EVALUATION OF MOTION ESTIMATION TECHNIQUES

FOR IMAGES DEGRADED BY ATMOSPHERIC TURBULENCE

Michael Vlasov - graduation project

Ben-Gurion University of the Negev, Beer-Sheva, Israel
Department of Electrical and Computer Engineering
Electro-Optical Engineering Unit

Abstract

Atmospheric turbulence is usually the main limiting factor for long-distance optical imaging. It distorts the wavefront of the incoming light, which results in image degradation, mostly by blur and lateral spatio-temporal distortion (image dancing). For many applications these phenomena are highly undesirable. Distortion, for example, would affect machine vision applications which require static background scene. Another example is super-resolution techniques based on frame stacking, which require registered video stream with sub-pixel relative displacements. Image processing techniques can be utilized to reduce these effects. Blur can be reduced by methods like deconvolution. The spatio-temporal movement distortion is usually addressed to using motion estimation and compensation techniques. The estimation produces a motion vector field, while the compensation "restores" the distorted image. As a result, a "boiling" turbulent video can be converted into a "frozen" stream, where turbulent motion in every frame is compensated relative to a specific time point.

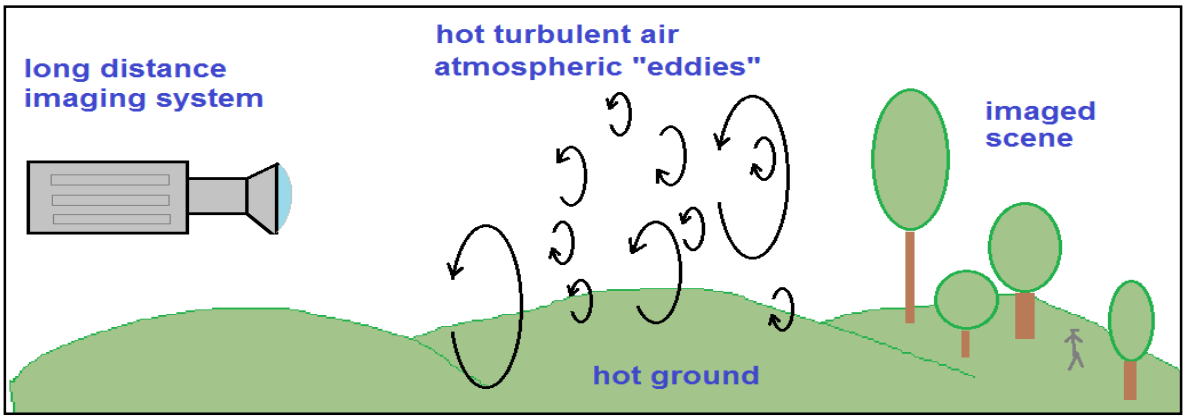
There are a variety of techniques for motion estimation, most common of which are block matching, and optical flow estimation (as the basic Lucas-Kanade method). This work focuses on analyzing and evaluating application of these techniques for compensating local distortions caused by atmospheric turbulence. A variety of statistical criteria, real life turbulent videos, and turbulence models were used. In order to perform a numerical evaluation - simplified models were developed for simulating atmospheric turbulence, which correspond to real life videos. These models feature variable distortion, blurring and noise. Several hundred motion fields and compensated images were computed, while specific focus was given to "fine tuning" estimation techniques by introducing and adjusting various parameters. PSNR and SSIM (structural similarity) methods were used for motion compensation evaluation. A visual inspection was performed in order to filter out techniques which produce isolated image artifacts. In addition, while most of the research in this field focuses on evaluating quality of restored images, we addressed the fidelity of the estimated motion field itself.

Contents

ABSTRACT 2
CONTENTS 3
TURBULENCE AND IT'S EFFECT ON IMAGING 4
SIMULATING ATMOSPHERIC TURBULENCE 5
SOLVING THE PROBLEM OF DISPLACEMENTS 10
MOTION ESTIMATION APPROACH 11
PERFORMANCE EVALUATION MODEL OF MOTION ESTIMATION METHODS 11
MOTION COMPENSATION 14
STATISTICAL COMPARISON 15
BLOCK MATCHING MOTION ESTIMATION 17
VARIANTS AND RESULTS OF BLOCK MATCHING MOTION ESTIMATION: 18
LUCAS KANADE MOTION ESTIMATION 31
VARIANTS AND RESULTS OF LUCAS KANADE MOTION ESTIMATION: 32
COMPARISON OF LUCAS KANADE AND BLOCK MATCHING MOTION ESTIMATION: 42
CONCLUSIONS 44
REFERENCES 45

Turbulence and it's effect on imaging

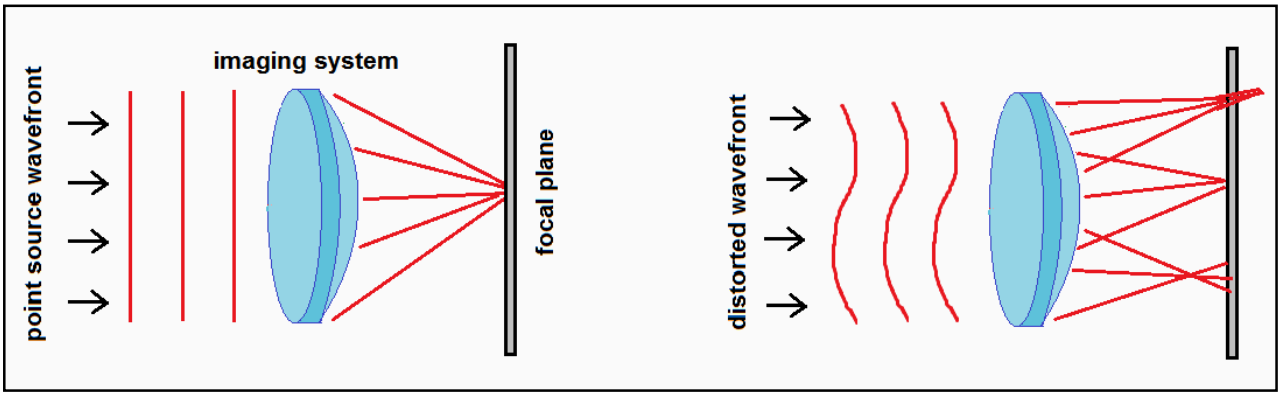
Atmospheric turbulence is a phenomenon caused by stochastic and chaotic movements of atmosphere, mostly due to convective flows and wind. As a result of turbulence - regions of atmosphere with different density, and therefore different refractive indexes are formed randomly (due to Gladstone-Dale law) [10]. These regions are also called "Atmospheric Eddies" and they come at various sizes [9].



Typical scene of imaging through turbulent medium

These eddies, or "atmospheric lenses" alter and deform the wavefront of light which passes trough them. As a result, while imaging through turbulent layers, the imaged scene is deformed, causing random blurring and image "dancing" (effectively it's causing temporal and spatial distortions).

For example let's take a look on a point-like source, which produces plane wave hitting the aperture of imaging system. Tilting the incoming wavefront (altering its phase) will shift the point source image over the focal plane. Spherizing of the wavefront will cause the focal point to come in or out of focus (causing blur). Distortions of the wavefront of higher order (as in the following image) will cause multiple occasions of these effects: splitting the point into several locations, and blurring it due to defocus.



Example of effect of turbulence on point-like source

Since these events are highly time dependent – during the integration time (exposure) these effects are accumulated, and additional blurring is introduced. Basically the PSF of a given optical system is no longer limited by diffraction and aberration parameters of optics, but also is subject to effects of turbulence.

Turbulence depends on many factors such as temperature, altitude and air pressure, landscape and obstacles, winds, humidity, etc. For example – turbulence is especially strong during the day, at low altitudes (due to convective currents caused by hot ground). In addition - effect of turbulence on imaged scene also depends on imaging parameters such as: distance, focal length and aperture, integration time, intervals between exposures, integration time, wavelength, etc.

These are examples of images taken from a long distance, on a hot day, which clearly show strong blurring and displacements of details. The left image demonstrates stronger turbulence effect, introduced by higher optical magnification:

Example images taken through turbulent air

Simulating atmospheric turbulence

The purpose of the work is to evaluate and compare different techniques of motion estimation, for images degraded by atmospheric turbulence. Visual evaluation of motion fields and restored turbulent images only give us quality comparison. Therefore in order to perform quantitive evaluation – the first step is to simulate atmospheric turbulence on a given image. The simulated turbulent images allow us to calculate a true motion vector field with known values, and to perform numerical comparison with estimated motion field, using statistical criteria.

Image degraded by atmospheric turbulence can be modeled using the following equation [1]

$$g(i, j, t) = D[x(i, j, t) * h(i, j, t), t] + \eta(i, j, t)$$

Where x is the original image, D is geometric distortion (warping), h is dispersive component (blurring), η is noise, t is time, i and j are image coordinates, and [*] is two dimensional spatial convolution.

Since this work focuses on techniques for estimating motion caused by atmospheric turbulence only, and doesn't address the problem of global motion (i.e. moving camera) or non-elastic motion (i.e. moving objects) - there was no need to introduce these types of motion into simulation model.

D represents stochastic distortion, caused by atmospheric "eddies" [9], [10] which are "pockets" in atmosphere with different refraction coefficient (caused by air density differences). These eddies flow in the atmosphere and come at different size scales. Since eddies are flowing in front of an imaging system, they are causing distortion of a wave front, which results in distortion of image in the focal plane.

In order to simulate this distortion (D) - we placed a grid of control points on an image, which defines the distortion component [6],[1]. Each control point is shifted, using rectangular random distribution [6] with a given maximum displacement parameter. The distortion value (D) for pixels in between the control points was interpolated using cubic spline interpolation, which achieved higher smoothness then bilinear interpolation used in [1]. Note that cubic spline interpolation method assumes smoothness of the destination function, which is indeed the case for turbulence induced distortion.

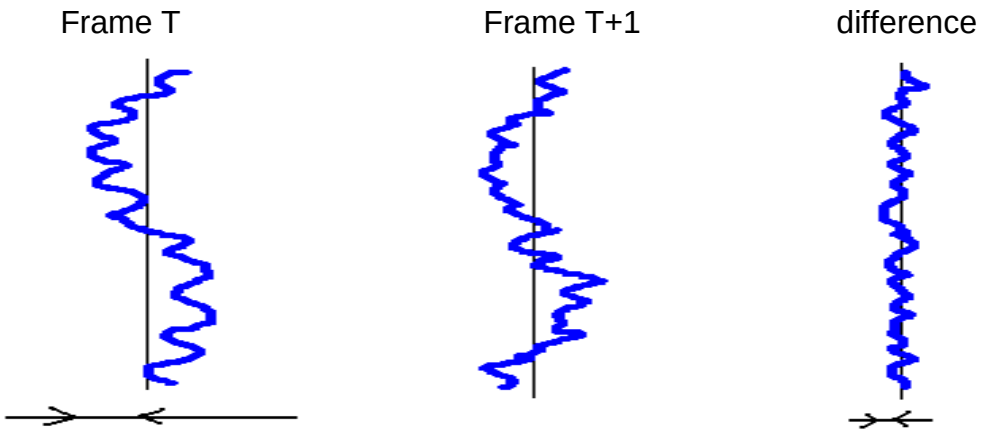
In addition to local turbulent distortion modeled by control point grid stochastic movement, we introduced a component at higher spatial frequencies. This was done in order to simulate larger atmospheric "eddies". The need for modeling this lower spatial frequency distortion became obvious after comparing the simulation with real turbulent images. An additional grid of control points with lower spatial density was placed. Similarly to the control grid we used to simulate local motion - rectangular random distribution was used. This low spatial frequency distortion was smoothened by Gaussian filter, in order to achieve continuous transitions between control points. The resulting low spatial frequency motion field element was added to high spatial frequency element, in order to produce final distortion motion field.

Several sets of parameters of simulated turbulence behavior were chosen. In order to base selection of the parameters on ground truth – We examined real life videos, with

different cases of turbulence. The parameters for each set were selected to make the simulated results match these real-life videos.

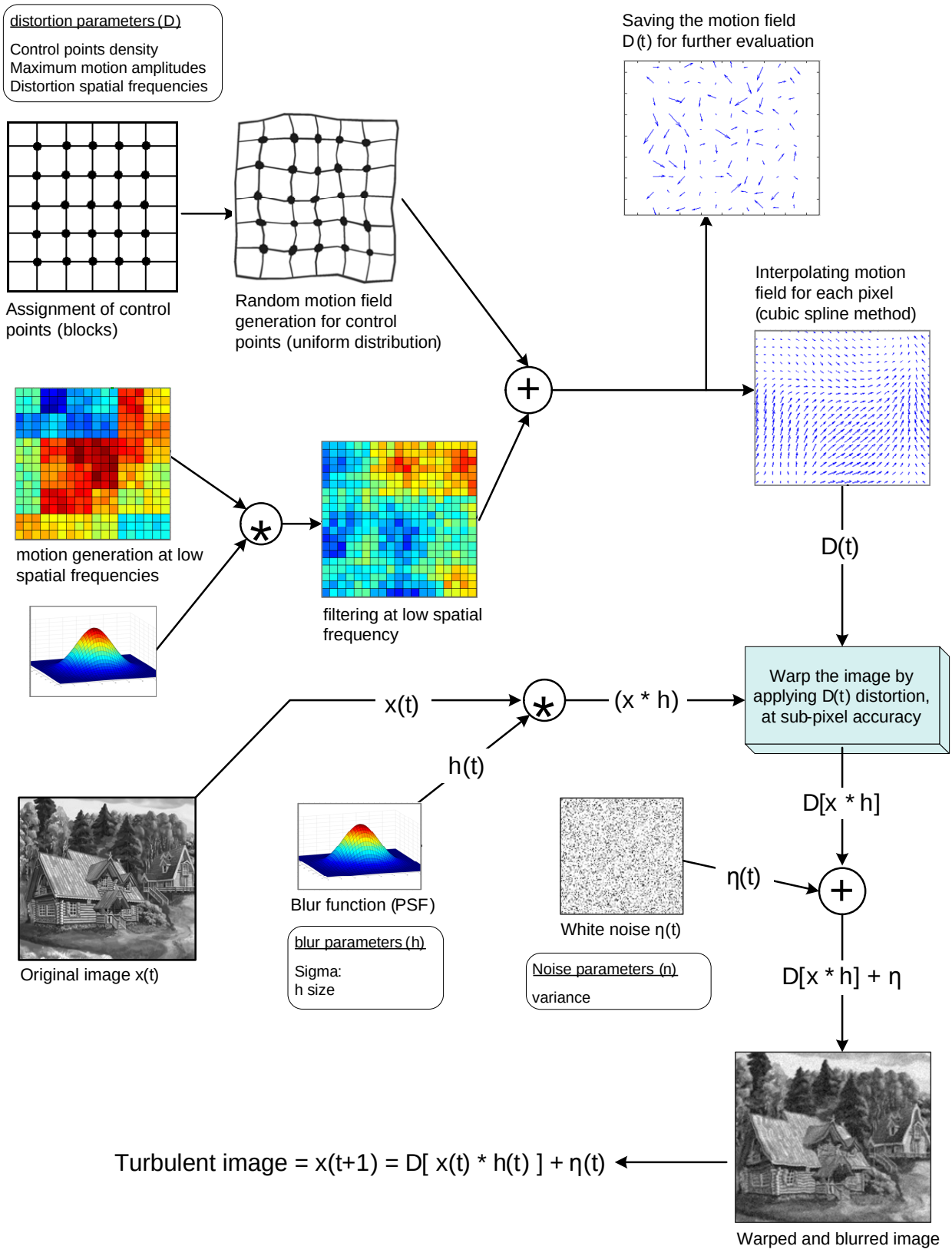
Higher spatial frequency component was chosen by examining consequent frames of a turbulent video. The lower spatial frequency components were chosen after examining frames with longer interval between them.

As it was mentioned earlier - atmospheric "eddies" come at different size scales. These eddies also flow in front of an object, due to wind, or convection [9], [10]. Since for larger eddies it takes longer to pass in front of the imaged scene - the distortion at lower spatial frequencies will also change slower (low spatial frequency components will also have low temporal frequency). This also means that when we take a video, degraded by atmospheric turbulence, two consequent frames will have high spatial frequency distortion difference, but will almost have no low spatial frequency distortion difference.



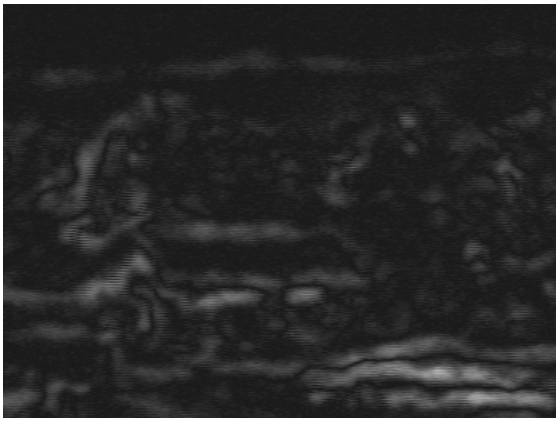
This is a reason for choosing frames with long interval between them, for allowing evaluation of image distortion magnitude at lower spatial frequencies.

The following workflow diagram summarizes the process of simulating atmospheric turbulence, which generates motion field D(t), and the resulting "turbulent" image x(t+1).

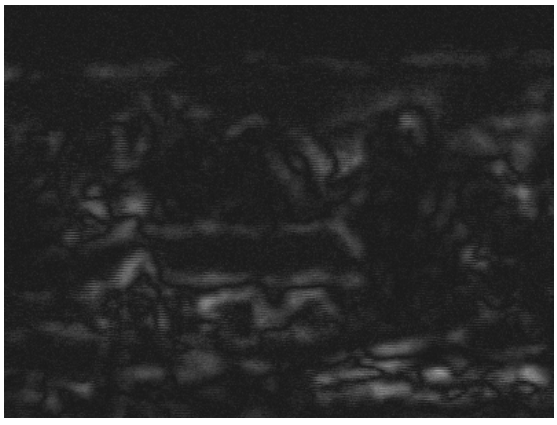


Atmospheric turbulence model – simulation workflow

Example of true turbulent frames from video "Houses2" (strong turbulence) :


Difference: 

Simulated video frames (with parameters which match real turbulence):


Difference: 

The following table shows sets of parameters which were input into the simulation model, in order to match several real life videos with different degree of turbulence:

Video	degree of turbulence	control points grid: small	control points grid: large	high freq. distortion amplitude	low freq. distortion amplitude	low freq distortion filtering σ	blur PSF h-size	blur PSF σ-sigma	white noise variance
Fields1	weak	10	80	0.8	1.2	2	2	1	0.0001
Fields2	medium	10	110	1.7	2.9	2	3	2	0.0001
Flir1	very weak	24	144	0.9	1	2	0	0	0.0002
Houses1	strong	10	140	1.9	4	2	3	2	0.00005
Houses2	very strong	18	144	2.8	6.5	1	4	3	0.0001

Input parameters into atmospheric turbulence simulation model

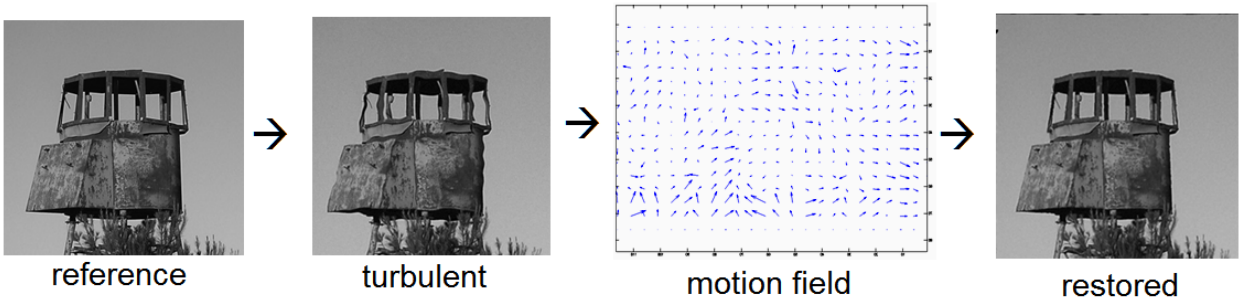
Solving the problem of displacements

There are two main effects of turbulence: Blurring and spatio-temporal displacements. Blurring causes loss of fine details since the spreading of PSF function causes the higher spatial frequencies to be attenuated. For long exposures for example (or a result of frame stacking), the integration of blur and displacement effects can be regarded as a Gaussian blur [6]. For short exposure video frames, however, these two phenomena need to be treated separately. Blurring is hard to formulize, since it causes by various factors such as defocus and higher order PSF distortions, integration of displaced points (with various levels of distortion). There are many methods of reconstruction the blurred image – for example blind deconvolution, or Kurtosis minimizations. We will not cover this subject here.

Spatio-temporal displacements cause loss of location and time information for details. This is especially harmful for motion detection algorithms in machine vision, and for compression difference based algorithms. This work focuses on estimating and restoring spatial displacements of image at software level, using common motion estimation techniques, and evaluating their performance for this purpose. Apart from the simulation model – we did not address temporal displacements of image. They are usually less harmful for most popular applications, and reconstructing them would require implementing three dimensional motion estimation algorithms (2D spatial and temporal).

Motion estimation approach

Using motion estimation is a practical way to compensate turbulence induced displacements, while working on a software level. Most popular methods in this field are either based on block matching, or on gradient methods (i.e. Lucas Kanade). Implementing motion estimation algorithms on a set of two images degraded by turbulence (reference frame and distorted frame) allow us to build a map of motion vectors, which is called “motion field”. Using this field we can restore the image, and “undo” the effects of turbulent induced displacements.



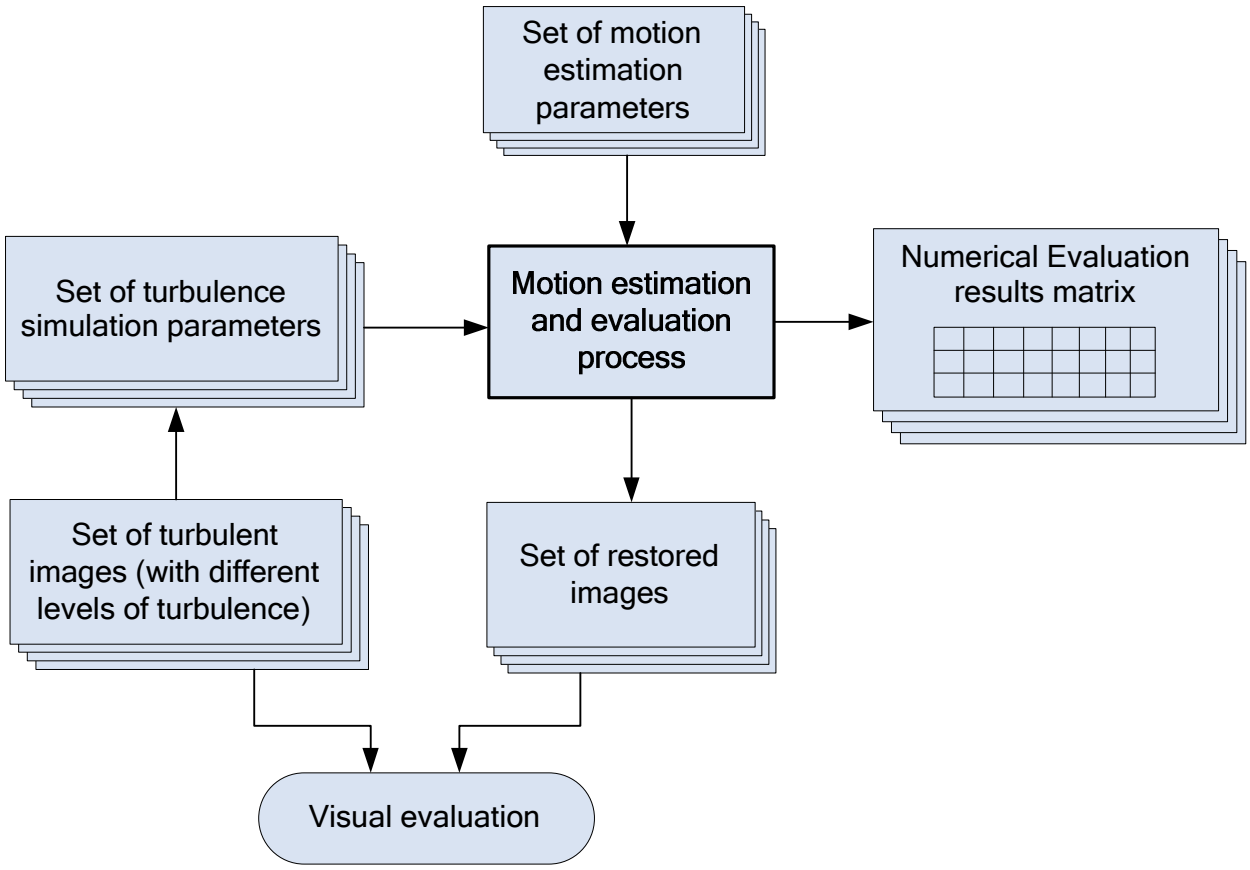
Motion estimation concept

Since in practice we can not obtain “original” or undistorted image – there is a need to produce a good reference frame. For example using time-averaged frame, taken over a specific period of time around the current time period of time: $t_0 \pm \Delta t$ .As a compromise, we can use one of the frames as a reference, at specific time intervals.

Displacements caused by atmospheric turbulence are smooth or “elastic”. We will not address here the problem of camera movement (which is usually described by various transformation models) and the problem of moving objects. These movements need to be dealt with separately.

Performance evaluation model of motion estimation methods

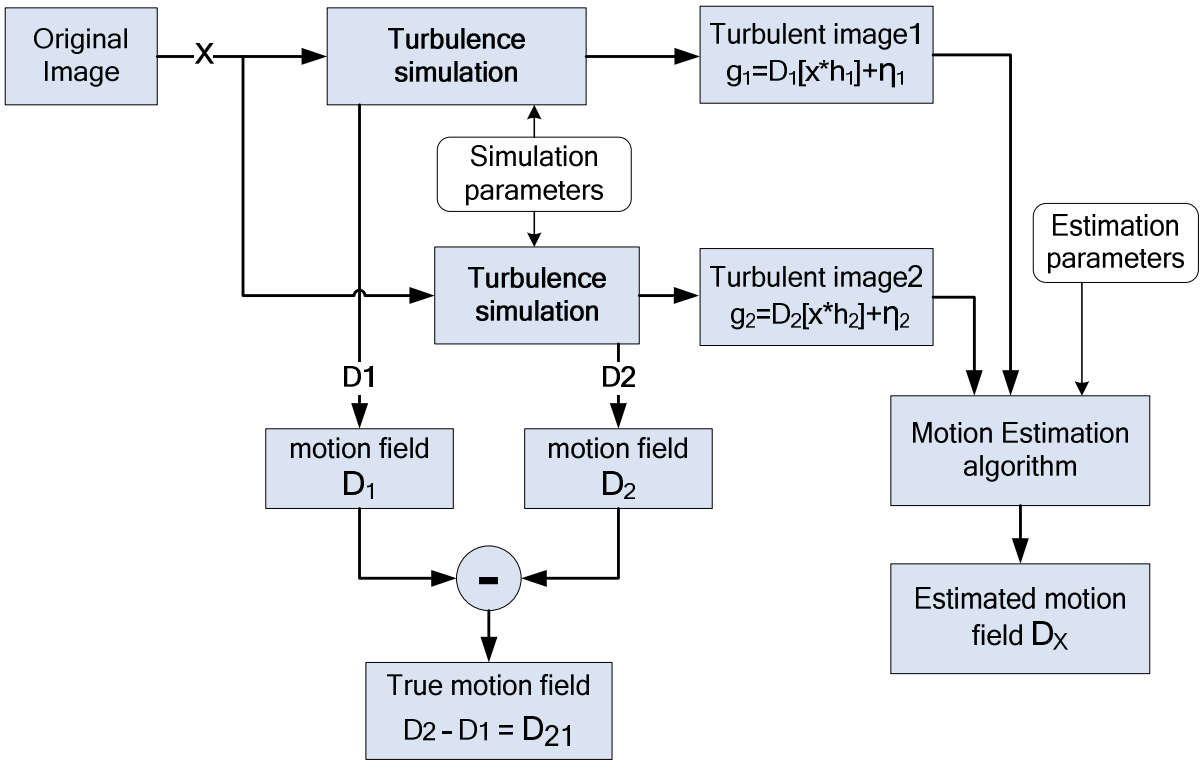
In order to evaluate performance of motion estimation methods for the purpose of compensating motion and displacements induced by atmospheric turbulence - an evaluation model was built, which included several entities: Set of real life videos with varying level of atmospheric turbulence, atmospheric turbulence simulation model with a set of parameters, motion estimation algorithms with a set of “tuning” parameters, motion compensation algorithms, and finally the estimation fidelity evaluation algorithm.



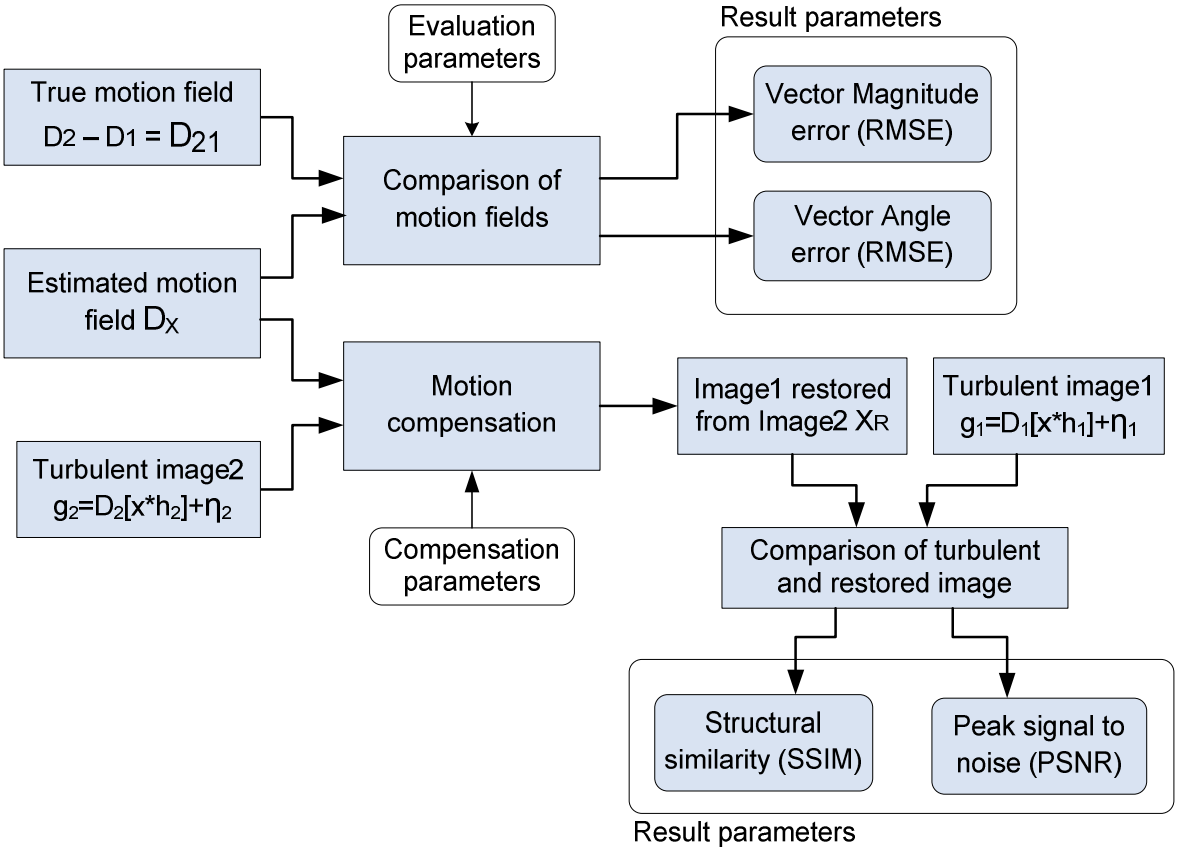
Evaluation model

One of the innovations was to introduce so called “tuning” parameters and variables into motion estimation algorithms, and evaluate their effect on performance. Another innovation is the method for evaluating performance of motion estimation techniques. It features statistical methods such as SSIM (structural similarity), and direct statistical comparison of true and estimated motion fields. These methods are used in addition to more popular PSNR analysis, which proved to be ineffective in our case.

The following workflow was developed to perform the comparison.



Workflow of motion estimating process

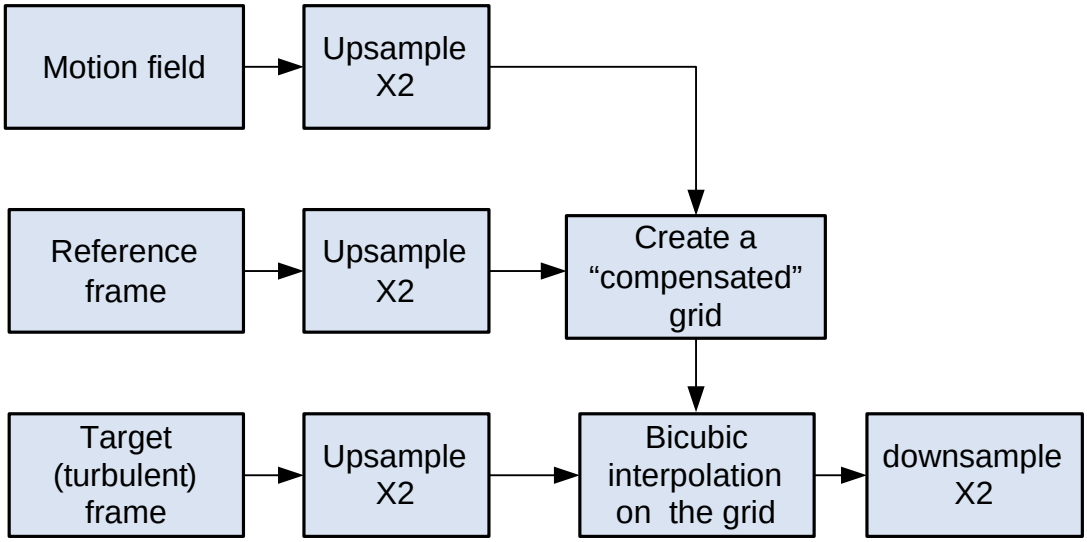


Workflow of evaluation process

Note that original image is fed twice into turbulence simulation process, in order to simulate two turbulent images at different time intervals (T1 and T2). Then turbulent image1 (at T1) is regarded as "reference".

Motion compensation

In order to compute the “restored” or a “compensated” image - we chose to interpolate the turbulent image on a reference image grid, using estimated motion field. This process is called “dewarping”. Bicubic interpolation method was used. This method proved to be highly superior over Bi-Linear interpolation. Bi-Cubic method was also tested, and produced almost similar results. Also in order to increase the accuracy – we performed interpolation on a sub-pixel level (half-pixel accuracy) by upsampling the images before the interpolation, and then downsampling them.



Motion compensation workflow

The “tuning parameter” we use here is upsampling factor, which determines ration of sub-pixel accuracy for motion compensation.

Simulations showed that using upsampling, before “dewarping” (which effectively increases the dewarping resolution) significantly increases the fidelity of reconstructed image, in terms of structural similarity criteria. The improvement of structural similarity of compensated image was in average twice as much, when upsampling was used during the de-warping process.

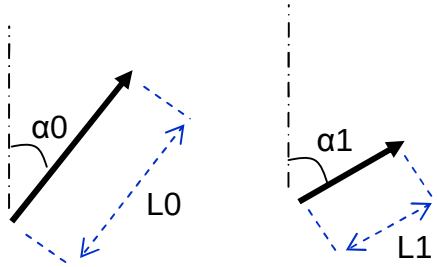
Statistical Comparison

In order to provide reliable comparison and evaluation results – we used several methods to compare the results of different motion estimation techniques:

Direct Comparison of motion fields:

Most works in the field of motion estimation evaluate the similarity between the compensated images, usually by computing PSNR. Since our main purpose was to evaluate motion estimated methods, and not the compensation techniques, we introduced direct comparison of motion fields, both the magnitude and the direction.

Each vector in the motion field has magnitude (L), and angle (α).



$$\Delta\alpha = \begin{cases} \alpha 1 - \alpha 0\ ,\ \Delta\alpha \le 180 \\ 360 - \alpha 1 - \alpha 0\ ,\ \Delta\alpha \le 180 \end{cases}$$

$$\Delta L = L1 - L0$$

Calculation of motion field magnitude error and angle error

In order to provide statistical information: Root mean squared errors between these two values were computed over the whole motion field.

$$RMSE_{magnitude} = \sqrt{\frac{1}{mn}\sum\sum[L_0 - L_1]^2} \qquad RMSE_{angle} = \sqrt{\frac{1}{mn}\sum\sum[\Delta\alpha]^2}$$

The result errors are given in pixels for the magnitude, and in degrees for direction.

PSNR:

Peak Signal to Noise Ration is the most commonly used technique for evaluation of processed image quality, and it's widely used in various processing and compression techniques. It involves computation of mean squared error for each pixel of original and compensated image, and normalizing it relative to maximum pixel value (255 in most cases, while 8 bits are used for each colour).

$$MSE = \frac{1}{m\,n}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2 \qquad PSNR = 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right)$$

As we will demonstrate later - this technique doesn't provide us with reliable results.

SSIM:

Structural Similarity Index is a much more reliable technique for comparing the similarity between images. While PSNR takes into account only values of a single pixel and sums it, SSIM technique takes into account surrounding pixel values by calculating statistical parameters over a certain window. This produces results which are much more similar to visual perception, and much more immune to noise.

SSIM index is calculated over a certain image block:

$$\mathrm{SSIM}(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

$$c_2 = (k_2 L)^2 \qquad k_1 = 0.01 \qquad L = 2^{\text{bits}} - 1$$

$$c_1 = (k_1 L)^2 \qquad k_2 = 0.03$$

Where μ_x, μ_y are averages, σ_x, σ_y are variances, σ_{xy} is covariance. X,Y are two images under test.

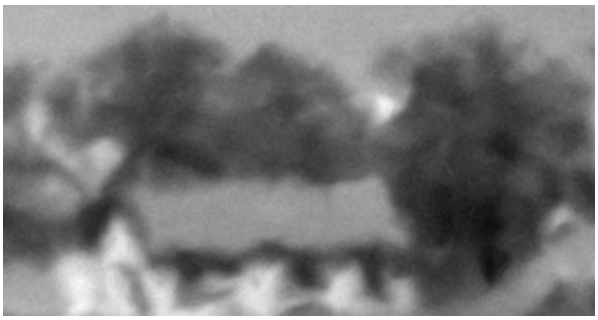
The results are in percents, while 100% represent images with perfect match, and 0% represents images with no match at all.

We computed two SSIM indices: one between original image and the distorted one. And another one between original image and one which was motion compensated. The difference between these two SSIM indices is called “SSIM improvement”, and it shows how much motion compensation improved the image, which was degraded by turbulence.

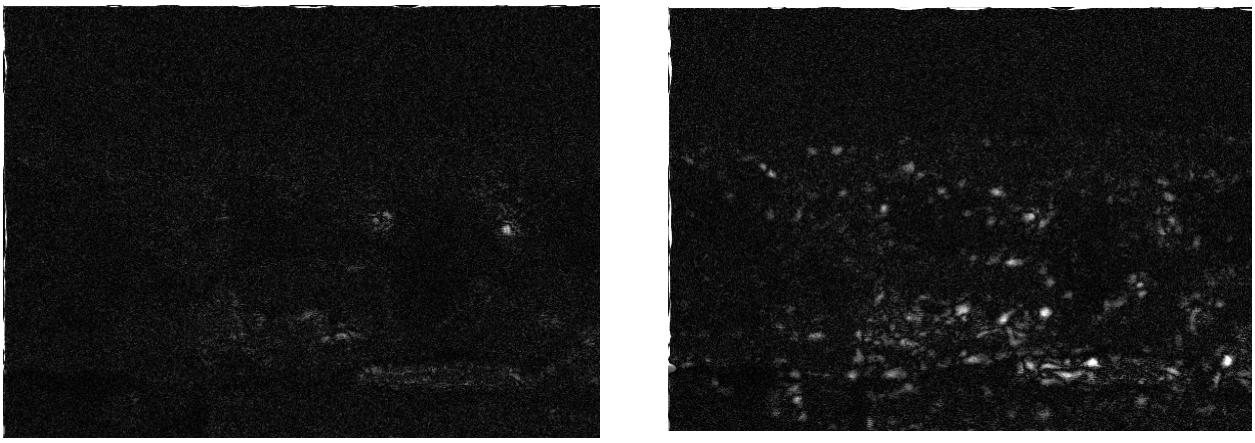
PSNR vs. SSIM:

The following is an example which compares SSIM and PSNR for a certain case.

Motion compensation method 1 (LK).



Motion compensation method 2 (BM)



Difference images for the above two cases

We can clearly see that method 1 (LK) produces much cleaner, and more accurate result then method 2 (BM). This is also confirmed by analysis and comparison of estimated and true motion fields (1.88 pixel average error compared to 2.67 pixels) Visual inspection of videos, which were compensated using both methods – also showed clear superiority of method 1 (LK).

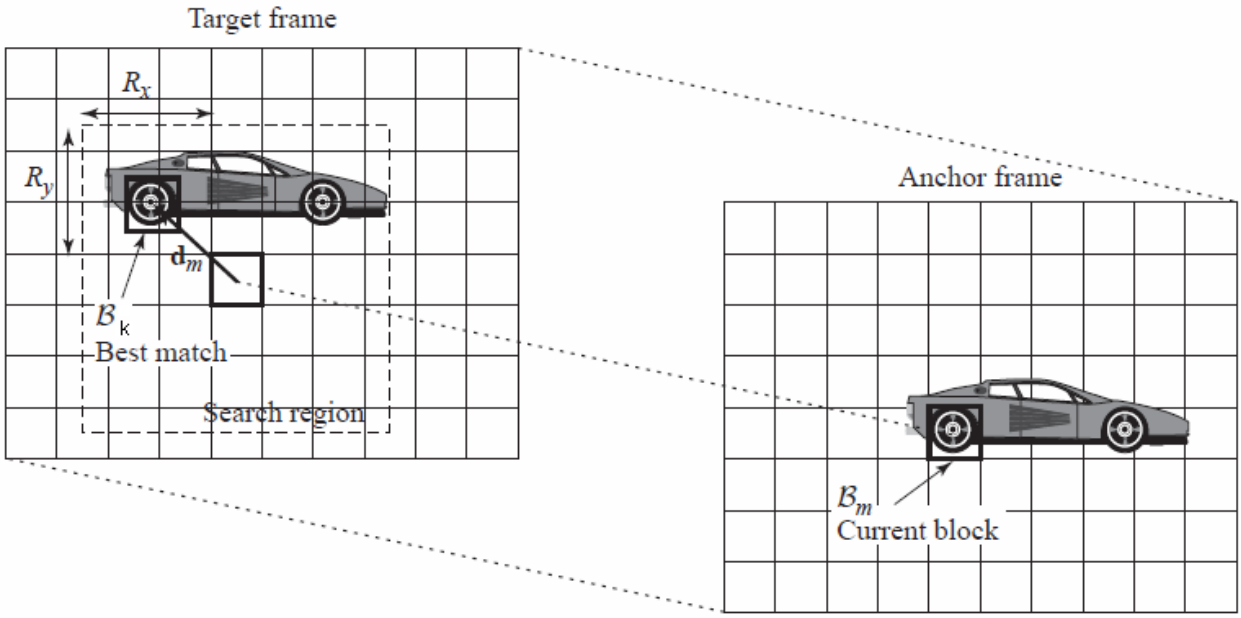
The calculated SSIM index improvement for the first method is 9.38%, and for the second (the less accurate) method is 7.05%. However the PSNR improvement is the opposite – 0.67dB in first method versus 3.19dB in second one. This clearly does not represent the true difference in accuracies of compensated images.

Since similar results were produced over a wide range of simulations – it is clear that PSNR technique can not be used for our purpose of comparing methods of motion estimation, in case of videos degraded by atmospheric turbulence.

Block matching motion estimation

Block matching is a popular method for motion estimation, which is widely used in video compression, particularly due to its low computational cost which enables it's use even in real time applications.

The idea is to divide image into blocks, while for each block we assume constant motion. First we pick a block in a reference frame (anchor frame), and then in the following frame (target frame) we search, within a certain "search region", for a block which matches the target block using some kind of comparison criteria. The difference between coordinates of block inside the reference frame, and the matching block inside the target frame, equals to motion vector of this region.



Block matching motion estimation

Variants and results of block matching motion estimation:

Block matching technique has many matching criteria and other “tuning” parameters. We used several sets of them, and compared the results.

SAD: Sum of absolute differences.

SAD is a matching method. We calculate the sum of absolute values of differences between each pixel for two blocks under comparison. Then we find for which of the blocks this sum is minimal.

$$SAD = \sum_{j=y-R_y}^{j=y+R_y} \sum_{i=x-R_x}^{i=x+R_x} | B_m(i,j) - B_k(i,j) | \quad \rightarrow \quad \text{MIN}$$

Where: (i,j) are pixel indexes inside the block, (x,y) are pixel indexes of the whole image, (Rx, Ry) are motion search radiuses, Bk and Bk are anchor (or reference) and target blocks respectfully.

XCOR: Cross-Correlation.

This matching method computes two-dimensional normalized cross-correlation map between search region of target frame, and a single block of anchor (reference) frame. The maximum of this map specifies the point of highest match between blocks.

SAD - half pixel.

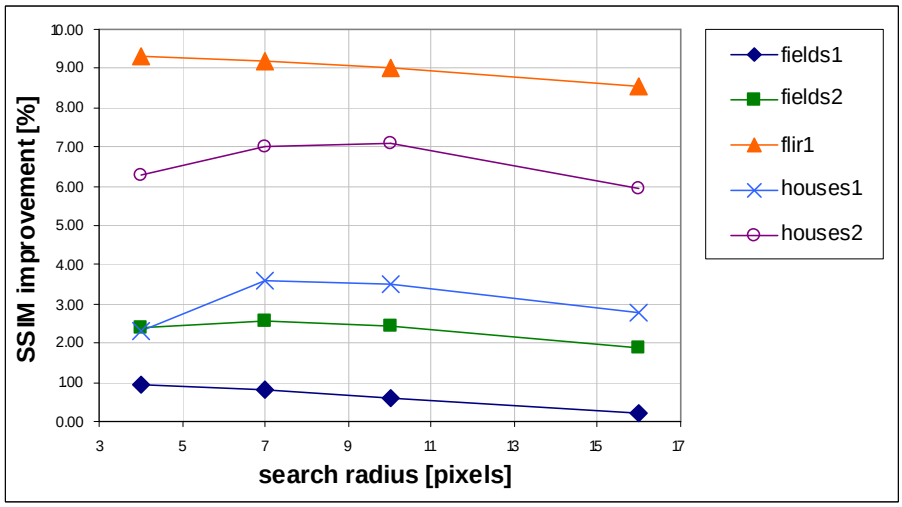
In order to achieve sub-pixel accuracy in block matching motion estimation – it’s possible to upsample the image before applying the algorithm. Simulations showed that

this has little effect on fidelity on motion estimation, and doesn't justify the method. Videos with high turbulence distortions will have especially low benefit from using sub-pixel motion estimation, since the overall motion magnitude is much larger then a pixel.

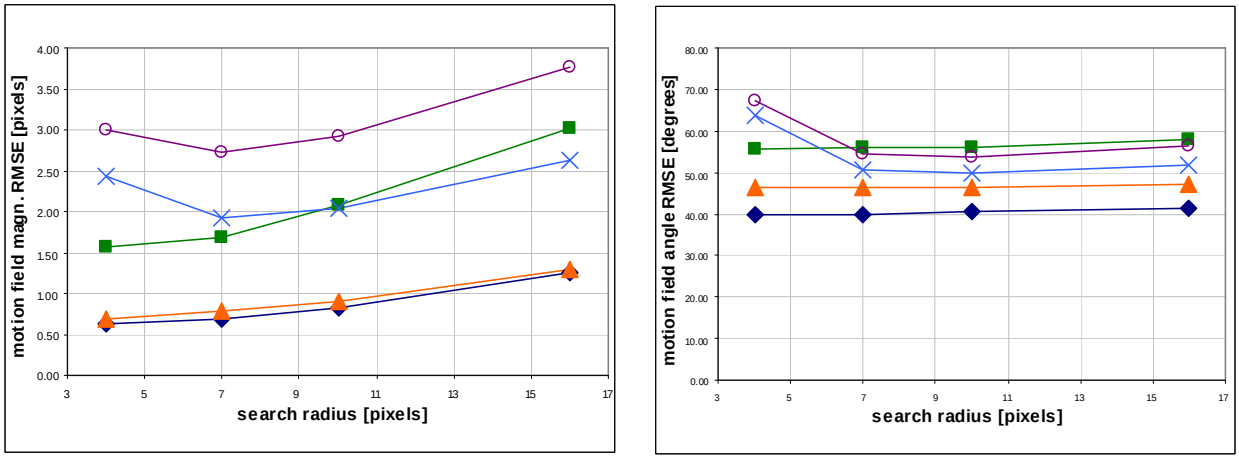
Search radius

Search radius defines the region which is searched for a matching block. A larger radius will enable us to compensate for larger movements (such as in cases of particularly strong turbulence). However this also increases chances for false matches.

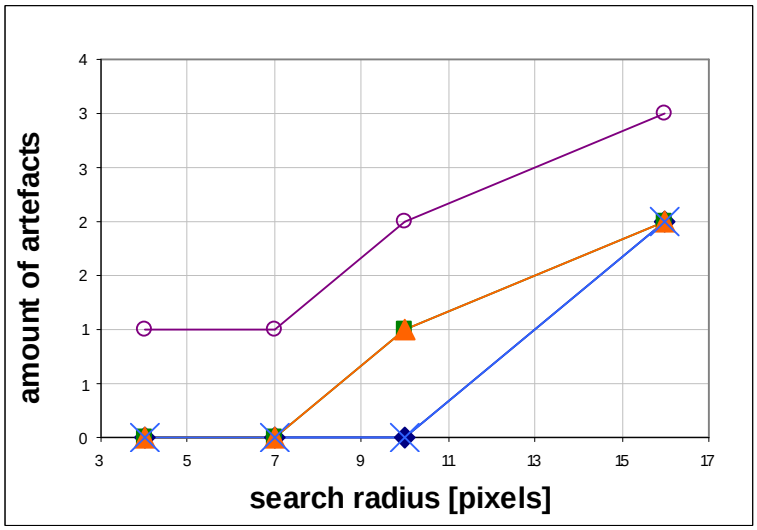
In the following graphs results of simulations are represented, and they show dependence of motion estimation fidelity on search radius. Each curve represents different level of atmospheric turbulence. "SSIM improvement" show the similarity of reconstructed image to original one (higher = better). "Motion field angle RMSE" and "motion field magnitude RMSE" show the errors of estimated motion field (lower = better), and finally the "amount of artifacts" represents the amount of local distortions and artifacts which were determined by visual inspection.



Fidelity of image reconstruction, as function of block search radius (higher = better)



Error in motion estimation, as function of block search radius (lower = better)



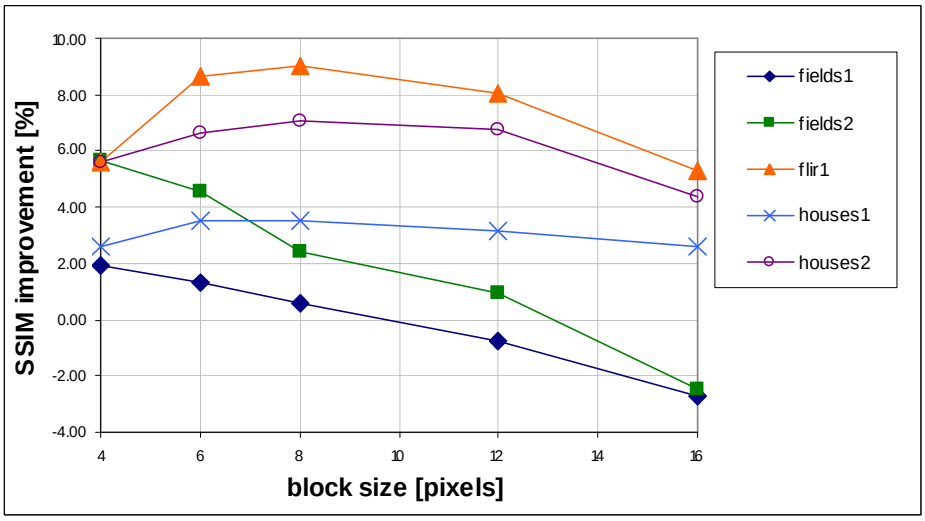
Amount of artifacts as function of block search radius (lower = better)

We can see that search radiuses between 7 to 10 pixels give us the best performance in videos with strong turbulence ("houses1" and "houses2" graphs), both in terms of fidelity of estimated motion field, and image reconstruction. For cases with low to medium level of turbulence – the lowest values of search radius give us the best performance. Also values of search radius higher then 9-10 give increased amount of local distortions and artifacts. They also increase the computational complexity.

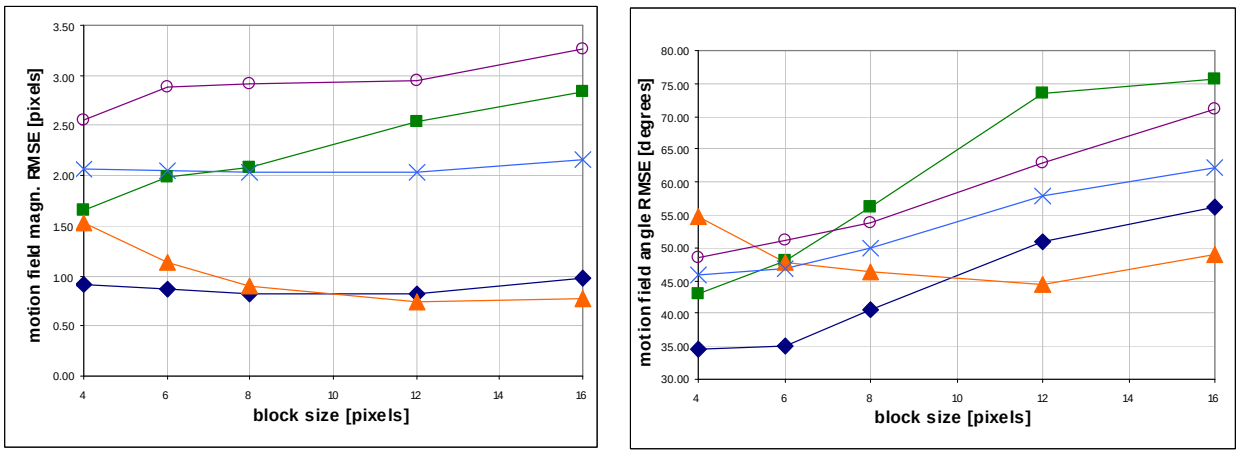
Bottom line: For a strong turbulence best performance is achieved using 8 pixels search radius, and for medium to low turbulence – 5 pixels or less.

Block size

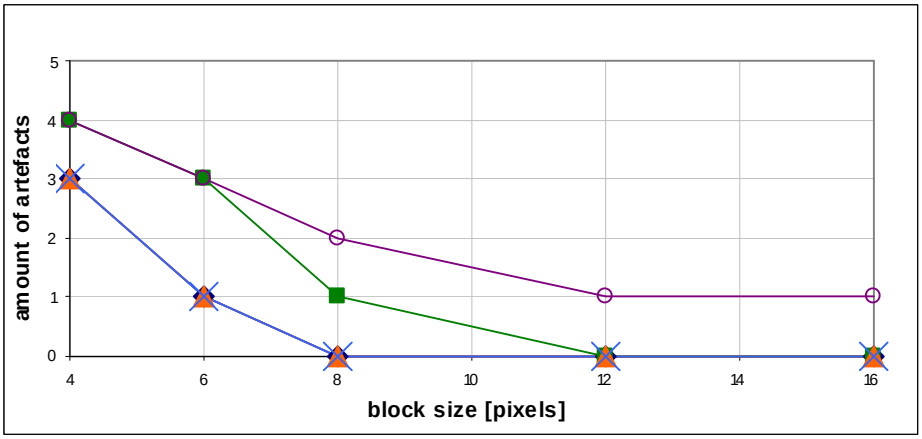
Block size determines the resolution of produced motion field. The smaller the blocks – the higher resolution we achieve, and the finer motion details can be revealed. The following images show performance of motion estimation, as a function of block size:



Fidelity of image reconstruction, as function of block size (higher = better)



Error in motion estimation, as function of block size (lower = better)



Amount of artifacts as function of block size (lower = better)

As we can see – for videos with strong to medium turbulence - block sizes of 6-7 give the best performance in terms of reconstructed image fidelity. For videos with weak turbulence - smaller block sizes achieve better fidelity in terms of SSIM. However we can see that for small block sizes (less then 8) there is increasing amount of distortions and artifacts, which makes use of such values impractical. Analysis of motion estimation field error doesn't give any deterministic result in this case.

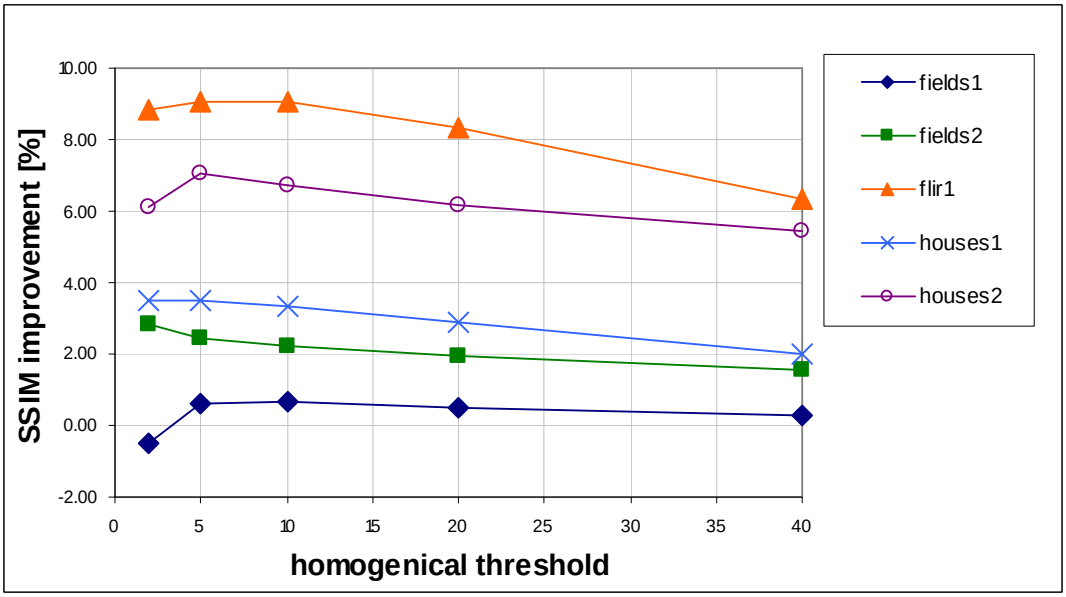
Bottom line: It's best to use block size of 8-10, in order to keep the balance between the fidelity of motion estimation, and amount of artifacts.

Homogeneity level and threshold:

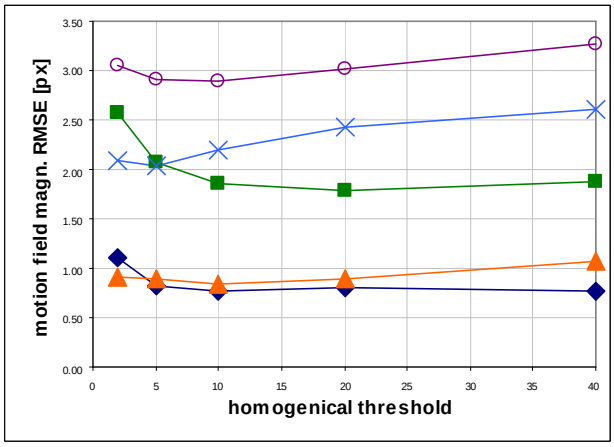
For block matching motion estimation there is a need to determine homogeneity threshold levels for the image, in order to prevent multiple match cases. This is because in case of homogenous areas – every block will show a high match to it's surroundings. By introducing a threshold – we filter out such areas. The optimum for this threshold is generally determined by the noise level, and level of texture.

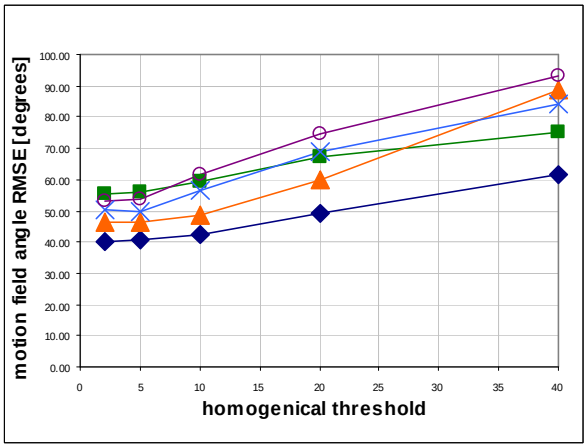
The following method was used for calculating homogeneity: We sort all the pixels from a specific block into a row vector, and calculate difference between highest and the lowest values. Some smoothening filter need to be used prior the calculation, in order to provide robustness against noise.

Homogeneity threshold determines sensitivity of the algorithm to noise, and its robustness. These are results of altering this value, and the effect on motion estimation fidelity:

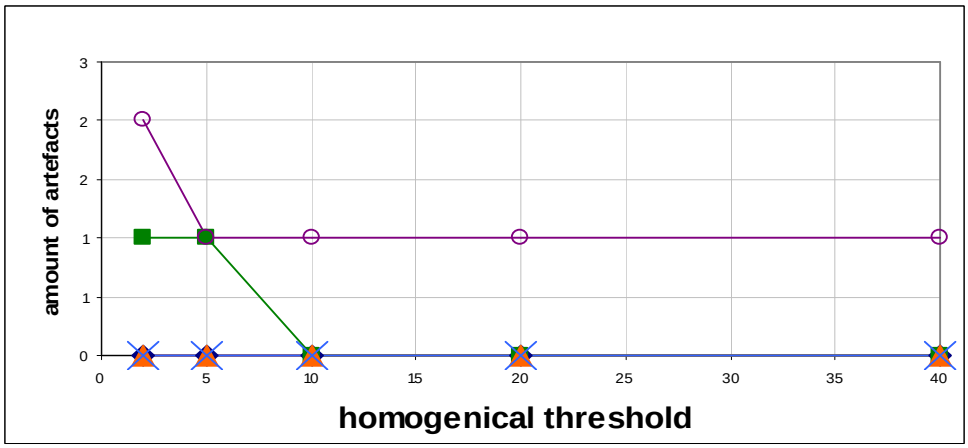


Fidelity of image reconstruction, as function of homogeneity threshold (higher = better)





Error in motion estimation, as function of homogeneity threshold (lower = better)



Amount of artifacts as function of homogeneity threshold (lower = better)

We see that lower homogeneity threshold values give us better performance in all the cases, both in terms of reconstructed image fidelity, and the motion field errors. However values 5 and lower produce increased amount of distortions, probably due to less robustness against noise.

Bottom line: It's best to use homogeneity threshold value of 10.

Filtering the image:

In order to provide robustness against noise we used Gaussian two dimensional convolution filter on image, prior to applying block matching algorithm. Since we assume smooth and elastic motion – such filter doesn't compromise the algorithm fidelity. Two filter parameters we can "play" with are sigma of the Gaussian filter, and size of the convolution filter kernel. These are results of simulations performed with different values of these parameters, while the first graph displays variations of filter size and sigma, as a function of experiment serial number. The other graphs display performance evaluation of motion estimation

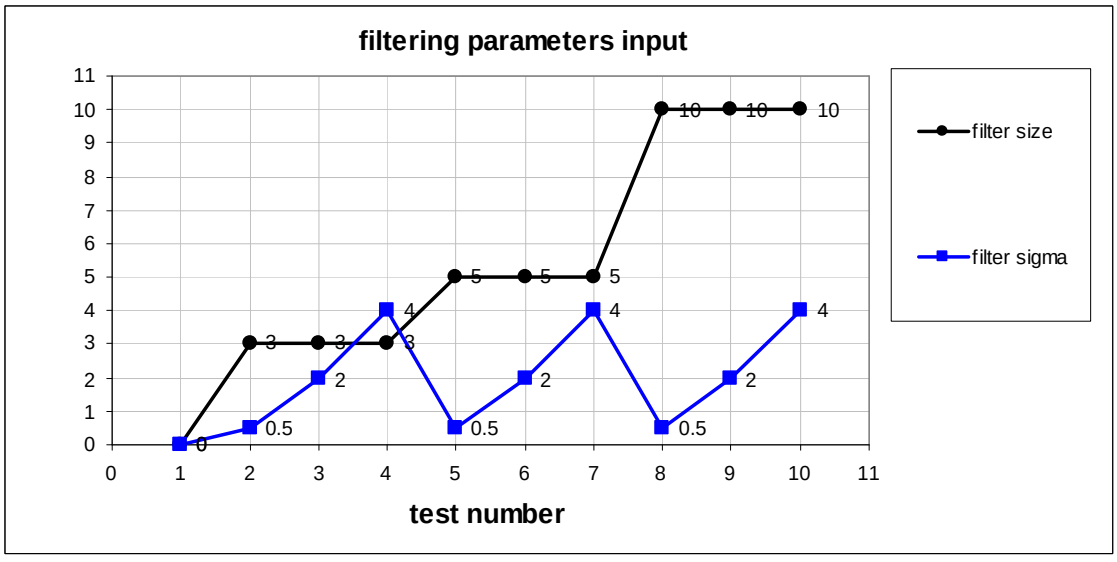
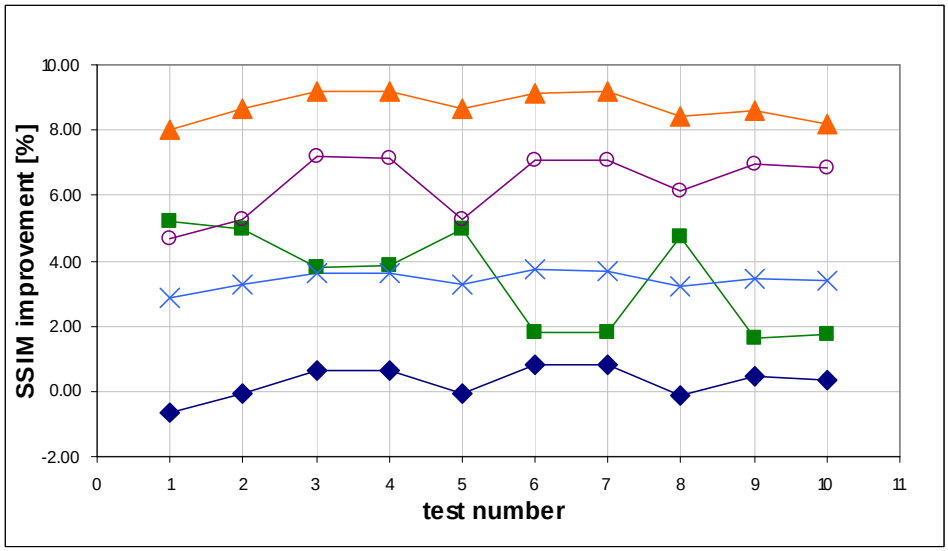
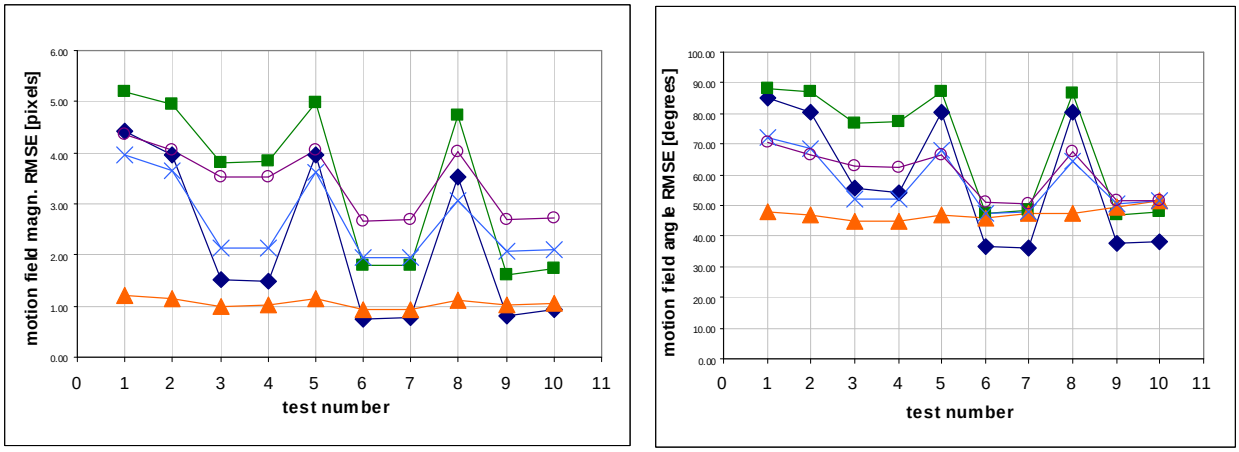


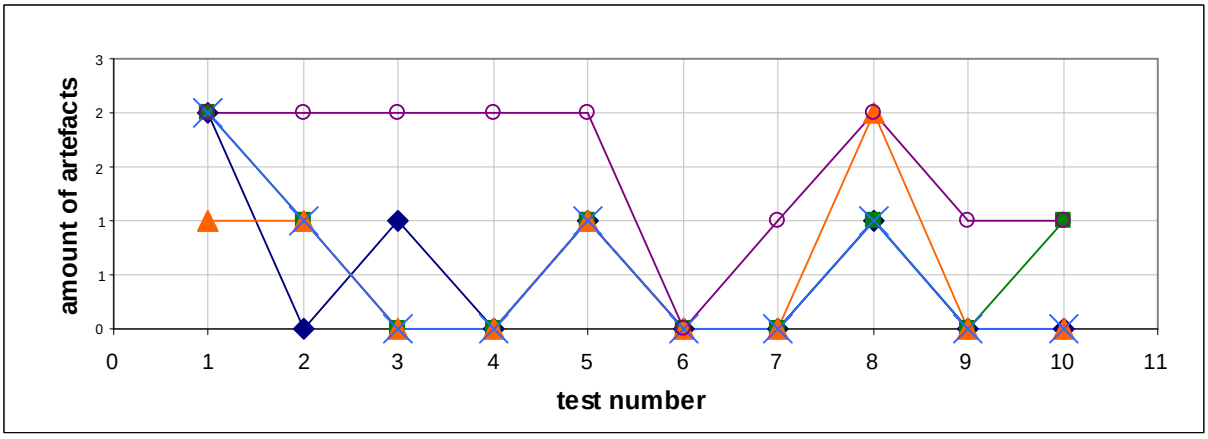
Image pre-filtering parameters and experiment serial number



Fidelity of image reconstruction, as function of filtering parameters (higher = better)



Error in motion estimation, as function of filtering parameters (lower = better)



Amount of artifacts as function of filtering parameters (lower = better)

It's clear that test number 6, with filter size = 5 and sigma = 2, produce best overall results. It's the only case in which there is a low amount of artifacts and distortions for all the cases of turbulence. In terms of estimated field error – this test, along with test number 7 – produce the best overall results. And also in terms of reconstructed image

fidelity – tests number 6 and 7 produce the best result for almost all cases (except “fields2” video test).

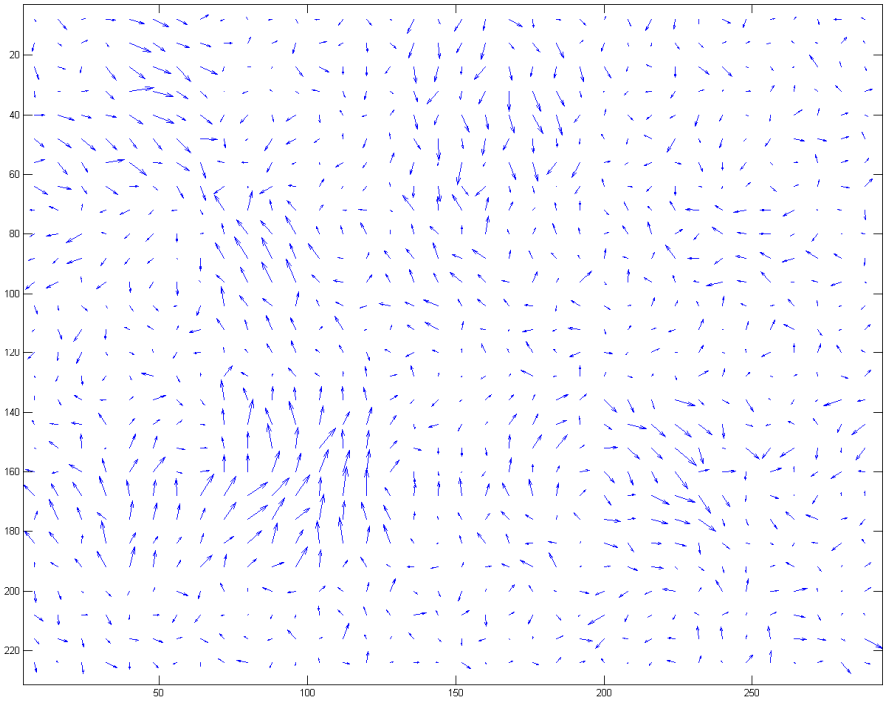
Bottom line: It’s best to filter the image before applying the algorithm, and the filtering parameters which produce the best overall results are: sigma = 2 and kernel size = 5.

Examples of motion fields:

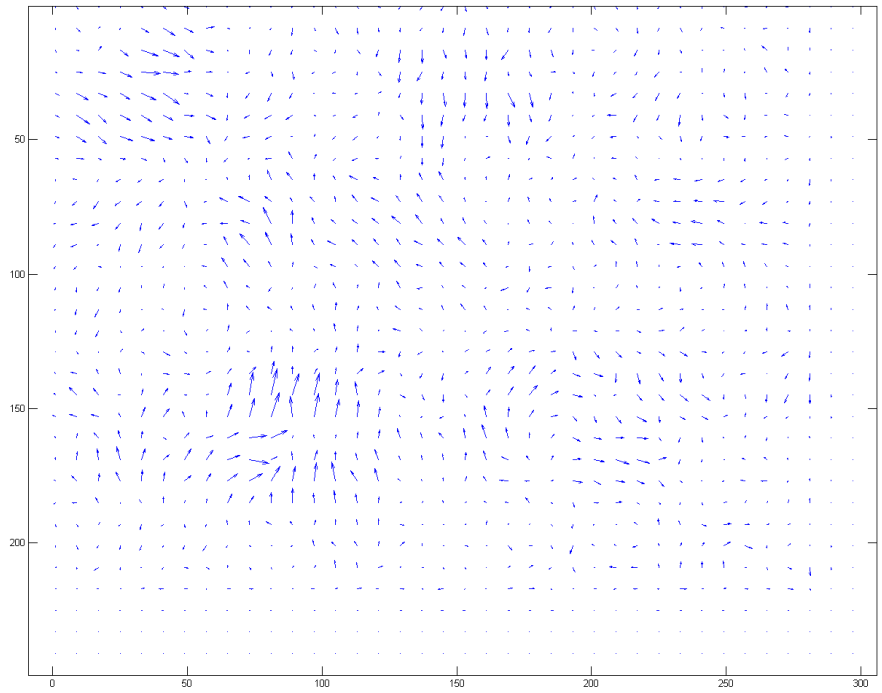
The following are examples of applying several different methods of block matching motion estimation on image which was distorted by simulated atmospheric turbulence.

Original image

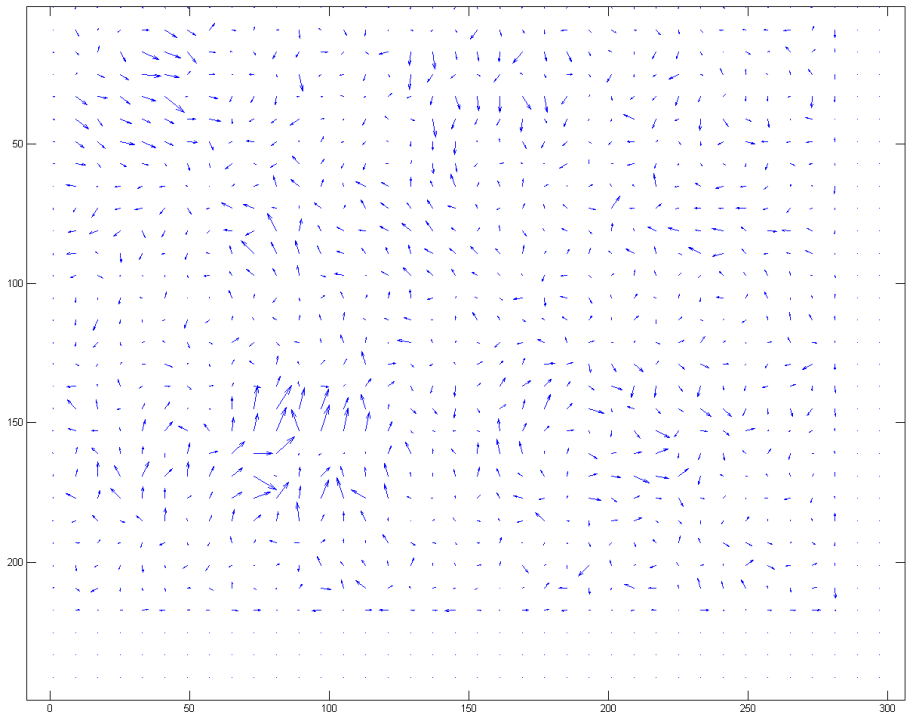
Image influenced by simulated atmospheric turbulence



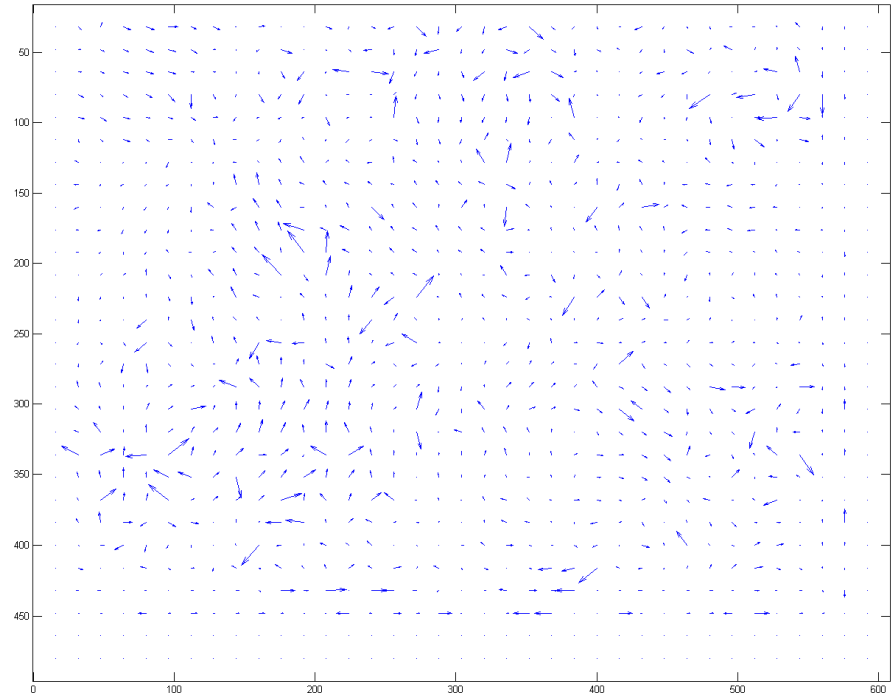
Actual motion field used for warping the images



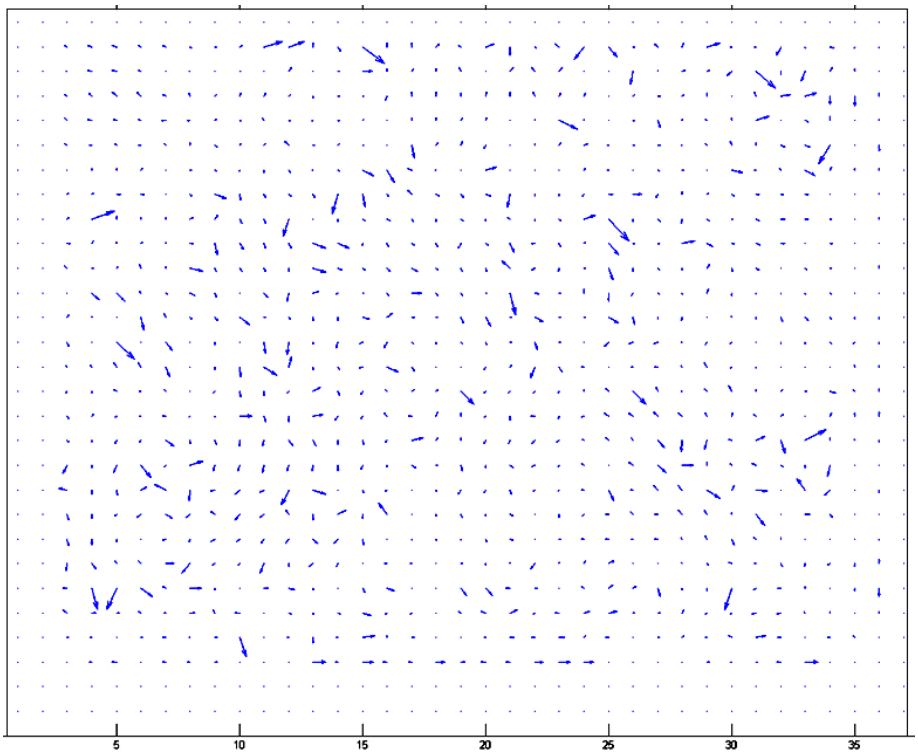
Estimated motion field, using block matching, SAD, strong filtering method



Estimated motion field, using block matching, SAD, weak filtering method



Estimated motion field, using block matching, half-pixel SAD, weak filtering method



Estimated motion field, using block matching, cross-correlation method

Examples of compensated images:

Video with weak turbulence (fields 1):

Distorted frame (t = T1)

Original frame (t = 0)

Compensated image with parameter set which gives best results for block matching motion estimation.

Compensated frame, SAD, filtering size=4, sigma=2, search radius = 4, block size =8

Example of compensated image with high motion estimation fidelity, but which is unusable due to high amount of local distortions and artifacts. In this particular case the distortions happened due to small block size, and large search radius, which caused high amount of false block matching, and as a result - distortions during compensation.

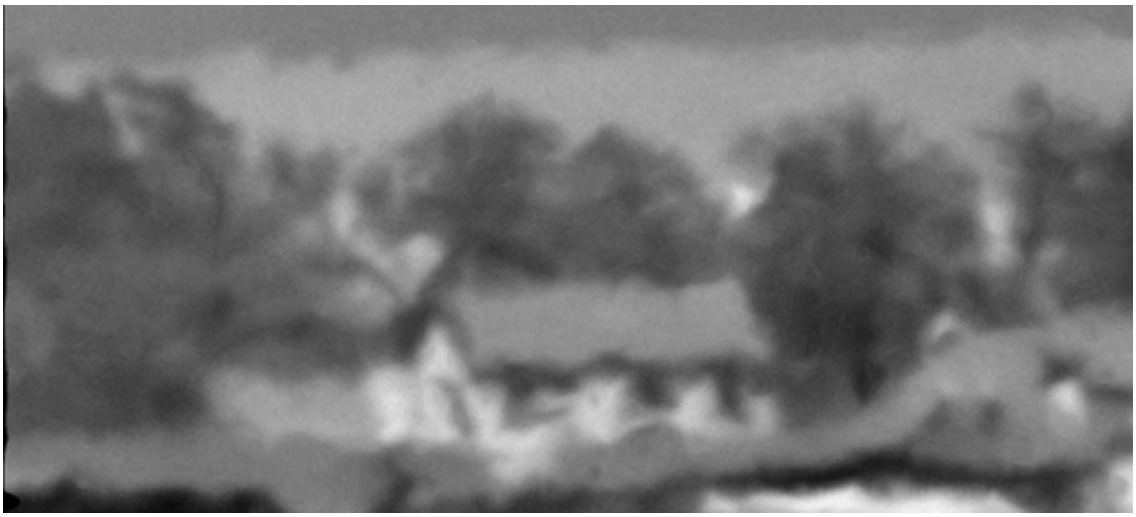
Compensated frame, SAD, filtering size=5, sigma=2, search radius = 10, block size =4

Video with strong turbulence (houses 2):

Distorted frame (t = T1)

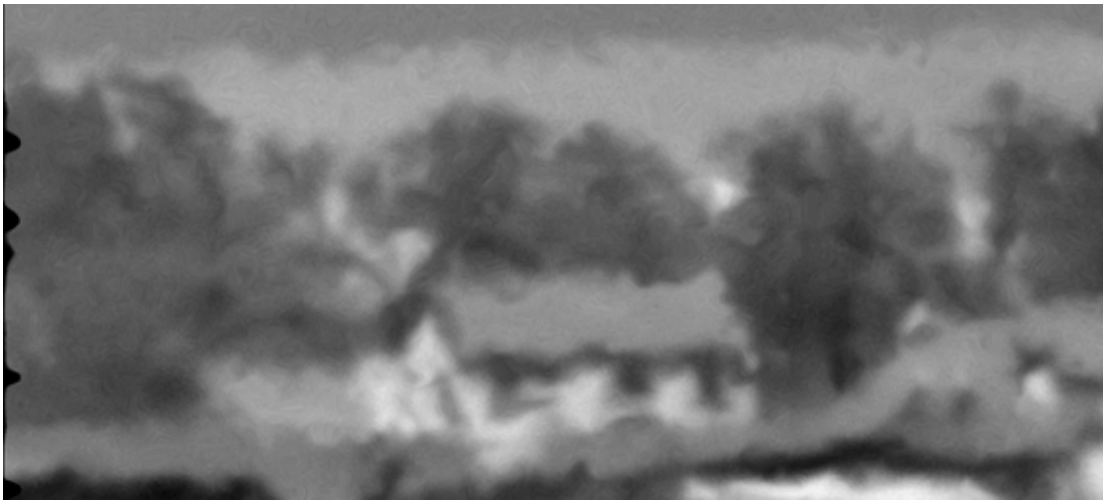
Original frame (t = 0)

Compensated image with parameter set which gives best results for block matching motion estimation.



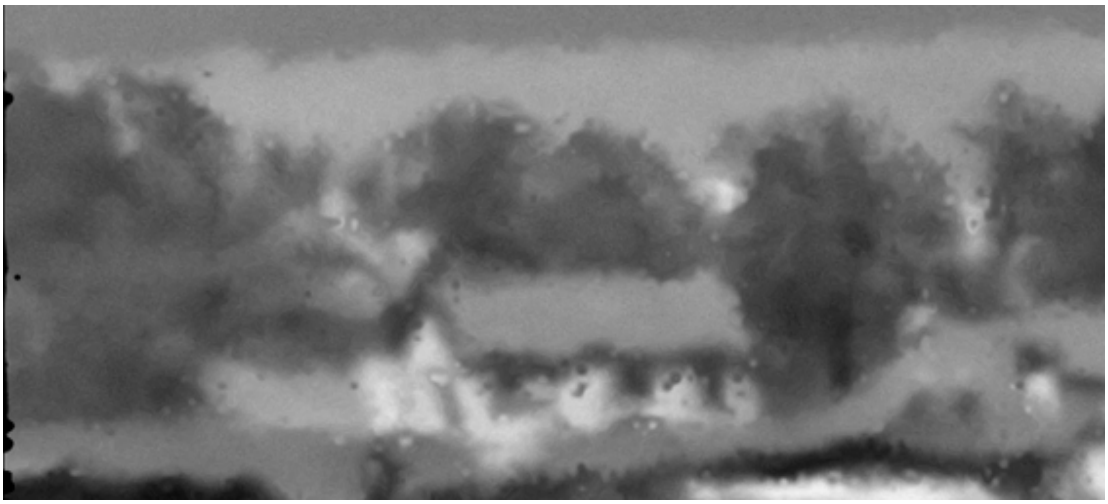
Compensated frame, SAD, filtering size=5, sigma=2, search radius = 10, block size =8

Compensated image without filtering – some distortions are visible.



Compensated frame, SAD, no filtering, search radius = 10, block size =8

Compensated image with high amount of distortions, which make it unusable



Compensated frame, SAD, filtering size=5, sigma=2, search radius = 10, block size =4

Lucas Kanade motion estimation

Lucas Kanade method is a derivative based technique for estimating optical flow. It involves much high computational cost then block-matching based methods, but also produces much more accurate result.

Lucas Kanade is a technique of solving “optical flow equation”. This equation is a result of an assumption that between two consequent frames in a specific image area (a “window) - the intensity is constant. This can be expressed by following equation:

$$I(x, y, t) = I(x + u, y + v, t + 1)$$

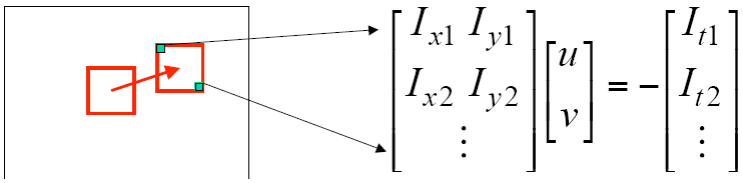
Taylor breakdown of the right side gives us the common form of the equation:

$$I(x, y, t) = I(x, y, t) + \frac{udI}{dx} + \frac{vdI}{dy} + \frac{Idi}{dt}$$

$$I_x u + I_y v = -I_t$$

This equation is ambiguous and impossible to solve, unless we introduce another constraint. This is called “aperture problem” – when it’s impossible to determine exact motion direction of an object moving through an inspection “window”, or aperture.

Lucas Kanade technique introduces this constraint. We assume that within a certain window – the motion is constant for all the points inside the window. Meaning we assume no relative motion between pixels which are really close to each other.



$$\begin{bmatrix} I_{x1} & I_{y1} \\ I_{x2} & I_{y2} \\ \vdots & \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_{t1} \\ I_{t2} \\ \vdots \end{bmatrix}$$

Lucas Kanade method – constant motion constraint

This allows us to build additional equations, one for each pixel inside the window. Solving this equation system using weighted least squares approach – produces two equations, which can be easily solved analytically:

$$A^T A\,\vec{u} = A^T b \rightarrow \vec{u} = \left(A^T A\right)^{-1} A^T b \rightarrow$$

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum I_x^{\,2} & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^{\,2} \end{bmatrix}^{-1} \begin{bmatrix} -\sum I_x I_t \\ -\sum I_y I_t \end{bmatrix}$$

Where (u,v) is the motion vector, Ix, Iy are spatial derivatives (dI/dx and dI/dy).

Lucas Kanade technique has several limitations. First - it requires image to have smooth edges in spatial domain. Therefore usually some sort of low pass blurring filter (i.e. Gaussian) is used in spatial domain in order to smoothen out the image, and ensure continuous derivatives.

Also this method assumes smooth or so called “elastic” motion to a certain degree. Since atmospheric turbulence also causes smooth, continuous motion flow – this doesn’t pose a problem for our purpose.

Often multi-scale motion estimation technique is applied (pyramides), in order to allow detection of large movements. However since atmospheric turbulence induces motion on a small scale only – use of this technique isn’t necessary in our case.

Similarly to block matching - In order to provide robustness to noise - we are forcing motion vector to be zero in homogenous regions (above a certain threshold).

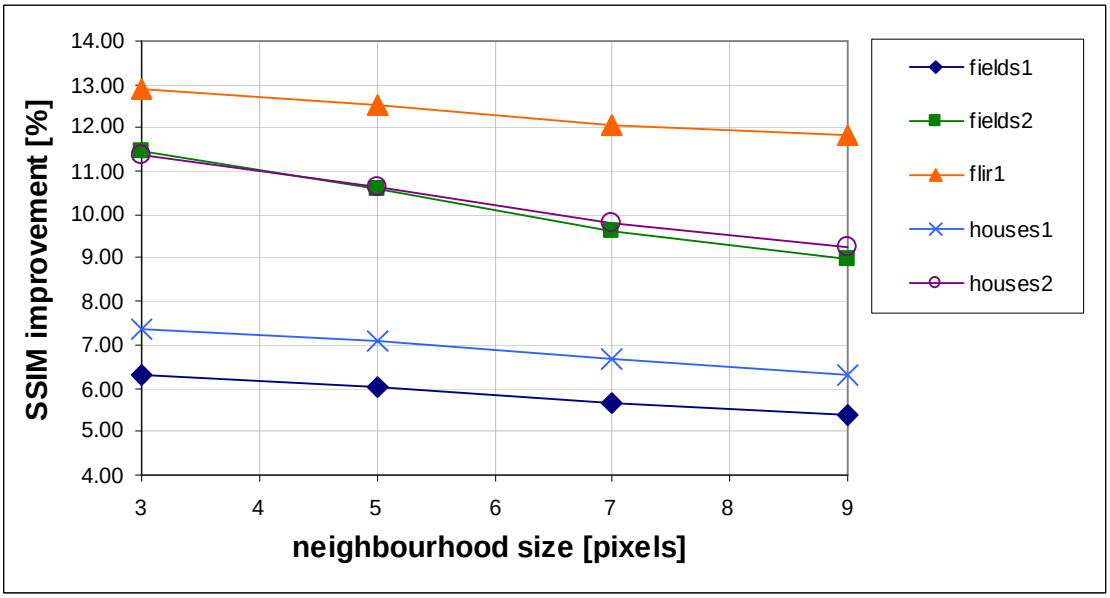
Variants and results of Lucas Kanade motion estimation:

Lucas Kanade method also has “tuning” parameters, which can be adjusted in order to achieve better performance.

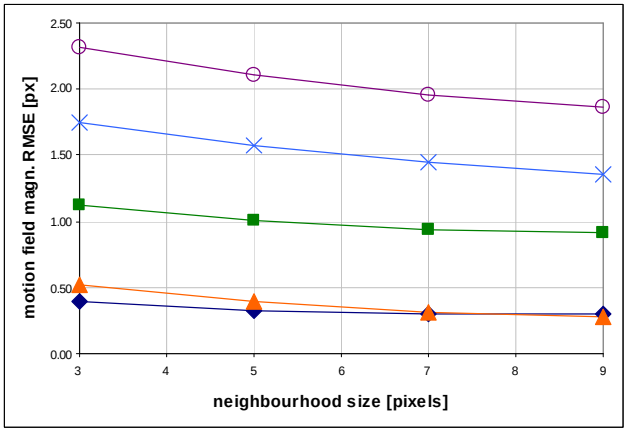
Neighborhood size

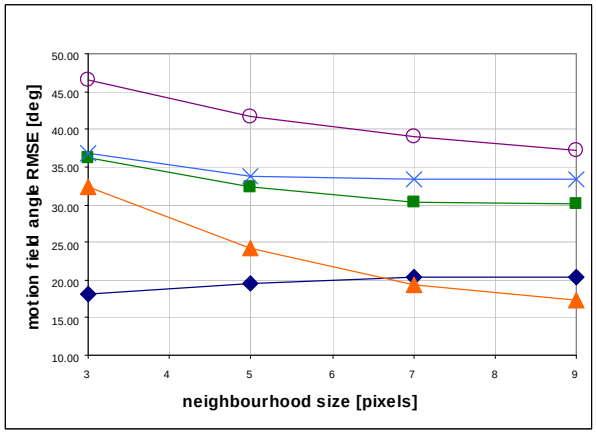
This parameter defines the “window” which is used to solve the aperture problem. A larger window assumes smoother motion, less sensitive to fine motion details, but also more robust to noise.

In the following graphs appear results of simulations, which show dependence of motion estimation fidelity on neighborhood size. Each curve represents different level of atmospheric turbulence. “SSIM improvements” shows the similarity of reconstructed image to original one (higher = better). “Motion field angle RMSE” and “motion field magnitude RMSE” show the errors of estimated motion field (lower = better), and finally the “amount of artifacts” represents the number of local distortions and artifacts which were determined by visual inspection.

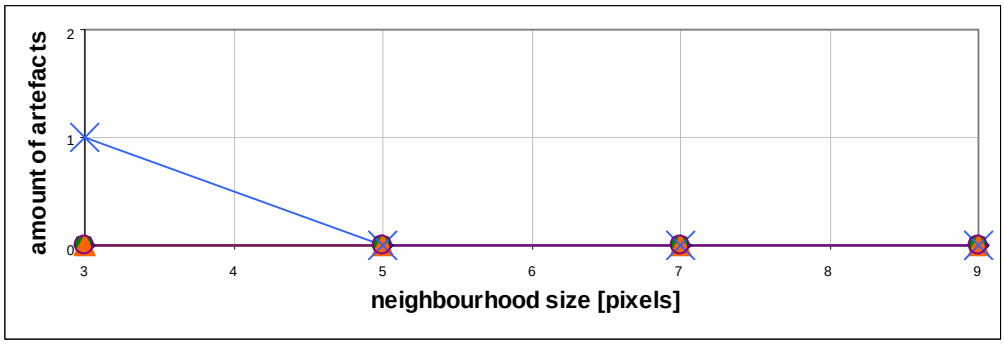


Fidelity of image reconstruction, as function of neighborhood size (higher = better)





Error in motion estimation, as function of neighborhood size (lower = better)



Amount of artifacts as function of neighborhood size (lower = better)

Generally – as long as neighborhood size is within reasonable limits (3 to 10) – It makes little effect on fidelity of motion estimation (visually indistinguishable). For very low values of 3 – small distortions begin to appear in some cases, therefore slightly higher values should be used. In terms of structural similarity of compensated image – smaller neighborhood values give slightly better results. Smaller neighborhood also requires less computational cost.

Bottom line: Neighborhood size of 5 can be safely used in most cases with good performance results.

Image filtering

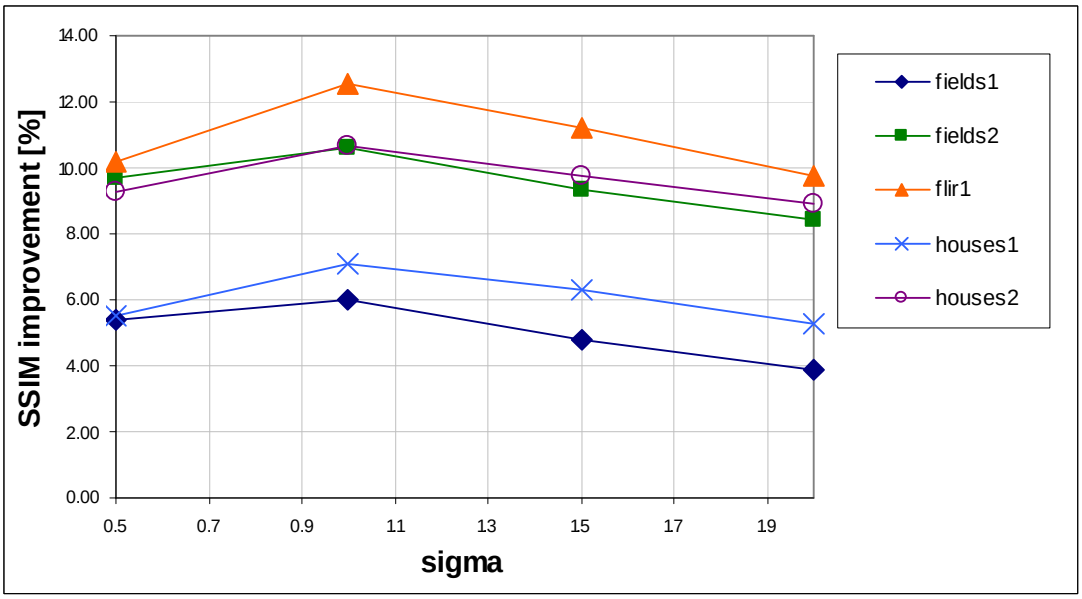
As was mentioned above – it is required to filter the image, in order to smoothen it, before applying the motion estimation algorithm.

We applied a low pass filter with Gaussian kernel, which is commonly used in this case:

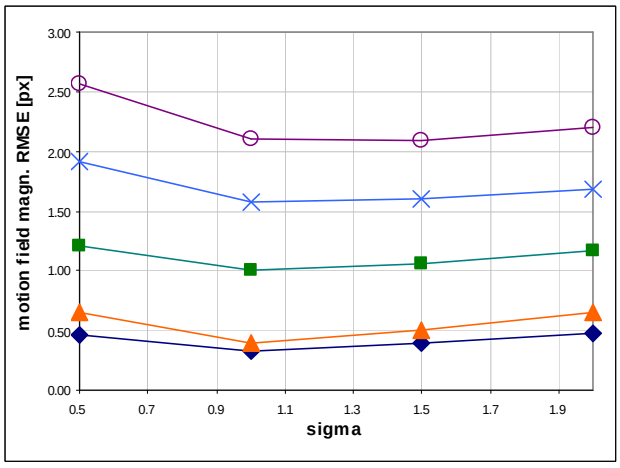
$$K = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{((x-k-1)^2+(y-k-1)^2)}{2\cdot\sigma^2}\right) \qquad k = \frac{6\sigma}{2}$$

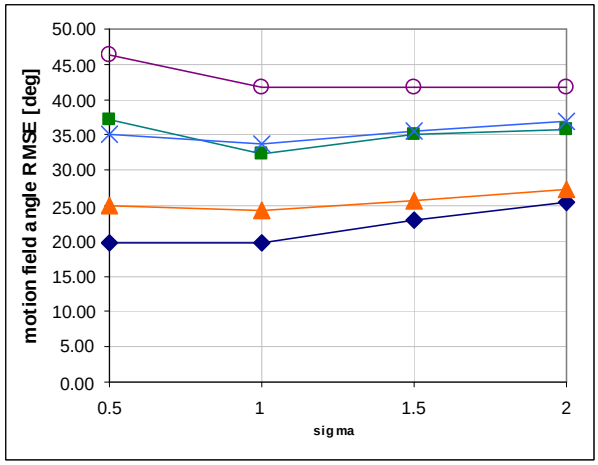
Where (x,y) are spatial coordinates, k is filter size, and σ is sigma.

The following graphs show effect of filtering strength (sigma) on motion estimation performance:

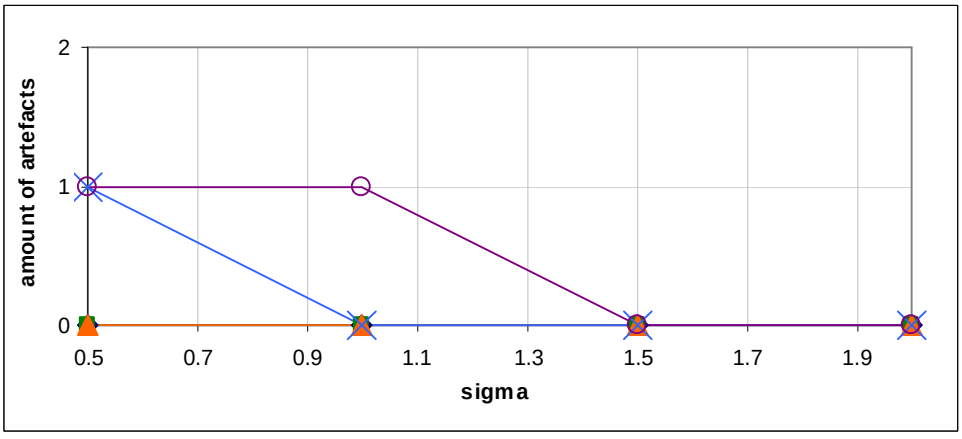


Fidelity of image reconstruction, as function of image-filtering sigma (higher = better)





Error in motion estimation, as function of image-filtering sigma (lower = better)



Amount of artifacts as function of image-filtering sigma (lower = better)

We can see that in terms of compensated image structural similarity – a sigma value of 1 produces the best result for all cases of turbulence. Same happens in terms estimated motion field magnitude and angle errors. Visual inspection confirms that reconstructed

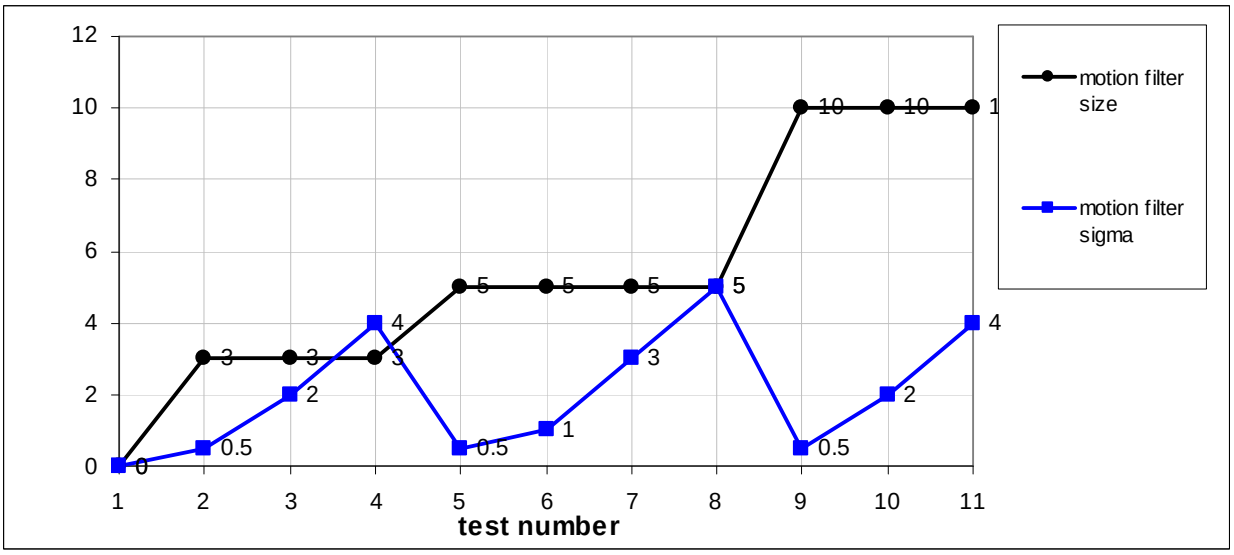
images with sigma value which is higher then 1 – differ significantly from the original image. For videos with extremely strong levels of turbulence – higher values of sigma give us slightly “cleaner” images. However this effect is so small, that only extreme cases should justify use of sigma value higher then 1.

Bottom line: In most cases: Gaussian kernel with sigma = 1 gives significantly better results then other values. In cases with extremely strong turbulence – use of slightly higher sigma value can be considered, in order to produce “cleaner” image, while sacrificing the motion estimation accuracy.

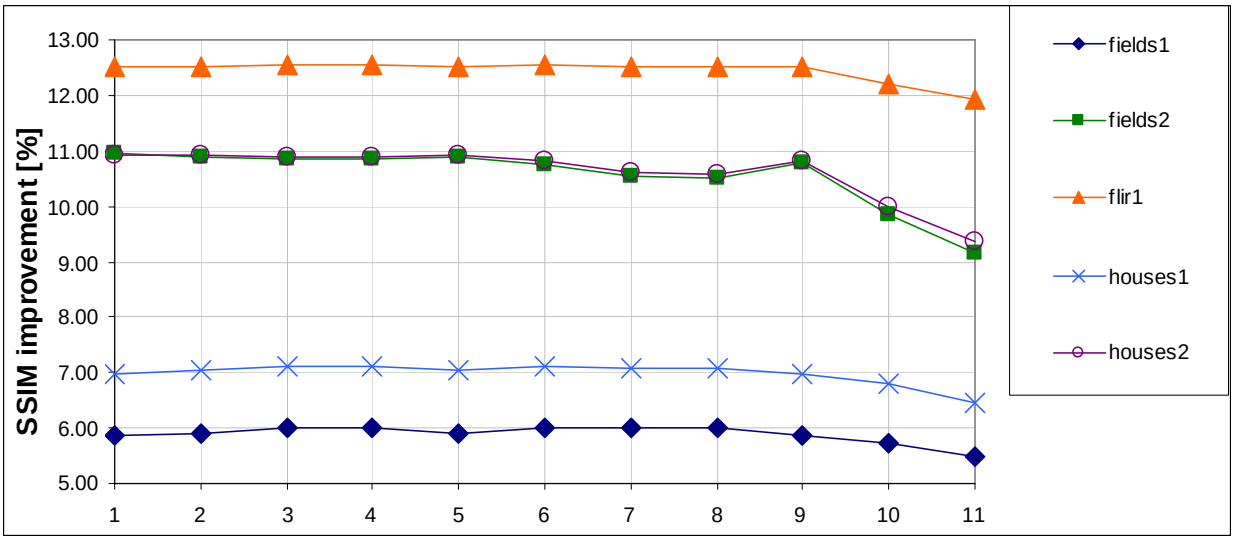
Filtering the motion field:

As opposed to block matching motion estimation – Lucas Kanade method produces motion vector for each image pixel, and not only for each block. Since we assume elastic and smooth motion – it can be beneficial to apply a low pass Gaussian filter on produced motion field result.

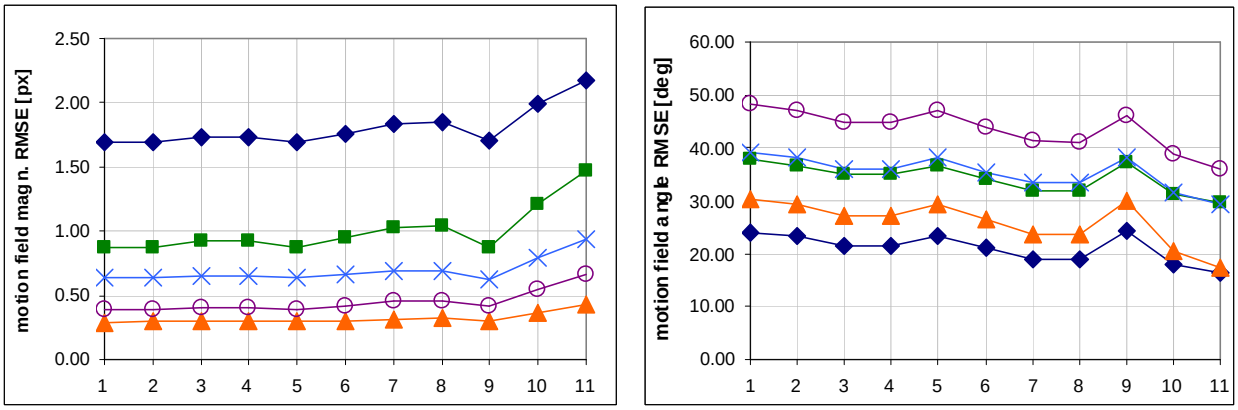
Two filter parameters we can “play” with are sigma of the Gaussian kernel, and its size. These are results of simulations performed with different values of these parameters. The first graph displays variations of filter size and sigma, as a function of experiment serial number. The other graphs display performance evaluation of motion estimation as a function of the same experiment numbers.



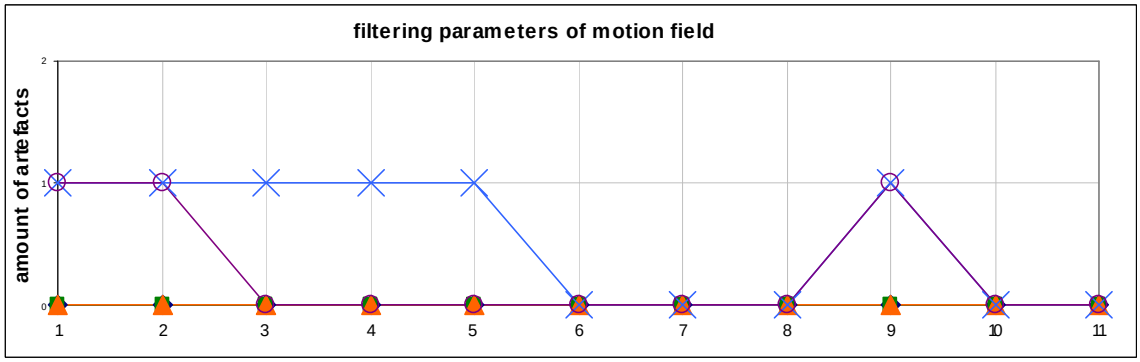
Motion field filtering parameters and experiment serial number



Fidelity of image reconstruction, as function of filtering parameters (higher = better)



Error in motion estimation, as function of filtering parameters (lower = better)



Amount of artifacts as function of filtering parameters (lower = better)

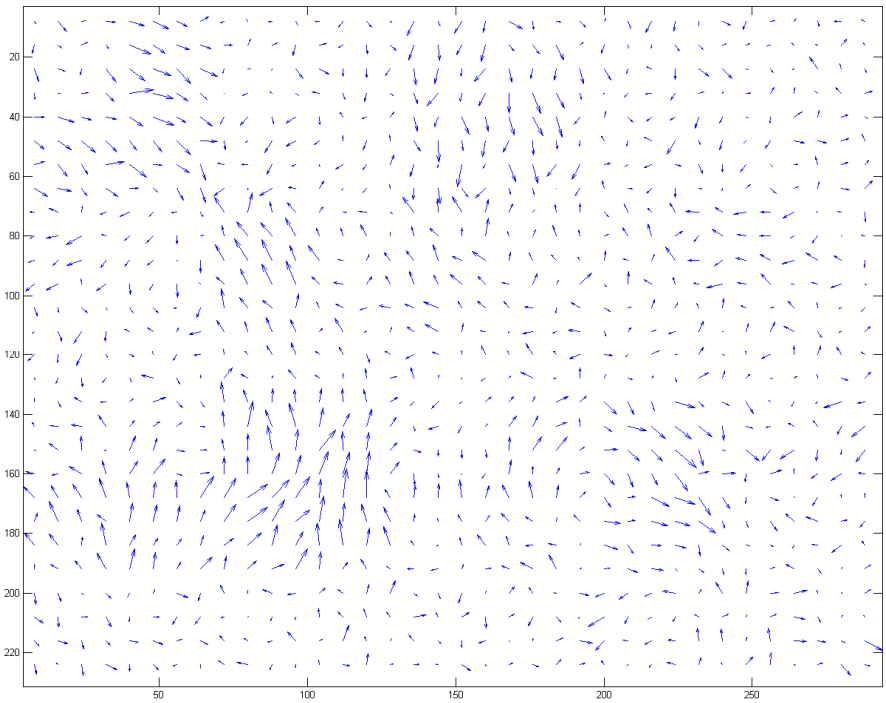
In terms of compensated image structural similarity - cases number 10, 11 are least accurate (due to strong filtering), while cases number 6, 7, 8 produce results which are fairly similar, or a little bit less accurate then all other cases. Lower values of sigma in these cases produces better results. Similar trend can be observed with error in estimated motion field magnitude. Motion field direction error, however, shows improvement in cases 6, 7, 8, and even more improvement in cases 10, 11.

However due to small differences in terms of motion field estimation fidelity in all these cases – main concern is achieving low amount of local distortions and artifacts. Visual inspection shows that for videos with weak turbulence – filtering the motion field doesn't produce big difference. However for videos with strong to very strong turbulence ("houses1" and "houses2") – visual inspection revealed significant amount of distortions and artifacts in most cases, except cases number 6, 7, 8 (with filter kernel size = 5) and cases number 10, 11 (with kernel size 10 and sigma 2 and 4 respectfully) which produced clean images.

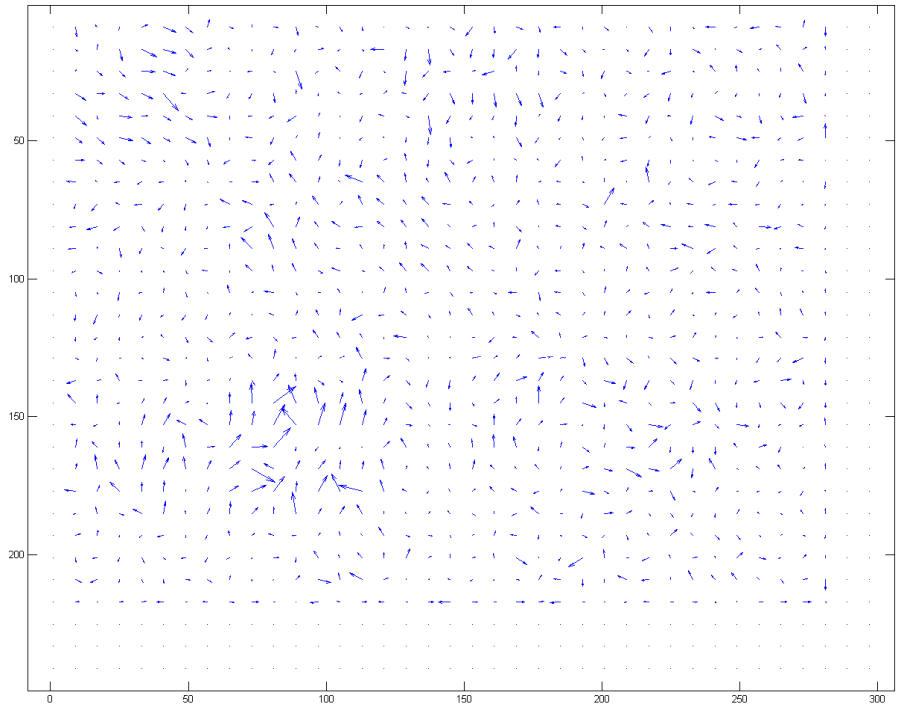
Bottom line:
Filtering the motion field improves the compensated image in terms of distortions and artifacts, and recommended values for Gaussian filter kernel are: size=5 and sigma=1. For images with especially strong turbulence: consider using larger kernel (10) with larger sigma (2-4).

Examples of motion field:

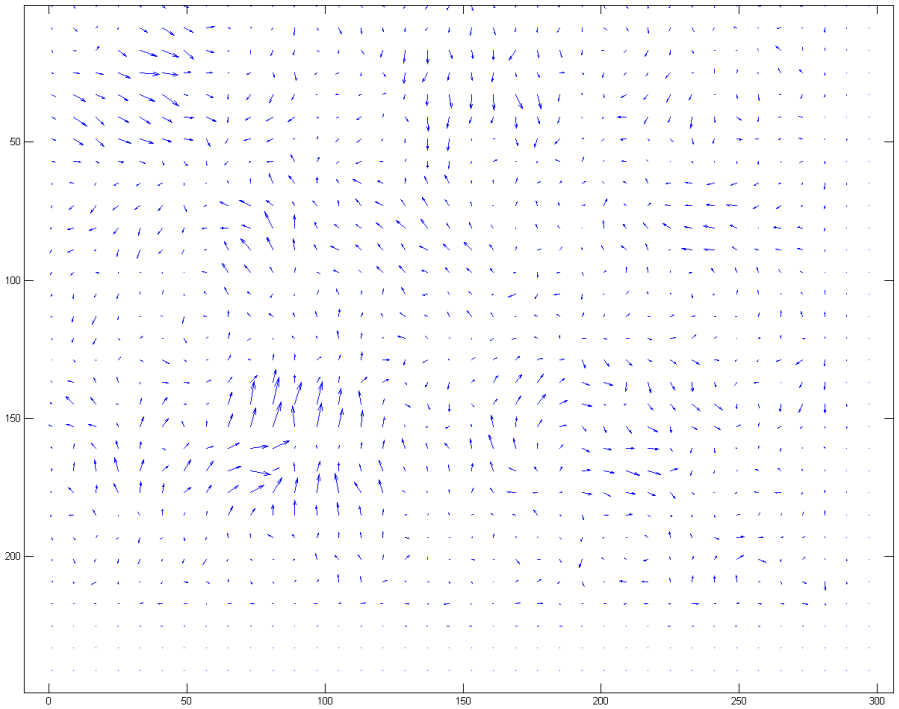
We used the same distorted image example which was used in block-matching motion estimation section:



Actual motion field used for warping the images



Estimated motion field, using Lucas Kanade method, no motion field filtering



Estimated motion field, using Lucas Kanade method, strong motion field filtering

<u>Examples of compensated images:</u>

Video with weak turbulence (fields 1):

Distorted frame (t = T1)

Original frame (t = 0)

Compensated image with parameter set which gives best results for LK method

Compensated frame, LK, image filter sigma=1, neighborhood=3, field filter size=5, sigma=2

Video with strong turbulence (houses 2):

Distorted frame (t = T1)

Original frame (t = 0)

Compensated image with parameter set which gives best results for LK method

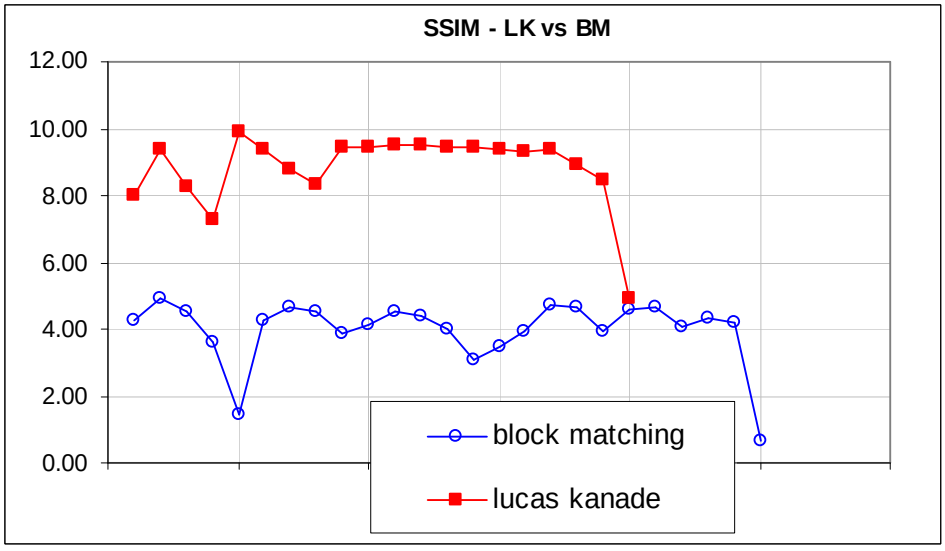
Compensated frame, LK, image filter sigma=1, neighborhood=5 , field filter sigma=2, size=5

Compensated image without motion field filtering – some distortions are visible.

Compensated frame, LK, image filter sigma=1, neighborhood=5 no motion field filter

Comparison of Lucas Kanade and Block Matching motion estimation:

The following graphs demonstrate the difference in improvement of image structural similarity, when it was reconstructed using both Lucas Kanade and block matching motion estimated techniques:



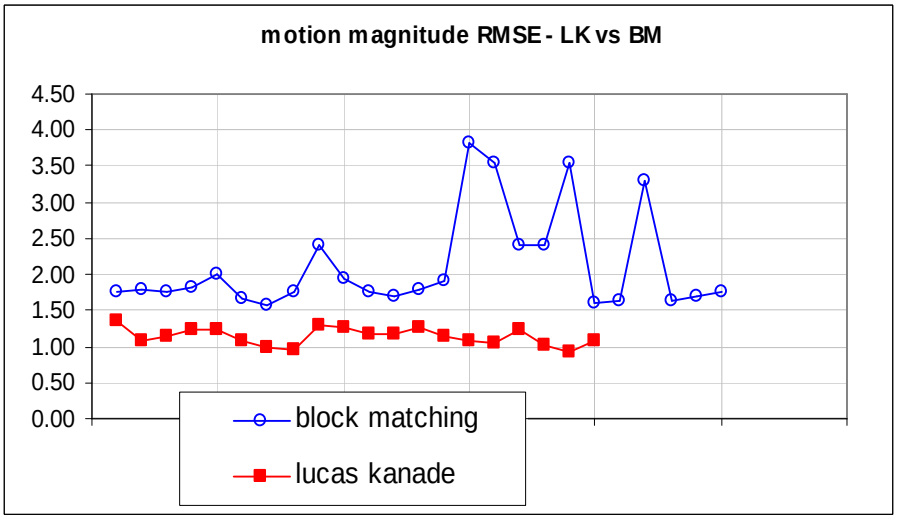
Comparison of SSIM improvement between LK and BM motion estimation methods (high = better)

Y axis shows the % of SSIM improvement, compared between distorted image, and the one restored using motion estimation technique. X axis shows different sets of parameters and variants. There is no correlation between X values for both graphs; therefore only the higher values should be taken into account, since they represent sets of parameters which produce the best result averages for all turbulence cases.

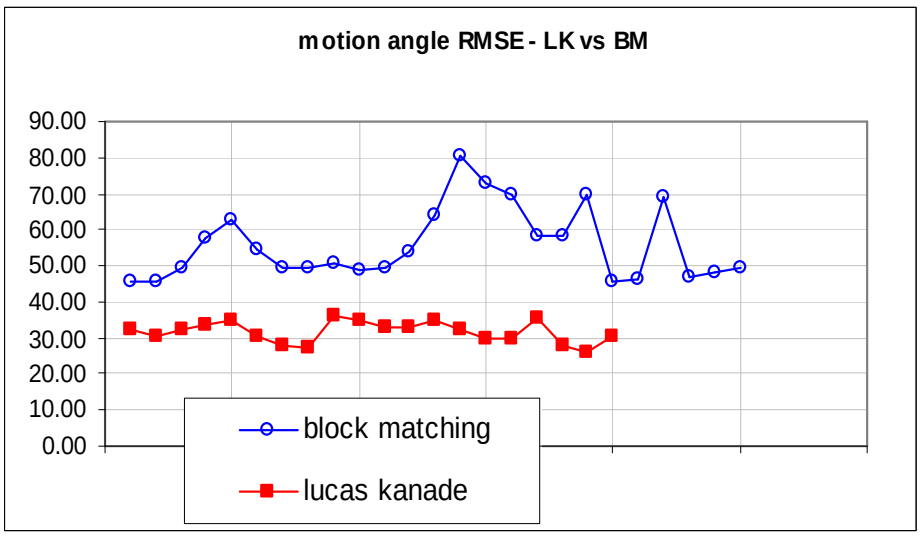
It is clear that Lucas Kanade technique in most of its variants – produces superior results which are approximately twice as accurate as block matching method.

Note that last result, which shows significant drop of SSIM improvement in both cases – is the case where no upsampling for motion compensation was used to achieve sub-pixel accuracy. This clearly shows importance of performing it.

The following graphs display comparison of magnitude, and the angle errors between LK and BM motion estimation techniques:



Comparison of errors in motion field magnitude errors between LK and BM methods (low = better)



Comparison of errors in motion field direction errors between LK and BM methods (low = better)

We can see that in this case also – Lucas Kanade technique is superior to block matching, and produces much less errors both in direction of estimated motion vectors, and in their magnitudes.

Conclusions

- Lucas Kanade motion estimation technique is highly superior over block matching based methods, and produces much more accurate result with fewer distortions in all cases. However LK technique has higher computational cost
- For both block matching and Lucas Kanade methods – set of “tuning” parameters and filters can be defined, and allow to improve the estimation performance.
- It was found that PSNR criteria for comparing the images is highly unreliable and unusable in cases of analyzing atmospheric turbulence and its compensation. Instead of PSNR – SSIM (Structural Similarity) index should be used.
- In order to successfully evaluate fidelity of estimated motion field – direct statistical comparison of motion fields can and should be used. This requires, however, a turbulence simulation model to be implemented.
- We showed that filtering the estimated motion field which was produced by Lucas Kanade algorithm - achieves more accurate and clean result.
- Optimal set of parameters for motion estimation technique, which gives the best average results in most cases, was found:
  Lucas Kanade method, neighborhood size = 5, Gaussian low pass filter of the processed image with sigma = 1 and size = 3, Gaussian low pass filter of estimated motion field with size = 5 and sigma = 1, upsampling ratio for motion compensation process = 2.
- For both simulation of the atmospheric turbulence effect, and compensation of this effect using estimated motion field – using cubic spline interpolation technique produces much better results then using bi-linear method.
- A good reference frame is required to achieve good results in compensating motion induced by atmospheric turbulence. A time averaged frame over a certain period of time would be the best choice. A single frame over specific time intervals is a faster method, but it produces less accurate results.
- It was found that in cases with strong turbulence - block matching technique gives results with higher accuracy then the same technique in cases with weak turbulence.

References

[1] Li, Dalong, et al. "New method for suppressing optical turbulence in video." Proceedings of the European Signal Processing Conference (EUSIPCO). 2005.

[3] Barreto, Dacil, L. D. Alvarez, and J. Abad. "Motion estimation techniques in super-resolution image reconstruction a performance evaluation." Virtual Observatory. Plate Content Digitalization, Archive Mining and Image Sequence Processing (2005): 254-268.

[4] Li, Dalong, Russell M. Mersereau, and Steven Simske. "Blur identification based on kurtosis minimization." Image Processing, 2005. ICIP 2005. IEEE International Conference on. Vol. 1. IEEE, 2005.

[5] Gepshtein, Shai, et al. "Restoration of atmospheric turbulent video containing real motion using rank filtering and elastic image registration." Proceedings of the 12th European Signal Processing Conference (EUSIPCO). 2004.

[6] Shimizu, Masao, et al. "Super-resolution from image sequence under influence of hot-air optical turbulence." Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on. IEEE, 2008.

[7] Hayakawa, Hitoshi, and Tadashi Shibata. "Block-matching-based motion field generation utilizing directional edge displacement." Computers & Electrical Engineering 36.4 (2010): 617-625.

[9] Quirrenbach, Andreas. "The effects of atmospheric turbulence on astronomical observations." A. Extrasolar planets. Saas-Fee Advanced Course 31 137 (2006): 137.

[10] Burger, Liesl, Igor A. Litvin, and Andrew Forbes. "Simulating atmospheric turbulence using a phase-only spatial light modulator." South African Journal of Science 104.3-4 (2008): 129-134.