MASTER

Motion robust techniques for camera-based monitoring of respiration

van den Helder, M.

Award date:
2011

Motion Robust Techniques for
Camera-Based Monitoring of Respiration

Maarten van den Helder

July 2011

Motion robust techniques for camera-based monitoring of respiration

Maarten van den Helder (0654815)
Technical University Eindhoven

***Abstract*—Unobtrusive vital signs monitoring is a hot topic in both lifestyle and the medical domain. The most promising methods at this moment are using video cameras. Existing methods for the monitoring of respiration are not robust for camera motion. There is a need for new methods that are more robust for this motion. This project investigates possible solutions to this problem by proposing new methods for global motion robust respiration monitoring. The main contribution of this paper is the use of a region of interest to protect the respiratory motion from being used in the compensation of the global motion. Two different quality metrics are proposed for analysing the accuracy of the different methods.**

I. Introduction

A. Context of the assignment

In hospitals it is important to measure and monitor the vital signs of patients. One of the goals is to get information about the health of the patients that you can not get from simply looking at the patient or asking questions. A specific vital sign to look for is the respiration of patients, which can be used as an early indicator of physiological deterioration. In literature it is suggested that cardiac failure is often preceded by significant changes in the respiration pattern [1]. However, currently the respiration has to be monitored by human inspection and to do this the person inspecting will have to watch and count the respiration of the patient for 30 seconds. In the current situation this very often does not happen [2].

Philips Research Eindhoven is one of the biggest research labs in the world. One of the many things being developed at the moment is camera-based vital signs monitoring. Not only for medical use in hospitals, but also for fitness equipment and in-car monitoring (e.g. drowsiness detection). A new algorithm for respiration monitoring has recently been developed in this project.

B. Problem definition

Although a first algorithm for respiration monitoring existed at the start of this project, this algorithm is sensitive to all kinds of motion. The camera has to be perfectly static, and the patient has to be stationary during the measurement. Basically, the only movement that is allowed in the view of the camera is the respiration itself. This is unpractical when the camera is located in a hand-held device.

This report will focus on the movement of the camera. Movement of the camera will produce *global motion* of the video sequence. Local motion is motion of an object inside the video, the compensating for such motion is outside the scope of this project. The difficulty of global motion compensating for respiration monitoring lies in the fact that not all the motion has to be considered when compensating the motion, since the motion of the chest not only has to be preserved, it is the element that has to be measured.

It should be noted that, during the course of this project, the existing respiration detection algorithm (*ProCor*) was also still in development. Limited information about the algorithm itself is available in this report. This is partly for reasons of confidentiality, the other reason is to keep the global motion robustness part of the algorithm separate from ProCor. This makes it easier to reuse the software for other (vital signs monitoring) projects and less sensitive to changes in other parts of the algorithm. ProCor takes a (static) video sequence that contains a respiration signal, and produces one value for each frame. This values can then be considered to be the differentiated respiration signal.

As a first benchmark, a test video was stabilised using the state-of-the art video stabilizer DeShaker [3]. The resulting video was processed by ProCor. The obtained respiration-signal was very noisy and no real respiratory information could be seen. This is probably caused by compensating all motion, including the respiration motion that needs to be measured.

C. Assignment

The Assignment was defined as follows: *To design, implement and test algorithms for global-motion-robust monitoring of respiration using a hand-held video camera.*

D. Outline

In the first part of this report (sections II and III) the different methods for global motion estimation and compensation are discussed. The report will focus on three techniques that were implemented in the code and makes mention of a couple of other techniques that are available.

The second part of the report describes the performed experiments and the results (section IV), this section will also explain the proposed quality metrics. Section V will briefly discuss the trade-offs between different methods. Section VI will conclude the report and propose future work.

II. Methods for motion estimation

A. Introduction

All applications of global motion estimation can generally be divided in two main groups: compression and image enhancement. The area of compression tries to find global motion in a video sequence as to improve the compression rate of a stored video. The area of image enhancement on the other hand tries to find and eliminate unintended global motion to enhance the video quality, e.g. eliminate jittering motion made by unsteady camera positions. The project described in this report has the goal to find and correct for the global motion in the video, to be able to retract an unrelated local motion signal (*the respiration signal*) from the video.

Motion estimation algorithms can be separated in two groups: global motion estimators and local motion estimators.

Global motion estimators attempt to find global motion in the video by looking at the frames as a whole, and not differentiating between blocks or objects. The output of a global motion estimator can be described in a global motion model like the three parameter model[4]:

$$\vec{D}(\vec{X}, n) = \begin{pmatrix} p_1(n) + p_3(n)x \\ p_2(n) + p_3(n)y \end{pmatrix}$$

where $\vec{D}(\vec{X}, n)$ is the displacement vector for a given pixel $\vec{X}$ in a given frame n, The parameter p_1 describes the motion in the x-direction, p_2 describes the motion in the y-direction, and p_3 describes the zooming of the camera. Note that in this report, the zooming of the camera is assumed to be negligible, since the person holding the camera is trying to hold it still. This implies that a simpler, two parameter model can be used. The two parameters p_1 and p_2 are combined in the *Global Motion Vector* ($\vec{G}$).

$$\vec{G}(n) = \begin{pmatrix} G_x(n) \\ G_y(n) \end{pmatrix}, \text{ and } \vec{D}(\vec{X}, n) = \begin{pmatrix} G_x(n) + x \\ G_y(n) + y \end{pmatrix}$$

Local motion estimators give multiple motion vectors per frame. Every motion vector only has the information about one pixel or pixelblock. However, it is possible to derive global motion information from the local vectors and thus using Local motion estimators to compute a global motion vector.

During our experiments, three different implementations for generating a global motion vector are used. This section explains the three methods. The compensation of the motion is discussed in chapter III.

B. Projection Based Motion Estimation

The most important difference between the projection based estimation and the other methods described in this report, is the fact that this method relies on global motion where the other two are actually computing local motion vectors and are extended with a computation to determine the dominant motion from the local motion vectors.

The computationally inexpensive algorithm divides the frame into a number of subframes (typically four) and makes vertical and horizontal *projections* of this sub-frame. The vertical projection $\vec{Pv}$ is a vector with the same height as the subframe, which values are defined as follows:

$$\forall i \in 0..\text{ (height\_subframe - 1) } : Pv_i = \sum_{j=0}^{width\_subframe-1} p_{i,j},$$

where $p_{x,y}$ is the Y-value of pixel x, y.

The definition of the horizontal projection is analogue to the vertical projection, only in the horizontal direction.

The algorithm does this for every frame in the sequence and compares the projections. For every subframe an optimal shift is calculated with respect to the previous frame. The optimal shift is defined as the shift of the 1D signal with the smallest SAD (Summed Absolute Differences) compared to the projection of the last frame. For a certain projection $\vec{Pv(n)}$ in framenumber n, and a certain shift s, and a maximum shift *maxshift* the SAD is defined as follows:

$$SAD(\vec{Pv}(n), s) = \sum_{k=\text{maxshift}}^{sizeOf(\vec{Pv})-\text{maxshift}} |Pv(n-1)_k - Pv(n)_{k+s}|$$

For both directions the median over the subframes is calculated and this way a global motion vector is generated. The median for a set of N values is the value with rank $\frac{N+1}{2}$ when N is odd. When N is even, the median is defined as the mean of the values with rank $\frac{N}{2}$ and rank $\frac{N}{2} + 1$.

The median operator will make sure that a motion value that only occurs in 1 subframe, that is probably local motion of an object, does not influence the global motion vector.

C. 3DRS

The Three Dimensional Recursive Search block-matching method (3DRS) is the second method that is implemented. This method is normally used for local-motion estimation, but it can also be used for global motion, by deriving one global motion vector from the many local motion vectors given by 3DRS. The method used is algorithm as described in [5], only without the extension to allow for sub-pixel motion vectors.

The basis of 3DRS is the normal block-matching algorithm. The frame is divided in blocks that are typically 8 by 8 pixels. Then for each block the best match in the image is searched with a minimisation technique such as the Summed Absolute Differences. The difference of 3DRS with respect to other block-matching methods is the 3-D approach to generate candidate vectors. Two assumptions have to be met for this algorithms to work efficient:

- Objects are larger than blocks
- Objects have inertia.

Object are defined as area's with the same actual motion vector.

The first assumption is necessary because only one motion vector is given per block. The second assumption is used in the method 3DRS uses for generating candidate vectors

The basic idea of 3DRS is to make up the candidate vectors by looking at the calculated motion vectors of the neighbouring pixel-blocks. Of course, if obtaining the vector for every pixelblock needs the vector from the neighbouring blocks, a causality problem occurs.

This problem is solved by looking at motion vectors in the previous frame. This is based the second assumption: *Objects have inertia*. Note that this is the reason why the algorithm is named three-dimensional. The other problem this idea is this:

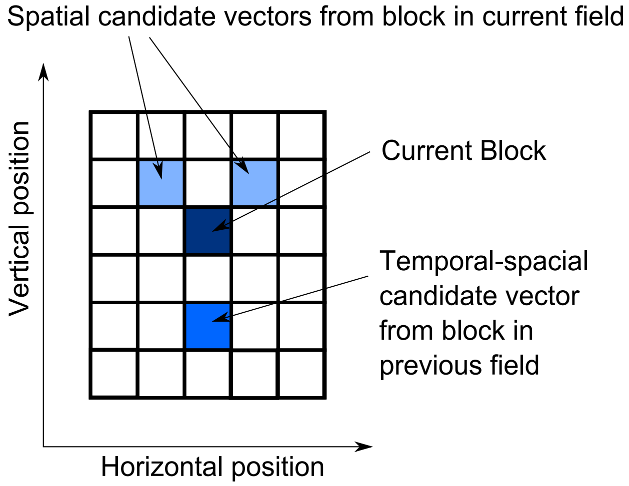

Fig. 1. Position of Candidate-vector blocks

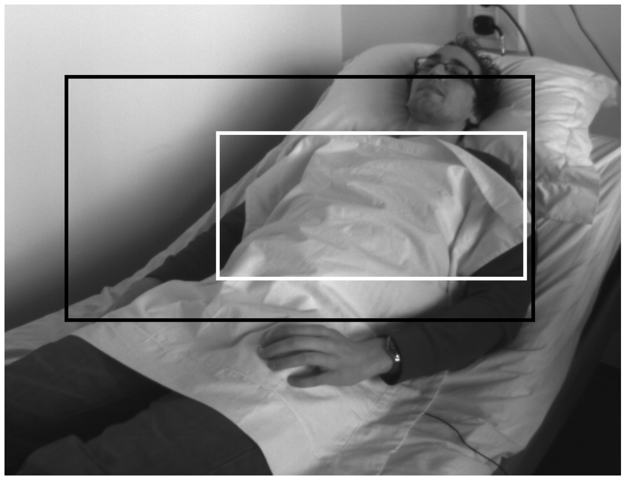

Fig. 2. Example of the position of both ROIs. shown in white is the RespROI, the NoMeasureROI is shown in black.

if all motion vectors are based on the value of other motion vectors, no new values can occur. Especially if the vectors are all initially 0, what is typically the case, everything will stay at zero. This is solved with the *update* vector, or noise vector. This is a (small) vector that is randomly chosen from a number of possible vectors. Each spatial candidate that is tried is always updated with such a update-vector.

The essence of the 3DRS algorithm is that once a block has a good vector, after a small number of frames all the blocks of the object in question will be checked against that vector and have obtained the right value. The algorithm gets even cheaper when a subset of the neighbours is used. The typical implementation of the 3DRS uses the "neighbouring" block as they are shown in figure 1.

D. *Optical Flow (Pyramidal Lukas Kanade,* PLK*)*

The optical flow method differs from the other methods since it is not designed to give a full vector field for the whole image. Rather than finding motion vectors for every part of the image, this method focusses on those part of the image that it expects good results from. To make that possible the image has to be searched for good features to track. This is done by looking for lines and corners in the image.

These features will be tracked with the Lucas Kanade Feature tracker. Each feature is a point in the image. For each feature-point in an image, the same pixel is searched for in the next image, within a specified search window. A pyramidal version of the Lucas Kanade algorithm was used as described in [6]. The implementation from *opencv*[7] is used with a pyramidal depth of 3. To find the initial features to track in the image frame, the method *cvGoodFeaturesToTrack* from opencv is used.

The basic of the Lucas Kanade Feature tracker is to for each feature-point $\vec{u}$, find the displacement vector $\vec{d}$ and the transformation matrix A that minimises the error function ϵ defined as follows:

$$\epsilon(\vec{d}, A) = \sum_{x=-\omega}^{\omega} \sum_{y=-\omega}^{\omega} \Big((I_{n-1}(\vec{x} + \vec{u}) - I_n(A\vec{x} + \vec{d} + \vec{u}) \Big)^2$$

Where $\vec{x} = (x, y)^T$. and ω sets the size of the integration window to $(2\omega + 1)^2$. The intuitive trade-off that has to be made concerning the size of the integration window is that a small window will give a high local accuracy, and a large window will be more robust against larger motion. The pyramidal algorithm solves this by using a multi resolution method. Several downsampled pictures are made (using an anti-alias filter) and the displacement vector is first found at the smallest image, which gives an initial guess for finding the vector in the the larger image, and so on.

III. Compensation of the estimated motion

The motion vectors obtained by one of above methods will be used to reposition the video frame and thus stabilise the video.

A. *Generating a single global motion vector*

The first task is to get the global motion vector from all the individual local motion vectors. With the Projection Based method this is not the case, since only one vector is obtained. But for the 3DRS- and the PLK methods a large amount of motion vectors are generated. It was decided to do this by taking the median of the x and y values of the all the vectors, and thus creating a global motion vector. The median is used because it is robust against small moving objects in the image, where a mean vector would be effected by this. An implemented way to improve on this is to take the alpha-trimmed mean in stead of the median. The alpha-trimmed mean for a set of N values is defined as the mean of the values with rank r for $\alpha < r < (N - \alpha)$ where r is the rank in a sorted list. In our implementation α is chosen in such a way that only five values are used. This method is only implemented for the PLK-method, since the nature of 3DRS causes the five values that are averaged to very likely be the same.

B. *Ignoring the respiration motion*

The main problem of motion robust respiration monitoring is compensating for motion (of the camera), where motion (of the respiration) has to be preserved. To achieve this, the option was built in the software to specify two different regions of interest: the *Respiration Region of Interest* (RespROI), and the *Not Measure Region of Interest* (NoMeasureROI). The RespROI is the area of the video frames that get sent to the respiration detection algorithm. It has to be chosen so that all motion that occurs inside this region is likely to be respiratory motion. This is typically the chest of the subject. The NoMeasureROI is used in generating the global motion vector. This region has to be chosen in such a way that all regions of the image where respiratory motion can occur are inside this rectangle. Typical areas to include are the chest, stomach, shoulders, and upper arms of the subject. An example of how the two regions are chosen is shown in figure 2. In the future it may be possible to have automatic selection of the two ROIs. For all different methods it holds that information from within the NoMeasureROI will never affect the value of the global motion vector. In practise, for 3DRS and PLK this means that all local motion vectors that are inside of the NoMeasureROI are not taken into account when calculating the median of the vectors. See figure 3 for a block diagram of the algorithm. In the projection based motion detection an implementation is made that defines eight subframes that together contain every part of the image but the NoMeasureROI.

C. Towards a true static video sequence

Most existing image stabilising algorithms only improve the *smoothness* of the camera motion. This means that camera motion will be present in the final video, only it will be more slower camera movement compared to the initial sequence. The respiration detection algorithm expects a video from a true static camera. The term *static video* is used to describe a video sequence that appears to be made with a stationary camera. Two different methods can be used to obtain a global motion vector that achieves static video. The first method that is considered is to actually calculate the motion from the first image in the sequence. For instance: in 3DRS, instead of letting the 3DRS-algorithm compare frame n and frame $n-1$, a copy of frame 1 is kept in memory as the *reference frame* and in each frame the 3DRS algorithm compares frame n with the reference frame. In the projection based algorithm, the new projections will be compared to the projections of the reference frame, in stead of the projections of the last frame. If the motion vector gets larger than a certain threshold value for long enough, the reference frame is refreshed with the current frame.

The second way of getting a static video uses a cumulative motion vector. Every time the global motion vector for two consecutive frames is computed, the x and y values are added to a cumulative motion vector, which is then used for the motion compensation. This vector is also reset if it gets larger than the threshold value for long enough.

In the case of the Lucas-Kanade algorithm, a refresh will also occur when the number of features outside of the NoMeasureROI is less then a defined value. And as a third option, a refresh can also be forced by the user.

D. Repositioning of the video-frame

The most straight forward way to do the motion compensation is translation of image locations. Each pixel in the compensated frame I_c is calculated according to the following equations:

$$I_c(\vec{X}) = \begin{cases} I_o(\vec{X} - \vec{G}_i) & \text{if } \vec{X} - \vec{G} \text{ is in the image frame} \\ \text{black} & \text{otherwise} \end{cases}$$

Where I_o is the original image-frame, and $\vec{G}_i$ is the rounded-to-integer version of the calculated global motion vector. Computationally, this method is relatively cheap, it is essentially a shifted frame copy. The most important problem, however, is the jittering effect on the video. This is caused by the sub-pixel motion of the image, that is corrected for on a integer-motion-based manner. Unfortunately, this effect is also very visible in the output of the respiration detection algorithm, *ProCor*.

An other, more computational expensive way of dealing with sub-pixel motion vectors, is to allow for sub-pixel motion image shifts. To do this every new pixel value can be calculated using *bilinear pixel fetch*. The pixels get interpreted as brightness values of a point in the middle of the corresponding pixel. The value of pixel $I_o(\vec{X})$ is then defined as the weighted average of the four pixels nearest to point $(\vec{X} + \vec{G})$ where the weight of each value is linear to the distance to the corresponding point. This calculation is usually done by first interpolating in one direction and then in the other, since this is computationally more efficient.

This function is only implemented for the PLK-method. This follows from the fact that the used implementation of 3DRS only gives pixel-accurate motion estimation, and it has no use to implement sub-pixel frame shifting with a pixel-accurate vector.

E. Sub pixel motion compensation in the respiration signal

An alternative to the bilinear pixel fetch, for overcoming the accuracy problem of the motion compensation only having pixel-accuracy, is to use the sub-pixel part of the motion vector directly on the output of ProCor. To do this, we should first have a basic understanding of the ProCor algorithm.

1) ProCor: The respiration monitoring algorithm is a very simple algorithm. For each video frame it makes a vertical projection as defined in section II-B *(Projection based method)*. Then a cross correlation function is calculated comparing the projection of each frame n to the projection of frame $n - lag$, where *lag* is the number of lag frames to avoid a noisy signal. The location of the peak of this correlation function is the output of ProCor, this can be considered to be the differentiated respiration signal. Hence, the respiration signal R is calculated as follows:

$$R(n) = R(n-1) + PC(n)$$

where $PC(n)$ is the output of ProCor in frame n. It is known from testing that the value coming out of ProCor is not only affected by the pixel shift, but also by the composition of the image. Due to the lag frames, a one-time one-pixel shift on the input image affects the next few consecutive values coming out of ProCor.

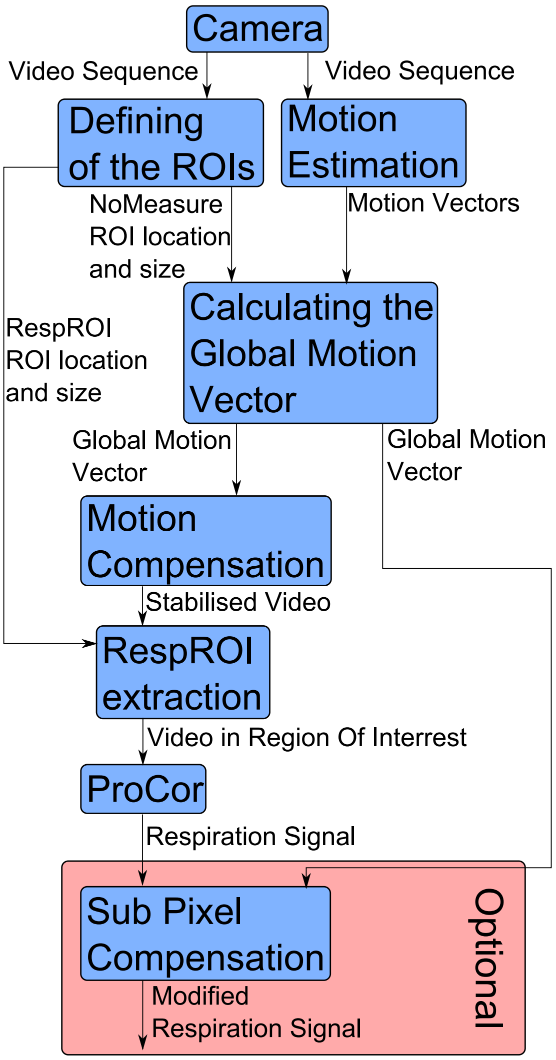

Fig. 3. The schematic for the algorithm.

For a static video sequence, the respiration movement has a fixed effect on the output of the respiration detection. It is assumed that for small camera motion (sub pixel motion) this effect on the ProCor values is the same for both static and moving cameras. So that with video sequences with small camera motion the value from ProCor will be the sum of the value from the respiration motion and the value from the motion of the video sequence. It is also assumed that the effect of small motions is linear in the output of ProCor: e.g. a shift of half the pixel width will add half the amount compared to a shift of one pixel width.

2) Compensating for motion on the respiration signal: Using this assumption, it can be reasoned that the desired respiration signal can be obtained by subtracting the effect of the motion of the video. The motion of the video is known in the case of a sub-pixel-accuracy global motion vector. Namely, it is the part of the sub-pixel-accuracy global motion vector that is not corrected for. When the shift is known the influence on the ProCor output can be calculated using the second assumption. But only if it is known how ProCor reacts to a single pixel shift. Since this value is dependent of the image composition there is no value that can be hard coded in an algorithm. Therefore the current implementation will calculate the *ProCor-Factor* (PKF), which is the value that is added to the output of ProCor with one pixel shift in the y direction. With this information, it can calculate the influence of the pixel accuracy motion correction and subtract it from the ProCor output, which after that will only consist of the respiratory data. The block diagram for this case is shown in figure 3. The new respiration signal R_{sm} will be calculated according to the following equation:

$$R_{sm}(n) = R_{sm}(n-1) + PC(n) - PKF \cdot \sum_{k=0}^{4} \left(\vec{G}(n-k)_y - \vec{G_i}(n-k)_y \right)$$

where $\vec{G}(n)_y$ is the y value of the global motion vector, and $\vec{G_i}(n)_y$ is the y value of the rounded-to-integer global motion vector

IV. Experiments

A. Introduction

To test the performance of the different implementations, a large amount of test videos is recorded.

Two kind of experiments are done. The first experiments that are considered in this report use the video-sequence from the static camera with added, synthetic, motion. This is done to compare the different implementations. Since the basic, non-motion-robust, respiration detection can be run on the static sequence, there is a *ground truth* that we can compare the results to.

The second experiments are done on video sequences from actual hand-held cameras. The results from this tests will show whether the good implementations in the synthetic sequences are also good with real hand-held camera material.

B. Setup

For each recording at least two cameras are used. One camera is mounted on a tripod, the other camera is hand-held during the whole recording. This way the videos are recorded at the same moment in time. Also a respiration measuring belt is worn by the test subject, to provide a reference signal.

A subject follows the following breathing-schedule: For all four positions complete the sequence in figure 4.

Where the four positions are as follows:

- Lying on back with the bed in sitting position
- Lying on the left side (facing the camera)
- Lying on the right side (not facing the camera)
- Lying face down
- Sitting with legs outside of the bed

C. Quality Metric

To interpret the results of the experiments, there has to be a quality metric to be able to give the different algorithms a rating in how good they are. The goal of the project is *To design, implement and test algorithms for motion robust monitoring of respiration using a hand-held video camera.*

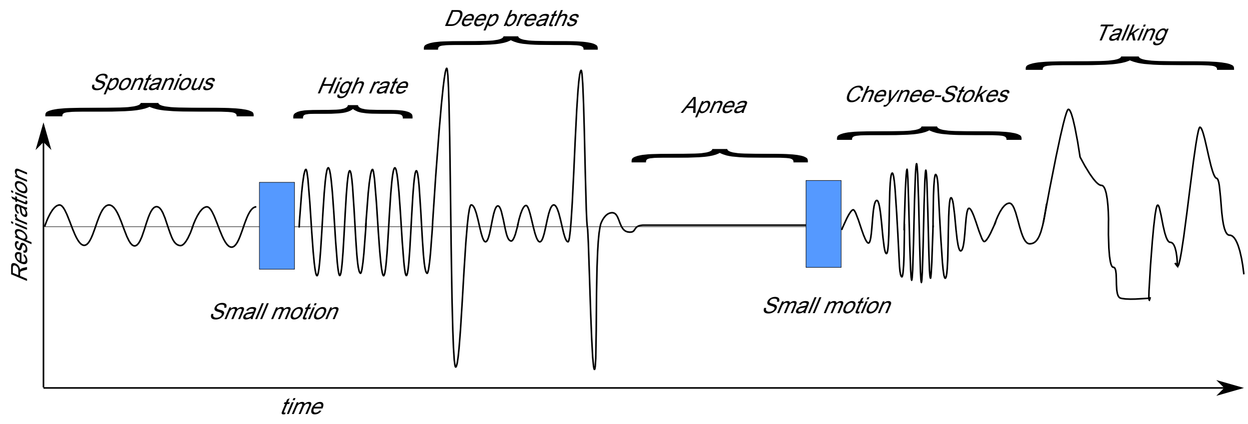

Fig. 4. The breathing pattern for the test sequences

From this it can be said that a algorithm is good if it can *monitor respiration*. Two different metrics are proposed for measuring the quality of the signal. Both make use of a *reference signal*, that is considered to be ideal. The intuitive meaning is that the signal will get a higher score, if it is more similar to the reference signal.

Two choices are possible for the comparison signal, namely the reference signal from the respiration monitor belt, or the respiration signal from the static camera for the same video sequence. The choice is made for the signal from the static camera. This way both signals are constructed by the ProCor algorithm, and the signals have similar amplitudes and characteristics. Two methods are proposed to construct such a metric.

The first method depends on a trained peak detector to calculate the respiration rate that can be extracted from the signal during the whole sequence. This is also done for the comparison signal. The metric in this case is the percentage of the video in which the two signals give the same respiration rate (within a allowed deviation). Note that this will only give a score for the corresponding video in combination with the corresponding algorithm. To give an overall score to the algorithm an average will be made of the scores of all available video sequences. In specific use-cases, it is also possible to make this a weighted average, were sequences that are considered more important will have a bigger weight in the final score.

The second method makes use of the correlation coefficient between the signal that is being analysed and the comparison signal. To remove large steps and offset both signals are first processed by a band-pass filter. A 4^{th}-order bandpass filter is used witk pass band $\frac{0.1}{fps/2} < \omega < \frac{1}{fps/2}$ After this the correlation coefficient is computed between the two filtered sequences. Note that this method can give undesirable results when there is motion other then respiration in the sequence, e.g. the subject changes position or a second person walks through the region of interest. The relatively large motion has a very large influence on the correlation function. To prevent this from having a effect on the score, these parts of the sequences must be ignored by the correlation function. This is done by calling the correlation function multiple times, once for every "pure respiration" subsequence of the sequence. It is evident that this method, like the first one, will give an independent score for each tested (sub)sequence. The same technique can be used here: compute a (weighted) average of all scores per algorithm.

Both techniques have advantages and disadvantages. The respiration rate method depends completely on the quality of a peak-detector, if the peak detector gives wrong peaks, the rate will be wrong and the algorithm will have a low score. The other fact is that it only looks at the respiration rate. This can be a advantage, since in a lot of possible use-case scenarios this is exactly what is desired from the algorithm. The disadvantage is that the respiration rate is not the only possible information in the signal, and some uses of respiration monitoring may need the other information. As described, the correlation method is sensitive to large motion.

D. *Distance to the camera*

A small experiment is also performed regarding the optimal distance from the camera. This experiment has the same setup as the other experiments with the exception that there are three in stead of two cameras, of which two are hand-held. The two hand-held cameras are held in different distances to from the chest of the subject. The two hand-held cameras have the same specifications and lenses. It must be noted that while it would be ideal for the experiment to have the same viewing angle on the subject, this cannot be done without one camera being in the way of the other.

E. *Results*

During the first exploratory tests and subjective evaluation of the signals it became clear very fast that both the "projection-based" motion detection, and the "Sub pixel motion compensation in the respiration signal (section III-E)" gave very poor —even unusable— results.

With the projection based motion estimation, both the reference frame-method and the cumulative-vector method are tested. In the reference-frame method, the poor result was due to the simplicity of the system; when large peaks in the projection fall outside the screen in the current frame, the optimal shift will likely be one that places the peak on the nearest other peak. It might be possible to get better results using a windowing function on the projection, but this is not tested.

The "Sub pixel motion compensation in the respiration signal" also gives very poor results. The reason is a false assumption. In the development of this method, it was assumed that *"with video sequences with small camera motion, the value from ProCor will be the sum of 1) the value from the respiration motion and 2) the value from the motion of the video sequence"*. This assumption is based on a model of ProCor that is not complex enough. The method could possibly be extended by taking into account different kinds of motion (not only in the y-direction) and different behaviour of ProCor with amount of detail in the video composition.

Due to the poor results, both these methods are discontinued and are not present in the further test results.

The results of the experiments (the respiration signals) are investigated to see which parts contain real respiration motion and which parts are the result of other motion in the frame; only the parts that have respiration motion are considered. This way only the respiration detection is tested and not the reaction to local motion within the ROI. All sub-sequences were then graded with the correlation-method. The peak-detector method is not used in this experiments, since there is no robust peak-detector available at the time of writing this report. All unexpected results are also compared manually.

1) Synthetic motion: Below are the results for the test with synthetic motion. The test is done as follows:

- The static video sequence was divided into five different subsequences, one for each position of the subject.
- These sequences are run trough the basic ProCor algorithm to generate a reference signal.
- A file with realistic motion vectors was created, this was done by pointing the camera to a contrast-rich scene at approximately the same distance as the subject would be from the camera in the hand-held scenario (approximately 1 meter). The recorded sequences were then processed with the PLK motion estimator, and the global motion vector was derived as described in section III.
- The static sequences were shifted according to the obtained motion vectors. This was done with a bi-cubical resample method.
- The sequences with synthetic motion are cropped to remove the shifting black borders.
- And finally the videos with synthetic motion are processed by the different proposed methods for stabilising.

After the division in good subsequences, there are ten sequences that all are evaluated with the correlation coefficient metric. These numbers are then averaged for each stabilising method and that gives the score for the method.

The expectation of this experiment is that the methods with the PLK method will give the best result, of which the methods with bilinear pixel fetch will be the top. PLK is followed by 3DRS with the reference frame, and the least result of this test will probable be for 3DRS with the cumulative motion vector. This is expected because the 3DRS method is very sensitive for motion vector propagation in large areas with little detail. PLK does not suffer from that since it only follows point that have good detail. The cumulative vector will suffer from cumulative errors, where the method with the reference frame will compute a "fresh" motion vector every frame.

Furthermore it is expected that the methods that use the NoMeasureROI will perform better than the ones without, since they will not compensate for motion that is actually respiration motion that needs to be measured by ProCor. The least significant factor will probably be the alpha-trimmed mean, the normal median performs pretty well on itself. Taking the average of the five vectors will have only a marginal effect on the output-signal, also because the five vectors will probably be very close to each other in terms of value. The result of this experiment can be seen in figure 5. The meaning

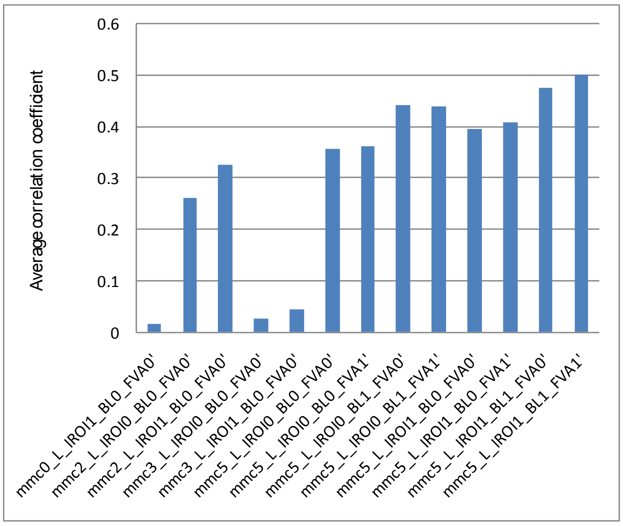

Fig. 5. The results of the synthetic motion experiment

of the names of the different methods can be found in table I, note that the different size of the region of interest is not used in this experiment. The bar chart in figure 5 shows that the test results almost exactly as expected. A example of certain signals and how they compare to the reference-signal can be seen in figure 6. The only point where the expectation did not hold is the case *mmc5_L_IROI0_BL1_FVAx* were the method with the median motion vector gets a slightly better result than the one with the alpha-trimmed mean. What also is notable is the large difference between the 3DRS with reference frame and the 3DRS with the cumulative motion vector. This effect is accounted for by the cumulative error. It is true that the PLK-method also makes use of a cumulative motion vector, and clearly does not suffer from this problem. However, the implemented 3DRS method only generates pixel-accurate motion vector. This means that each error will be larger, and the accumulated error will naturally become much larger as well. The 3DRS implementation with the reference frame will not suffer as much from this error, since it will compute independent new motion vectors for each frame.

2) hand-held camera: The experiments with the synthetic motion have shown that the methods work, and how good they work in relation to each other in the very basic case. To see how they work in a more realistic use-case the experiments must be done with videos from real hand-held camera's.

The experiments are done in the same way as the synthetic-

mmc: Method of Motion Compensation	mmc0: No Motion compensating mmc2: 3DRS with reference frame mmc3: 3DRS with cummulative motion vector mmc5: Pyramidal Lucas Kanade (PLK)
L/S: size of ROI	L: large ROI, same for every sequence S: Small ROI, unique for each sequence
IROI: state of the NoMeasure (inverse) ROI	IROI0: not active IROI1: active
BL: method of image shift	BL0: No Bilinear, Pixel accurate image shift BL1: Bilinear pixel fetch image shift
FVA: Five Vector average	FVA0: off, only the median vector is used FVA1: on, the average of the five most median vectors is used

TABLE I
NAMING CONVENTION FOR THE DIFFERENT METHODS IN THE TEST RESULTS

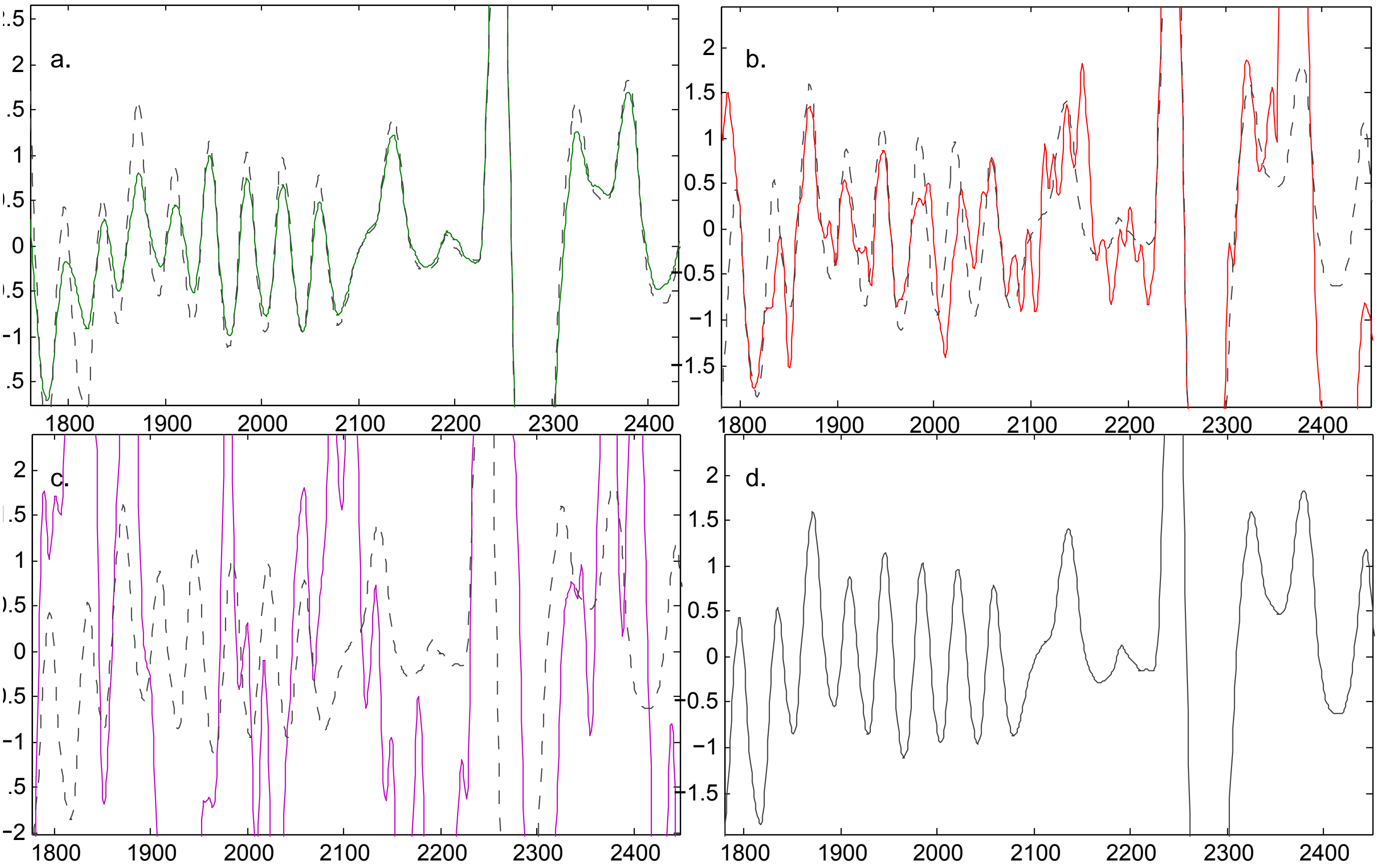

Fig. 6. The (band pass filtered) results of the synthetic motion experiment: The dashed line is the reference signal, the solid lines are:
a) The best scoring method (mmc5_L_IROI1_BL1_FVA1),
b) The best 3DRS-based method (mmc2_L_IROI1_BL0_FVA0),
c) Non-usable method (mmc3_L_IROI1_BL0_FVA0),
d) The reference signal.

motion experiments. The difference is that the references are compared to real hand-held-camera video sequences. Some specific sequences are disregarded in the test because the respiration was too hard to detect in this sequences. Mostly this was because the person holding the camera was moving it to much (intentionally shaking the hand holding the camera). Some sequences suffer from a lot of rotating movement. Since we do not estimate or compensate for rotation these sequences do not give a usable result with any of the proposed methods.

The expected result of this experiment is that the overall score will be a lot lower compared to the synthetic motion. However, the relative position of the different methods is suspected to be the same. The size of the ROI is not differed in the synthetic-motion experiment. It is expected that the smaller, hand-picked ROIs will give a significant better performance than the general, larger ROI. Note that with the smaller ROI, both the RespROI and the NoMeasureROI are smaller and manually defined per sequence. The results can be seen in figure 8, and an example of some of the signals can be seen in figure 7. The naming convention can be found in table I. Two things are very remarkable. Firstly, the poor results of the PLK-method when a large ROI is used. Three of them are even scoring lower then the reference sequence where no motion compensating is used. This can be explained by the fact that when a large ROI is used, most objects seen in the part of the image that is not in the ROI have a different distance to the

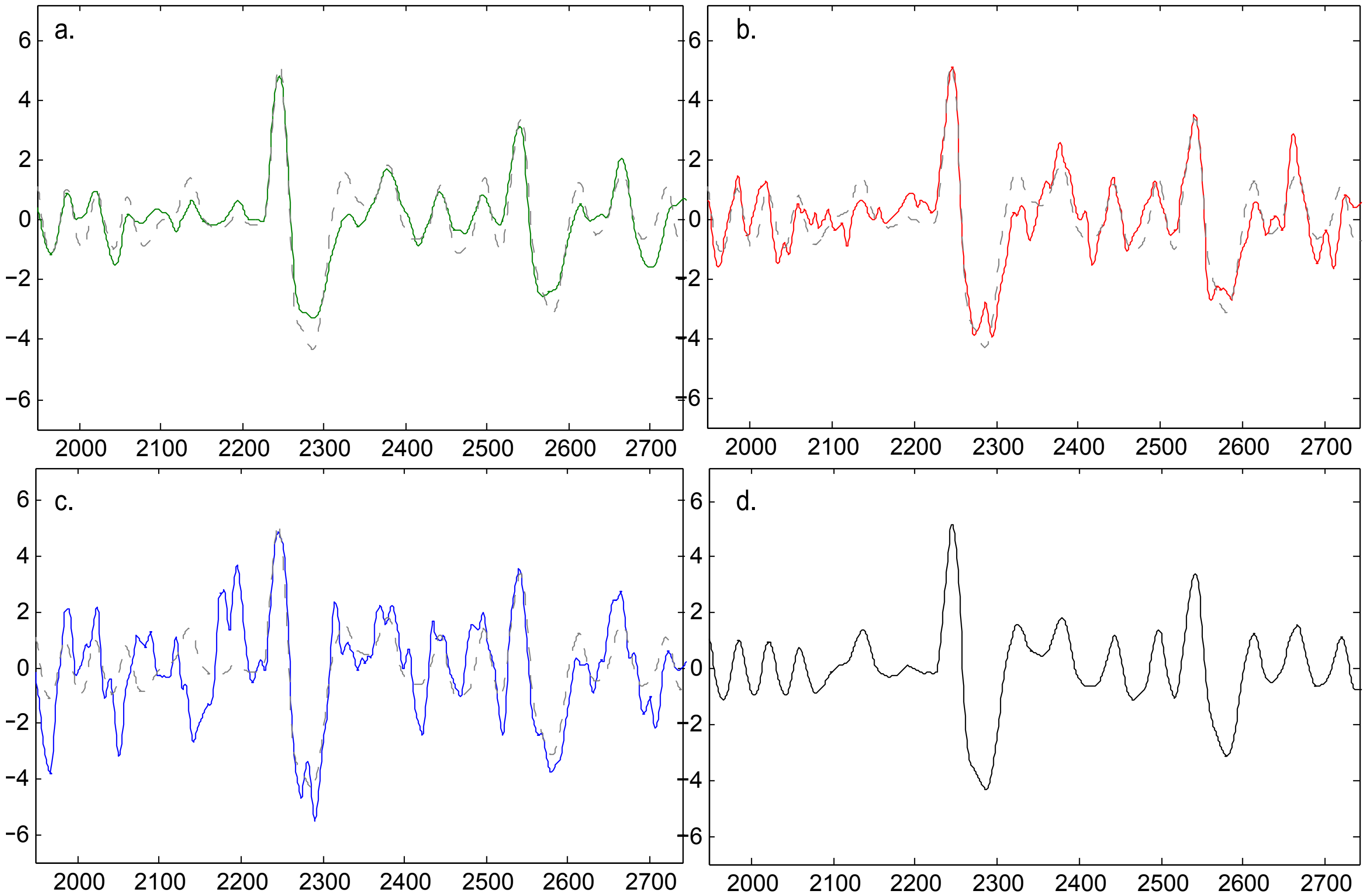

Fig. 7. The (band-pass filtered) results of the hand-held camera experiment: The dashed line is the reference signal, the solid lines are:
a) The best PLK-result (mmc5_S_IROI1_BL1_FVA1), b) The second-best PLK result (mmc5_S_IROI1_BL0_FVA1) ,
c) The result with the best correlation score (mmc2_S_IROI1_BL0_FVA0), d) The reference signal.

camera as the test-subject. (see figure 2 for a example).

The other notable fact is that one of the 3DRS-based methods (*mmc2_S_IROI1_BL0_FVA0*) has the highest score of all. However, when looking at the graph of the signals, it is clear that the signal from the PLK-method is still preferable over the 3DRS method (see figure 7 a and c). The strange score is probably due to the imperfection of the quality metric. The exact explanation remains an open question.

When these things are taken into account, the rest of the methods are graded similar to the synthetic motion experiments.

3) Different distances from the camera: One small test is done considering the optimal distance from the camera. The intuition is that when the camera is closer to the subject, the motion of the camera will have less influence on the motion of the video-sequence. And less motion in the video-sequence would mean a better respiration signal. However, the results for the camera that was very close-by (approximately 30 cm) are very unexpected. For some sequences the results follow the expected order that was seen in the results of the hand-held camera, with better scores. But for a equally large number of sequences the results are very different.

Particularly the PLK-method scores a lot of negative values on the correlation metric. Values occur as low as -0.74. Since the used quality metric is based on the correlation coefficient

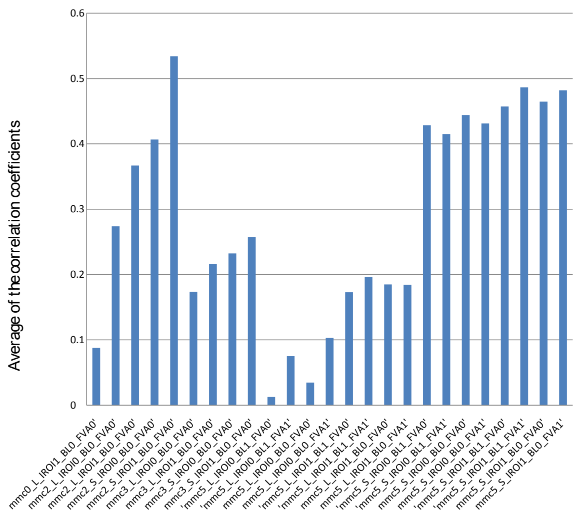

Fig. 8. The results of the hand-held-camera experiment

for comparing a signal against its reference signal, this means that a high negative correlation is found. When the signal is inspected manually (see figure 9) it is easy to see that some parts of the video give a inverted respiration signal. Further investigation of the signals and the global motion vector show that when the camera is very close to the subject, the chance is very large that most tracked features contain respiration motion. This way the obtained global motion vector will contain respiratory motion and in some cases it will overcompensate so much that the RespROI will contain motion opposite to the respiration motion. The problem is that when the NoMeasureResp is chosen to contain all areas where respiratory motion can occur there is not enough detail left in the image to estimate global motion from.

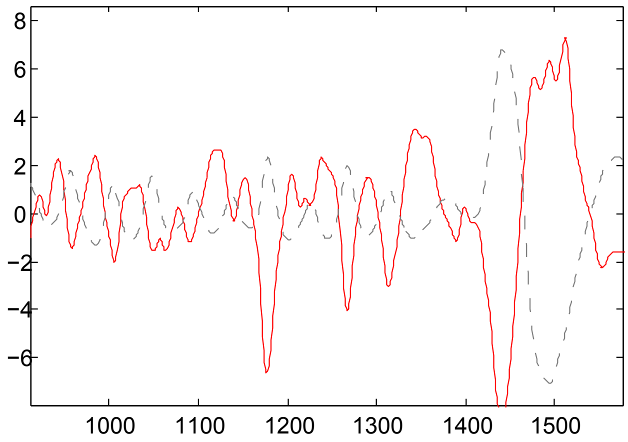

Fig. 9. Overcompensating when camera is too close to test subject: dotted line: reference signal, solid line: PLK, small ROI

V. Trade-offs

In the results section it is shown that there is a significant difference between the results of the different methods. This section will discus how computationally expensive these methods are. As a method for comparing the three best methods, the average computation time per frame is recorded. The compared methods are: 3DRS with reference frame, PLK with pixel accurate image shift, and PLK with bilinear resampling image-shift. The results are as follows:

Method	Average computation time
3DRS reference frame	0.1198 s
PLK no bilinear fetch	0.0706 s
PLK bilinear fetch	0.1001 s

These results are measured on a PC with an Intel core 2 duo processor running Windows XP at 1.86 GHz

Note that the implementation of 3DRS is a non-optimized implementation and the OpenCV method for PLK is a optimized one. Also, the bilinear pixel fetch used in the PLK-method can be optimized, for getting more pixels with the same displacement vector at the same time. The way it is implemented now, it is independently called for each pixel. On top of that, the PLK can be optimized to only track features outside of the NoMeasureROI. An other optimisation, that is useful for all methods, but especially for the bilinear pixel fetch, is to only reposition the image when the RespROI is extracted from the image to be sent to ProCor. This way only the pixels that are needed are fetched.

Nevertheless, the differences in speed are explained quite intuitive. 3DRS is a very fast method if a full vector-field must be calculated. However, a whole vector field is not needed in the case of this problem. The number of computations that is needed per motion vector will be much smaller for 3DRS, but 3DRS has to compute 5640 motion vectors per frame for a typical camera resolution and blocksize. The pyramidal Lucas Kanade feature tracker only computes a fraction of this number. The typical value used in the experiment is 100 features per frame.

Because of all the above, 3DRS is not considered to be the right algorithm for this problem.

The real trade-off decision will be the extra computational power (needed for the bilinear pixel fetch) against a more accurate respiration signal.

VI. Conclusion

Robustness of respiration monitoring to global motion has been investigated. New methods have been designed and tested. The designed methods are based on the principle that the global motion of the image frame must be determined without affecting analysis of the respiratory motion in the video. To achieve this, a region of interest is manually defined in the scene (*NoMeasureROI*). This region should include all the areas of the video where respiratory motion can occur. The algorithm is designed in such a way that any information from within the NoMeasureROI can not influence the global motion vector, which is used for global motion compensating.

Estimation of the global motion vector can be done in several ways, producing three different proposed methods. One method uses a projection based global motion estimator, another method is based on the 3DRS motion estimation algorithm, and the last one is based on a pyramidal implementation of the Lucas and Kanade feature tracker.

Two quality metrics are designed for objective evaluation of respiration signals obtained with different methods. The first quality metric focuses on the respiration rate and is dependent on a peak detector for noisy respiration signals. The second quality metric is based on the correlation analysis of the respiration signal and a reference signal.

The method that provided the best results based on the correlation-analysis quality metric, is the method using the pyramidal implementation of the Lucas and Kanade feature tracker. The tracked features are classified based on spatial position to exclude the motion vectors that contain respiration motion. This is done by disregarding all features that are placed inside the NoMeasureROI. The values of the global motion vector are then defined as the alpha-trimmed mean of all remaining vector values for both the x and the y value.

During the experiments it has been observed that the quality of the particular peak detection algorithm makes an impact on performance of the whole respiration monitoring algorithms as well as on the reliability of the objective metric.

VII. Future work

Future work in this project will include optimization of the algorithms for computational cost and implementation in a embedded system in a hand-held camera device. Furthermore, it is suggested to extend the global motion model to include rotation, since the current algorithm is not robust to that kind of motion. An even more motion robust manner would be to implement Motion/Depth Estimation as is done in [8] this way the algorithm could compensate motion for the right camera distance.

REFERENCES

[1] M. Folke, L. Cernerud, M. Ekström, and B. Hök. Critical review of non-invasive respiratory monitoring in medical care. *Medical and Biological Engineering and Computing*, 41(4):377–383, 2003.

[2] F. McGain, M.A. Cretikos, D. Jones, S. Van Dyk, M.D. Buist, H. Opdam, V. Pellegrino, M.S. Robertson, and R. Bellomo. Documentation of clinical review and vital signs after major surgery. *Med Aust*, 189:380–383, 2008.

[3] Gunnar Thalin. Deshaker - video stabilizer for virtualdub, January 2011.

[4] M. Goetze. Generation of motion vector fields for motion compensated interpolation of hdtv signals. *Signal Processing of HDTV*, pages 383–392, 1988.

[5] G. de Haan and P.W.A.C. Biezen. Sub-pixel motion estimation with 3-d recursive search block-matching. *Signal Processing: Image Communication*, 6(3):229–239, 1994.

[6] J.Y. Bouguet. Pyramidal implementation of the affine lucas kanade feature tracker - description of the algorithm. Technical report, Technical report). Intel Corporation, 2001.

[7] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[8] G. Zhang, J. Jia, W. Hua, and H. Bao. Robust bilayer segmentation and motion/depth estimation with a handheld camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.