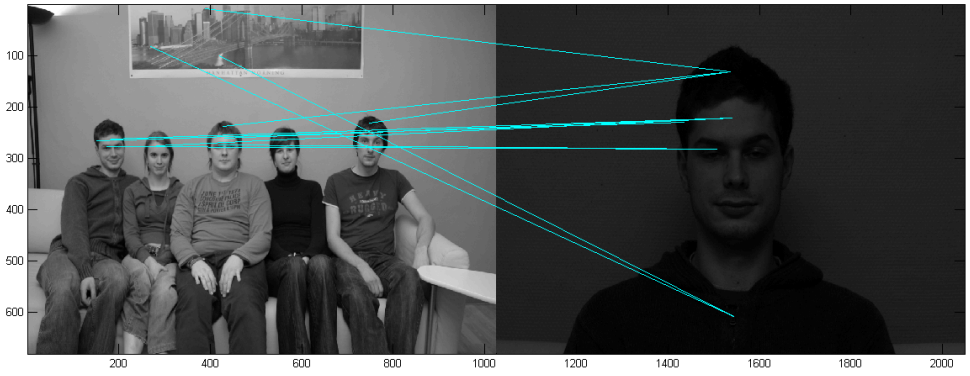

figure 60: matching result of scene and test person 3, frontal view, darker illumination

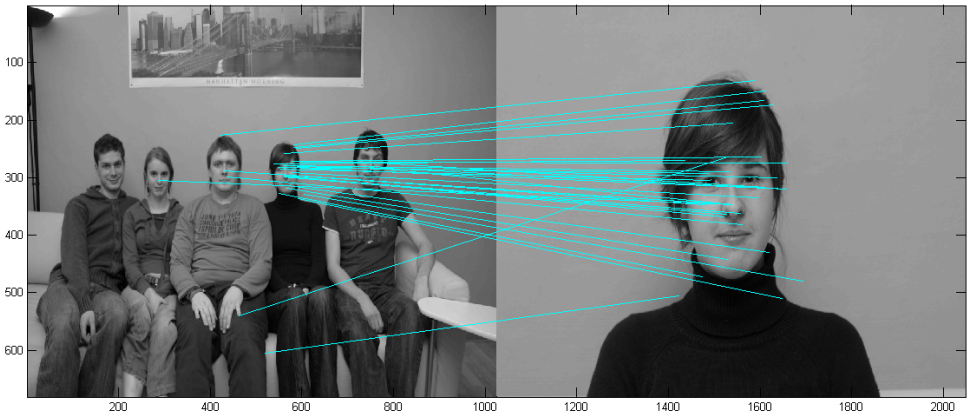

figure 61: matching result of scene and test person 4, frontal view, normal illumination

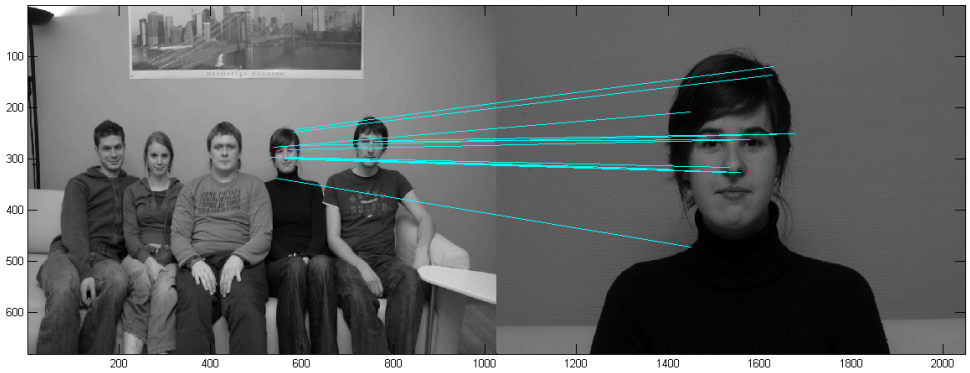

figure 62: matching result of scene and test person 4, frontal view, dark illumination

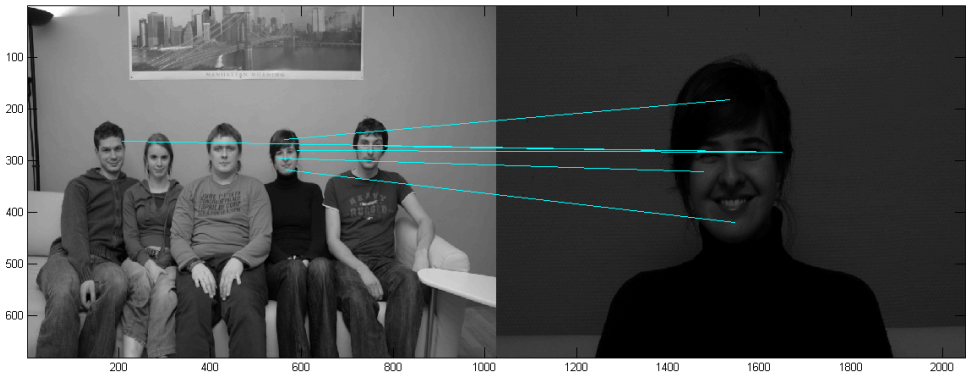

figure 63: matching result of scene and test person 4, frontal view, darker illumination

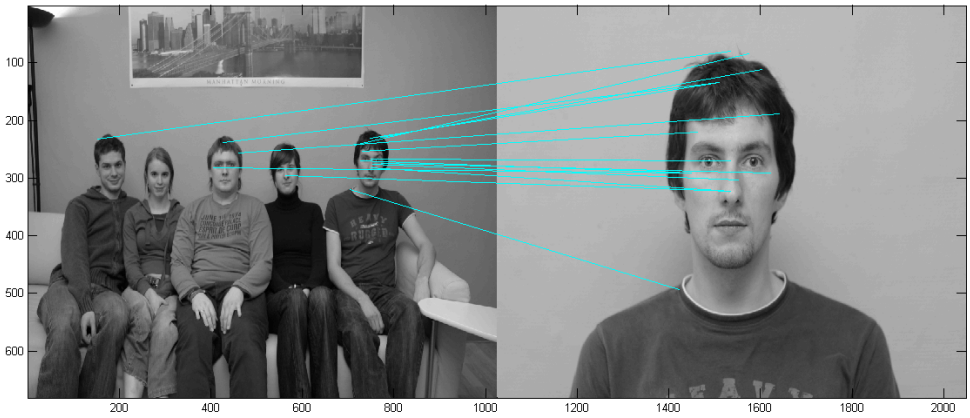

figure 64: matching result of scene and test person 5, frontal view, normal illumination

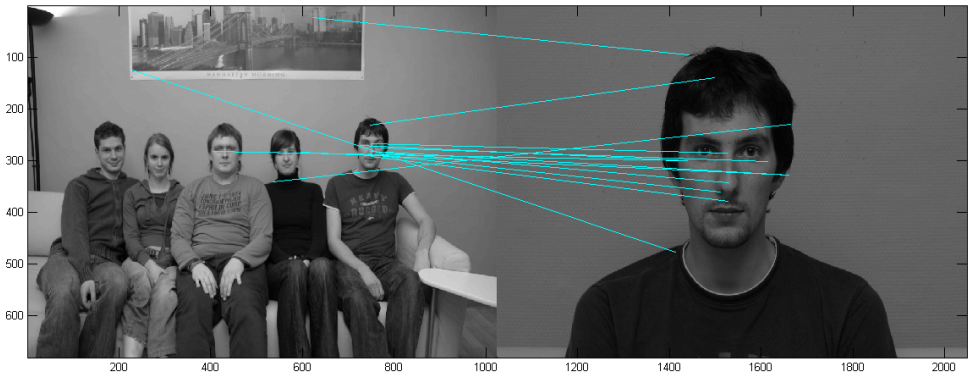

figure 65: matching result of scene and test person 5, frontal view, dark illumination

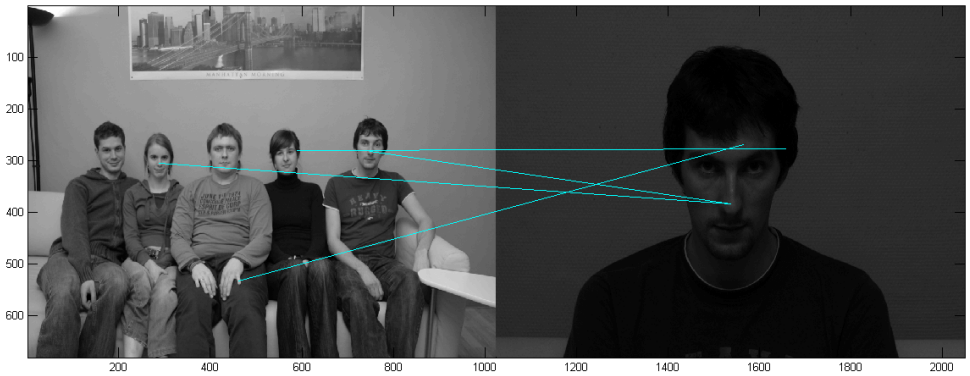

figure 66: matching result of scene and test person 5, frontal view, darker illumination

12.2.3. Experiment 3: the influence of change in scaling of the test person on the matching process.

In the third experiment the influence of scaling the test person on the matching process is investigated. Taking into account the limited amount of images and the strong influence of other

parameters - such as change in viewpoint (test person 5) or facial expression (test person 1) – we have to conclude the experiment should be repeated using artificial scaling.

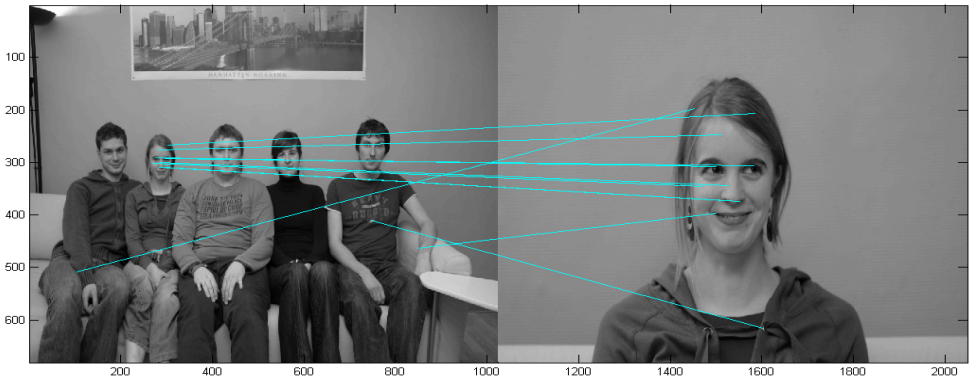

figure 67: matching result of scene and test person 1, frontal view, normal distance

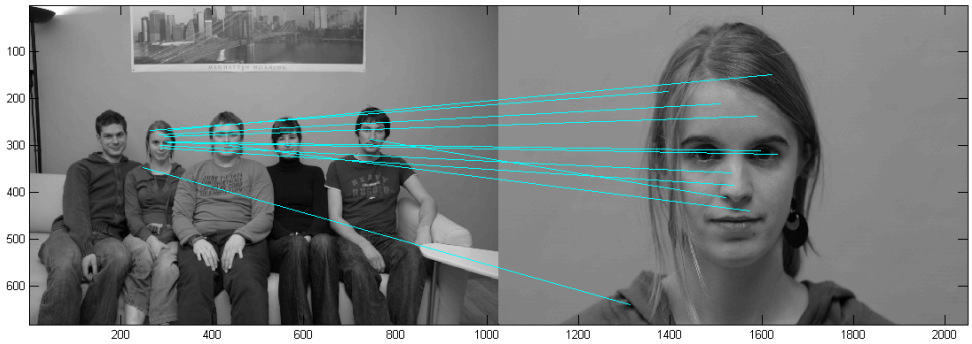

figure 68: matching result of scene and test person 1, frontal view, close distance

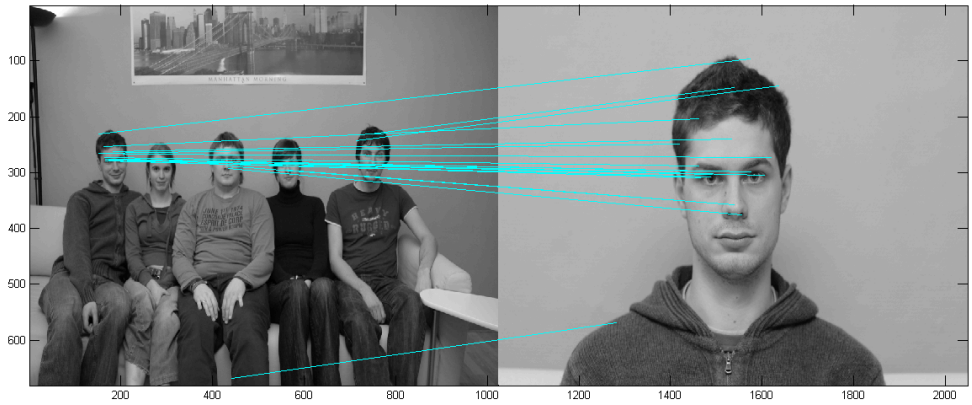

figure 69: matching result of scene and test person 3, frontal view, normal distance

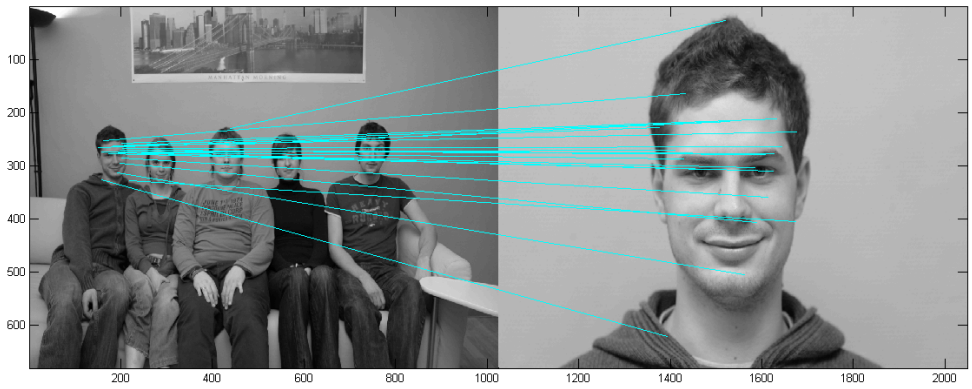

figure 70: matching result of scene and test person 3, frontal view, close distance

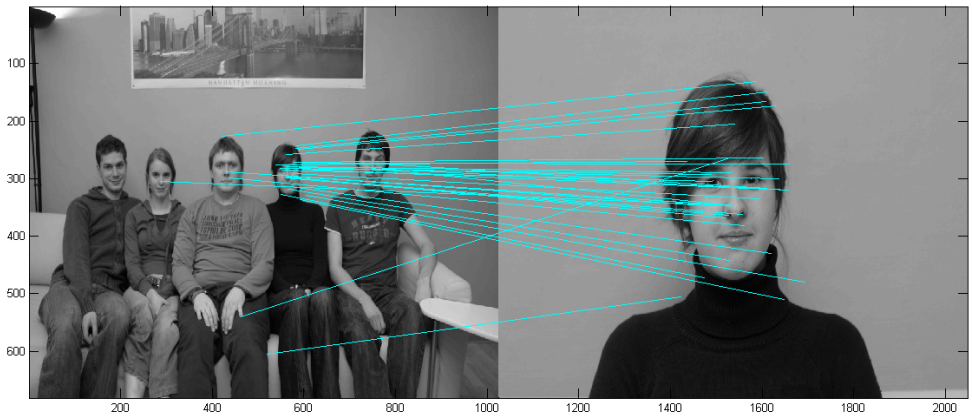

figure 71: matching result of scene and test person 4, frontal view, normal distance

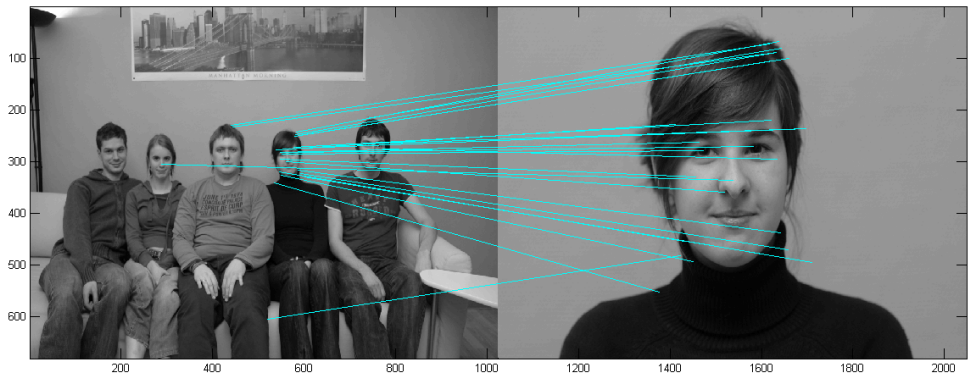

figure 72: matching result of scene and test person 4, frontal view, close distance

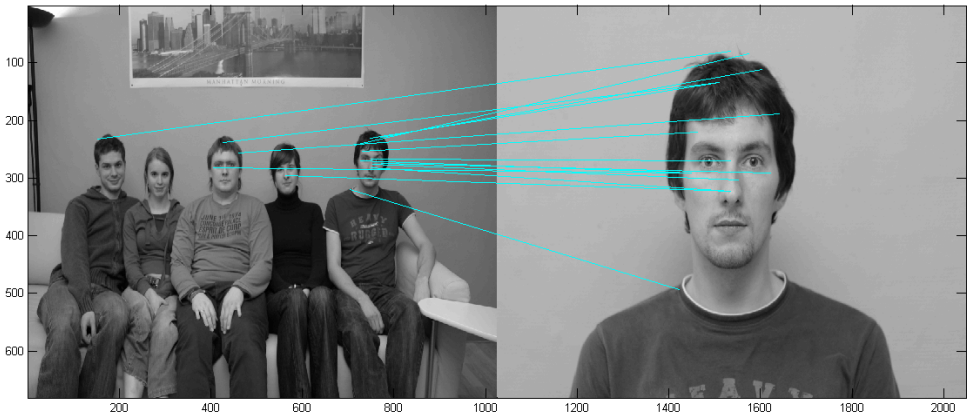

figure 73: matching result of scene and test person 5, frontal view, normal distance

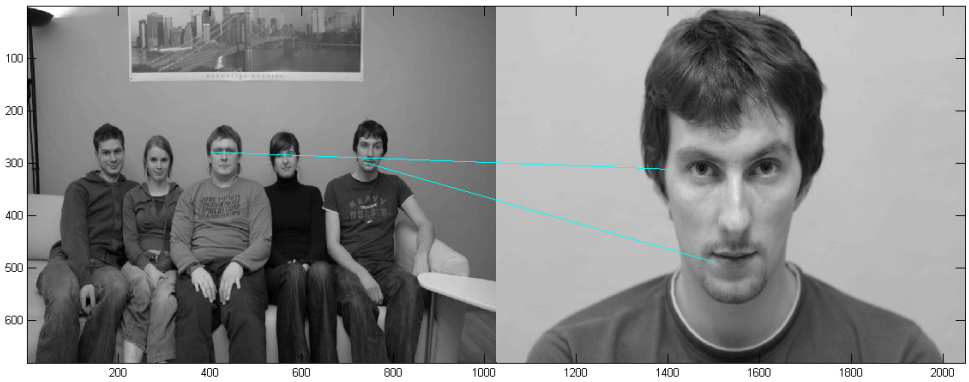

figure 74: matching result of scene and test person 5, frontal view, close distance

12.2.4. Experiment 4: validation of the matching process under a variety of image deformations and influences.

In this experiment we tried to match a person in a scene undergoing the following effects:

1. Rotation
2. Scaling
3. Change of viewpoint
4. Change of illumination

It is clear the matching process fails to recognize the person, although it should be noticed that even for humans the matching is not straightforward.

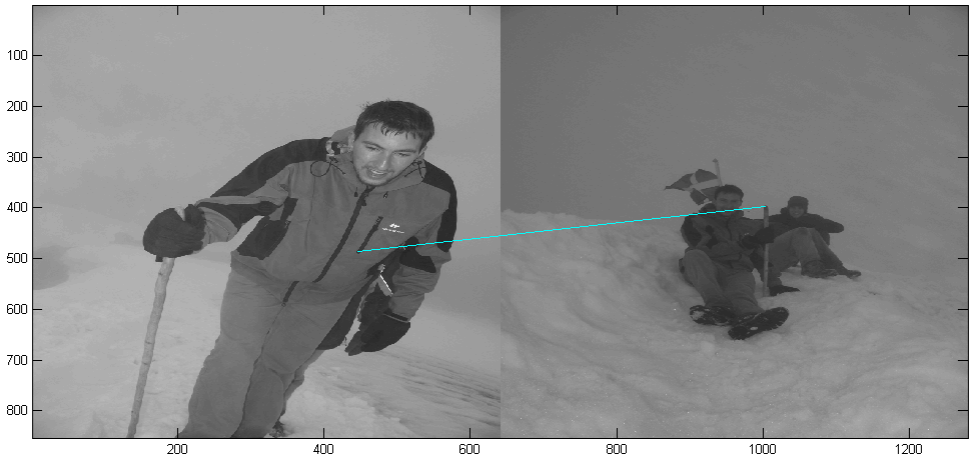

figure 75: matching result of a person in a more difficult scene, taking into account rotation, scaling, change of viewpoint and change in illumination

13. Applications

The SIFT algorithm is used in a variety of applications, including view matching for 3D reconstruction solving for structure from motion, motion tracking and segmentation, robot localization, image panorama assembly, epipolar calibration, …

The algorithm is also used in the AIBO entertainment robot, in which it is used for recognizing facial expressions of people and finding its charge station.

figure 76: image panorama assembly

figure 77: robot localization, motion tracking

figure 78: the AIBO entertainment robot

14. Future work

One interesting aspect that should be researched further is exploring other descriptors. Using a quadrature descriptor - a descriptor in with we only describe a quadrant of the region around the location of the keypoint instead of a circular region - should allow more points along the border of object to be matched correctly. Having more features the separation between true positives and false positives can be improved using stronger separation criterions. Especially in video data this is true.

Another approach could exist in also including illumination invariant color information in some descriptors, allowing again a stronger separation.

Using "a priory" knowledge about the position of the object to be matched, the reliability of the matching process can be increased, while lowering computational burden. One such technique could exist in using motion prediction algorithms (cfr. mpeg compression) and setting a bounding box in which matching feature points should be found.

Also worthwhile investigating is integration of newer techniques such as SURF (Speeded Up Robust Features) and GLOH (Gradient Location and Orientation Histogram).

Using a neural network approach we could train the descriptors (eg. by assigning weights to the components of the descriptor vectors) to result in a better matching process.

Since it is clear that that mammalian vision still performs a lot better than computer vision in recognizing object in a range of deformations, studying recent achievements in biological studies might also yield promising results.

15. Conclusion

In this report we reviewed the algorithm as described by David Lowe and validated the algorithm against some test sequences. Although we need to point attention to the fact that the validation was more a qualitative proof of concept rather than a fully detailed quantitative validation, we can conclude that up until some point the algorithm performs quite well, however there is a lot of room for improvement as was discussed in the previous section.

16. Acknowledgements

I would like to thank Roos, Tim, An and Thomas (photographer) for their collaboration in generating the test sequence for recognition of people.

17. References

Edelman, S., Intrator, N. and Poggio, T. 1997. Complex cells and object recognition. Unpublished manuscript: `http://kybele.psych.cornell.edu/_edelman/archive.html`

Harris, C. and Stephens, M. 1988. A combined corner and edge detector. In Fourth Alvey Vision Conference, Manchester, UK, pp. 147-151.

Hartley, R. and Zisserman, A. 2000. Multiple view geometry in computer vision, Cambridge University Press: Cambridge, UK.

Koenderink, J.J. 1984. The structure of images. Biological Cybernetics, 50:363-396.

Lindeberg, T. 1993. Detecting salient blob-like image structures and their scales with a scale-space primal sketch: a method for focus-of-attention. International Journal of Computer Vision, 11(3): 283-318.

Lowe, D.G. 2004. Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision, 60(2): 91-110

Luong, Q.T., and Faugeras, O.D. 1996. The fundamental matrix: Theory, algorithms, and stability analysis. International Journal of Computer Vision, 17(1):43-76.