OPTIMIZATION BASED IMAGE REGISTRATION IN THE PRESENCE OF MOVING OBJECTS

F. Karimi Nejadasl[1], **B. G. H. Gorte**[1], **Serge P. Hoogendoorn**[2], **and M. Snellen**[1]

[1]Institute of Earth Observation and Space System
Delft University of Technology
Kluyverweg 1, 2629 HS, Delft, The Netherlands
Tel: +31 15 27 88337, Fax: +31 15 27 82348
F.KarimiNejadasl@tudelft.nl,B.G.H.Gorte@tudelft.nl
[2]Transport and planning section
Delft University of Technology
Stevinweg 1, 2628 CN, Delft, The Netherlands
S.P.Hoogendoorn@tudelft.nl

KEY WORDS: Registration, Optimization, Differential Evolution, Nelder-Mead, 3D Euclidean

ABSTRACT:

To increase robustness in registration of image sequences, we investigate a featureless method. This paper formulates the registration problem as an optimization of an energy function between a reference image and a transformed of target image. A result parameters are estimated using a global optimizer, Differential Evolution, followed by a local optimizer, Nelder-Mead. Our experiments show that the proposed algorithm perform, robustly in a large variety of image content from the road almost empty surrounding to more cluttered one and from simple road shape to more complex.

1 INTRODUCTION

This paper describes an algorithm to co-register images in an image sequence that is recorded from a non-stable platform, in this case a helicopter hovering above a highway. Such image sequences are used to collect statistics concerning the driver behavior in busy (nearly congested) traffic. Typically, we record highway stretches with a length of 300-500m during one hour or more. We use a b/w camera with 1300*1030 pixels, which gives a ground resolution of approx. 25-40 cm, at a frame rate of 10 fps. Because of the large data volumes, we aim at fully automatic image analysis, which means that for each vehicle in the scene we record the position on the highway as a function of time (in 0.1s increments). The accuracy and precision of the recordings have to be such that reliable estimates can be derived for the speeds, accelerations/decelerations and reaction times (when does a driver start to brake after its predecessor does?).

An important step is the co-registration of all images in the sequence. Turbulence generated by a helicopter that is hovering at one position causes random movements leading to severe deterioration of platform stability. The pilot has the difficult task to prevent the helicopter from (slowly) drifting away from the wanted position, and she/he definitely cannot control random movements, that cause misalignments of images in the sequence.

From a typical flying height in the order of 400m and with about 55 degree viewing angle, consecutive images (recorded at 0.1s intervals) show misalignments of up to 3m.

Misalignment is caused by the combined effects of helicopter translations (in x, y and z directions) and rotations (around the x, y and z axes). Using a Gyro stabilizer on a camera dampens out a rotor or motor vibration in turbulence caused by helicopter during a hovering time. However, it cannot provide the stabilized image sequence over a long time, for example half an hour, due to accumulation of movements with increasing time. Moreover, the Gyro stabilizer cannot prevent effects of slower rotation or (any) translation. Therefore we opted for a software post-processing solution to obtain fully automatic co-registration.

In this solution the image is processed frame by frame, starting from a reference frame, which (for simplicity) we will assume to be frame 1 in the sequence. Assuming that frames $2 \cdots n$ are already registered to frame 1, which means that the transformation $T_{i,1}$ between frame i and frame 1 is known, we process frame $i+1$ in three steps as follows:

1. compute $T_{i+1,i}$, the transformation between image intensity I_{i+1} and image intensity I_i
2. compute $E_{i+1,1} = T_{i,1}T_{i+1,i}$, the estimate transformation between I_{i+1} and I_1
3. used $E_{i+1,1}$ as the initial value for computing $T_{i+1,1}$

This strategy prevents registration errors to accumulate. Matching consecutive images (step 1) is easier (i.e. less error-prone) than matching arbitrary images, since the misalignment is limited. In step 3, this problem is avoided by providing an accurate approximate value to the matching process. The first step is still the most demanding one and is the main focus of the paper: how to register consecutive images from a sequence.

Usually, the transformation between two images is calculated based on common features, which have to be identified first by using, for example, an interest operator such as the Förstner operator or SIFT (Lowe, 2004).

In our case there are moving objects in the scene that confuse this process; the common features only should be selected from fixed objects. Automatic techniques require a good distribution of features and moreover, classification of moving and fixed objects (Kang et al., 2005), (Pless et al., 2000). Having a good feature distribution is highly dependent on image content.

In this paper, we are aiming at an image registration independent of the image content. In order to find the transformation parameters, we use all the pixels from the image. The transformation that minimizes the difference between the transformed image and the reference image, which is expressed in a so-called energy function, is considered to provide the best transformation. Hereby it is implicitly assumed to have a limited number of moving objects compared to the total number of image pixels.

The rest of the paper is organized as follows: The image registration method with two different transformation models are described in Section 2. The details of searching algorithms used for parameter estimation and scaling parameters are explained in Section 3. In Section 4, we report experimental results on four different dataset. Conclusions and recommendations are given in Section 5

2 IMAGE REGISTRATION

Movement of the helicopter causes movement of camera mounted below it. Therefore reconstructing the second image from the first image is possible by knowing the movement of camera and the distance of an object in a scene to a camera. However, we neglect the effect of relief.

With wrong transformation parameters, the transformed image is not the same as the first one. The inequality is visualized by differences between the first image and the transform image. The Mean Square Error (MSE), is used to express the misalignment between the transformed image and the reference image. The optimized transformation parameters are those that provide the maximum agreement between the transformed image and the reference one.

The equality condition is disturbed by moving objects and by brightness variation.

We assume that the percentage of moving objects is very small relative to the total number of pixels, and that severe local brightness variation is also not existing. Consequently, the transformation parameters are the ones which the difference between the transformed image and the reference image is minimum. As mentioned earlier, the difference is expressed with the MSE. In the other word, the transformation parameters are obtained by minimization of the MSE between the transformed image and the reference image.

The detail of the transformation model comes in the following section. The energy function and parameter space are discussed in the section 2.2.

2.1 Transformation Model

We have tested two transformation models: the projective without shearing and different scale parameters, and the 3D Euclidean model (Hartley and Zisserman, 2004).

This projective model without shearing and different scale parameters is expressed as follows:

$$x_1 = \frac{s\cos(\theta)x_2 + s\sin(\theta)y_2 + t_1}{v_1x_2 + v_2y_2 + 1}$$

$$y_1 = \frac{-s\sin(\theta)x_2 + s\cos(\theta)y_2 + t_1}{v_1x_2 + v_2y_2 + 1}$$

where s, θ, t_1, t_2, v_1, and v_2 are respectively scale, rotation, translational and special projective parameters. x_1 and y_1 are image coordinates of the first image and x_2 and y_2 are for the second image.

The camera motion caused by helicopter is described by the 3D Euclidean transformation model. The camera motion is calculated in the camera coordinate system.

The relation between two camera situations is expressed as follows:

$$\tilde{X}_{cam1} = [R|T]X_{cam2} \tag{1}$$

where R, T $\tilde{X}_{cam1}$, and X_{cam2}, are respectively the 3×3 rotation matrix, the 3×1 translation vector, the camera coordinate for the first situation and the homogenous camera coordinate for the second situation. These parameters are represent by their parametric values as follows:

$$\tilde{X}_{cam} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}, \; X_{cam} = \begin{bmatrix} M \\ N \\ O \\ P \end{bmatrix} \tag{2}$$

$$\Rightarrow \begin{cases} X = \frac{M}{P} \\ Y = \frac{N}{P} \\ Z = \frac{O}{P} \end{cases}, \; R = \begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix}, \; T = \begin{bmatrix} T_X \\ T_Y \\ T_z \end{bmatrix} \tag{3}$$

By replacing them in Equation 1, we have:

$$\begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix} = R \begin{bmatrix} M \\ N \\ O \end{bmatrix}_2 + TP_2$$

The final relation between two camera situations is:

$$\begin{cases} X_1 = R_1 \begin{bmatrix} M \\ N \\ O \end{bmatrix}_2 + T_X P_2 \\ Y_1 = R_2 \begin{bmatrix} M \\ N \\ O \end{bmatrix}_2 + T_Y P_2 \\ Z_1 = R_3 \begin{bmatrix} M \\ N \\ O \end{bmatrix}_2 + T_Z P_2 \end{cases} \tag{4}$$

$$\xrightarrow{\lambda = P_2} \begin{cases} X_1 = \lambda(R_1\tilde{X}_{cam2} + T_X) \\ Y_1 = \lambda(R_2\tilde{X}_{cam2} + T_Y) \\ Z_1 = \lambda(R_3\tilde{X}_{cam2} + T_Z) \end{cases} \tag{5}$$

The data is measured in the image coordinate system. Therefore we need to convert the camera to image coordinates. The relation between two coordinate systems requires internal camera parameters which are calculated in the camera calibration process. We calibrated the camera using a calibration toolbox (Bouguet, 2007) to remove lens distortion and to calculate the calibration matrix, K, for coordinate conversion.

$$K = \begin{bmatrix} f_1 & 0 & p_x \\ 0 & f_2 & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

The conversion relation is formulated as: $\tilde{X}_{cam} = K^{-1} X_{im}$

where $X_{im} = [\ X \quad Y \quad Z\ ]^T_{im}$, $x = \frac{X}{Z}$, $y = \frac{Y}{Z}$. (x, y) and (X, Y, Z) are respectively image coordinate in pixel unit and camera coordinate in metric unit.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}_{cam} = \begin{bmatrix} f_1^{-1}(X_{im} - p_x Z_{im}) \\ f_2^{-1}(Y_{im} - p_y Z_{im}) \\ Z_{im} \end{bmatrix} \tag{6}$$

where $f_1 = \frac{F}{PS_x}$ and $f_2 = \frac{F}{PS_y}$ are focal length in pixel unit. F is focal length thus $Z_{cam} = F$. PS_x and PS_y are respectively pixel size in x and y directions.

By dividing the left side of the equation 6 by Z_{im} the final relation between camera and image coordinates is obtained.

$$\begin{array}{lcl} X_{cam} & = & PS_x(x - px) \\ Y_{cam} & = & PS_y(y - py) \\ Z_{cam} & = & F \end{array} \tag{7}$$

In equation 7, PS_x and PS_y change the pixel unit to metric unit: $\overbrace{PS_x \underbrace{(x - px)}_{pix}}^{m}$.

The relation between two camera situations Equation 4 is reformulated in the image coordinate using Equation 7:

$$\begin{cases} PS_x(x_1 - p_x) = \lambda(R_1 \begin{bmatrix} PS_x(x_2 - p_x) \\ PS_y(y_2 - p_y) \\ F \end{bmatrix} + T_X) \\ PS_y(y_1 - p_y) = \lambda(R_y \begin{bmatrix} PS_x(x_2 - p_x) \\ PS_y(y_2 - p_y) \\ F \end{bmatrix} + T_Y) \\ F_1 = \lambda(R_3 \begin{bmatrix} PS_x(x_2 - p_x) \\ PS_y(y_2 - p_y) \\ F \end{bmatrix} + T_Z) \end{cases} \tag{8}$$

The final relation between two image coordinates is obtained by dividing the first and the second formulas from Equation 8 to the the third one:

$$\begin{cases} x_1 = p_x + f_1 \dfrac{R_1 \begin{bmatrix} f_1^{-1}(x_2 - p_x) \\ f_2^{-1}(y_2 - p_y) \\ 1 \end{bmatrix} + \frac{T_X}{F}}{R_3 \begin{bmatrix} f_1^{-1}(x_2 - p_x) \\ f_2^{-1}(y_2 - p_y) \\ 1 \end{bmatrix} + \frac{T_Z}{F}} \\ y_1 = p_y + f_2 \dfrac{R_2 \begin{bmatrix} f_1^{-1}(x_2 - p_x) \\ f_2^{-1}(y_2 - p_y) \\ 1 \end{bmatrix} + \frac{T_Y}{F}}{R_3 \begin{bmatrix} f_1^{-1}(x_2 - p_x) \\ f_2^{-1}(y_2 - p_y) \\ 1 \end{bmatrix} + \frac{T_Z}{F}} \end{cases} \tag{9}$$

T_X, T_Y and T_Z are in metric unit.

2.2 Energy Function and Parameter Space

In this section, we describe the energy function for an image-content-free registration (no restrictions with regards to image content). In order to find the transformation parameters, we use all image pixels. We have assumed here to have a limited number of moving objects compared to the total number of image pixels.

The energy function selected is:

$$F(p) = \min \sum_x \sum_y |\tilde{I}_{i+1}^{T(p)}(y, x) - \tilde{I}_i(y, x)|^2 / N \tag{10}$$

with p being the vector containing the unknown transformation parameters, six in our case. By minimizing the energy function, the transformation parameters are obtained. Where $\tilde{I}_i$, $\tilde{I}_{i+1}^{T(p)}$ and N are respectively the normalized reference image intensity, the normalized transformed of the target image intensity and the number of pixels in the common area after transformation. Comparing the normalized images removes the global brightness variation effect.

Applying the transformation matrix (T) results in geometrically transformed coordinates of $i + 1$th image, X_{i+1}. With bilinear interpolation, the $i + 1$th image is radiometrically transformed. Mathematically the combined operation of the geometrical and radiometrical transformations of the I_{i+1} is:

$$I_{i+1}^T = I_{i+1}(T(p)_{i+1,i} X_{i+1})$$

Bilinear interpolation changes the brightness values. However the amount of change is limited to the intensity values of 3×3 neighboring pixels.

3 SEARCH ALGORITHM

Searching the whole parameter space for finding the optimum value is computationally very expensive. The complexity of Equation 10 is $O(\prod_{i=1:6} n_{p(i)})$ where the $n_{p(i)}$ with is the number of all possible values for each parameter, $p(i)$. One could imagine the real number space, $\mathbb{R}$, as the range of each parameter. However, not every combination of parameters is allowed.

Each parameter has a certain range beyond which the transformed image is meaningless. Moreover, for each parameter there is a sensitivity value such that within the sensitivity value the transformed images are equal. Although calculating range and sensitivity of parameters reduces the searching space, it is still huge.

Therefore, we have applied a global optimization technique. Here Differential Evolution (DE) (Price et al., 2005) is used to find the global optimum.

DE starts with an initial population of q randomly (McKay et al., 1979) chosen parameter value combinations $\mathbf{m}$. These $\mathbf{m}$'s are improved during successive generations of constant size q, in the sense that a descendant replaces an $\mathbf{m}$, becoming its successor, if it has a lower energy. The distinctive feature of DE is the way in which these descendants are created. Various ways to generate new $\mathbf{m}$'s exist, but here only the following procedure is considered. At the start of generation k the parameter vectors $\mathbf{m}_{k,1}, \cdots, \mathbf{m}_{k,q}$ are given and for each of them a descendant is created. To create a descendant $\mathbf{d}_{k,i}$, a partner $\mathbf{p}_{k,i}$ is constructed as follows:

$$\mathbf{P}_{k,i} = \mathbf{m}_{k,j_1} + F(\mathbf{m}_{k,j_2} - \mathbf{m}_{k,j_3}) \tag{11}$$

With the three different $\mathbf{m}$-vectors chosen at random from the population and F being a scalar multiplication factor between 0 and 1. The descendant $\mathbf{d}_{k,i}$ of $\mathbf{m}_{k,i}$ results from applying crossover to $\mathbf{m}_{k,i}$ and $\mathbf{p}_{k,i}$ with crossover probability pc. A higher value of pc leads (on the average) to more dimensions of $\mathbf{p}_{k,i}$ being copied into $\mathbf{m}_{k,i}$. Descendant $\mathbf{d}_{k,i}$ only replaces $\mathbf{m}_{k,i}$, becoming its successor, if its energy is lower. The setting parameters of DE are population size q, multiplication factor F, crossover probability pc and the number of generations NG. The values chosen for the setting parameters are chosen according to (Snellen and Simons, 2007).

The result of DE is not precise. We used a local optimizer to converge to a more precise result. The Nelder-Mead downhill simplex algorithm (NM) (Lagarias et al., 1998) is used as an local optimizer. It does not require the calculation of derivatives.

Often, the result obtained from DE is good enough to use it as the final result without using complementary NM method.

The range of the parameters is calculated for initializing DE method. The result of DE is used as an initial value for NM method. To speed up the process, we calculate the result in the scale space. In the first step parameters are obtained by using DE in the lowest image scale. Later on these parameters are used as an initial values in NM method in the same image scale. The process continues for the higher image scale till the main image scale using the scaled result of previous stage as for the initialization of NM method. The discussion about which parameter should be scaled and amount of scaling is presented in Section 3.1.

3.1 Scale Space

Each candidate of parameter combination is evaluated by an energy function, which requires image transformation. The image transforming for full size images is time consuming. The transformation and therefore the whole process is faster if we calculated the parameters in a scaled image. The parameters also need to be scaled. In each transformation model, parameters are scaled according to scaling the image coordinates in such a way to compensate the coordinate scaling.

In projective equation is the lowest scale, n:

$$x_1/2^{n-1} = \frac{s\cos(\theta)x_2/2^{n-1} + s\sin(\theta)y_2/2^{n-1} + t_1/2^{n-1}}{v_1 * 2^{n-1}x_2/2^{n-1} + v_2 * 2^{n-1}y_2/2^{n-1} + 1}$$

$$y_1/2^{n-1} = \frac{-s\sin(\theta)x_2/2^{n-1} + s\cos(\theta)y_2/2^{n-1} + t_1/2^{n-1}}{v_1 * 2^{n-1}x_2/2^{n-1} + v_2 * 2^{n-1}y_2/2^{n-1} + 1}$$

t_1, t_2 are scaled down, v_1, v_2 are scaled up to 2^{n-1}. The rest of the parameters stays intact.

In the same way for 3D Euclidean model (Equations 9), K and the transformation parameters are scaled down to 2^{n-1}. The rotational parameters remaining the same.

We calculate the parameters using DE in the lowest scale. The results are then used as an initial values in NM method in the lowest scale. These parameters are scaled for the higher scale and used as an initial value to initiating NM. The final result is the one which is calculated for the main image scale.

4 RESULTS AND DISCUSSIONS

We used three levels of the image scale using gaussian filter with 3×3 windows, with one for mean and 0.5 for standard deviation. In the third scale (smallest scale), DE is calculating the first results using 100 number of generations, 16 population numbers, crossover rate 0.55 and multiplication factor 0.6. The obtained result is initializing the NM for more precise result. This result is scaled for next NM calculation in higher scale till the main scale. Only 50 times iteration for the third and the second levels and 10 for the main image scale is enough to get the final result. With Matlab implementation, we got about 6 minutes for DE and 4 minutes for NM.

Figure is illustrated a minimum of population and its energy in each generation. We run the program for two different runs which is demonstrated by a black and red color. The results are stabilized after about 100 frames. The correlation between parameters are displayed in the right figure of Figure . The relation between energy value and each parameters in the whole generation and population are demonstrated in Figure 2. Figure 3 is the energy values for two parameters together. The energy values are represented by colors. The blue color shows that the population has become more stable. In the above-mentioned figures, we used Apeldoorn images which will be visualized later and the 3D Euclidean model.

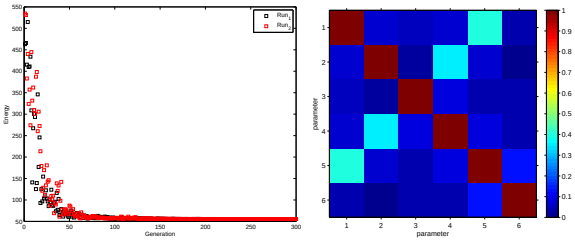

Figure 1: left: Minimum energy value of population is visualized in each generation. right: Correlation between the parameters

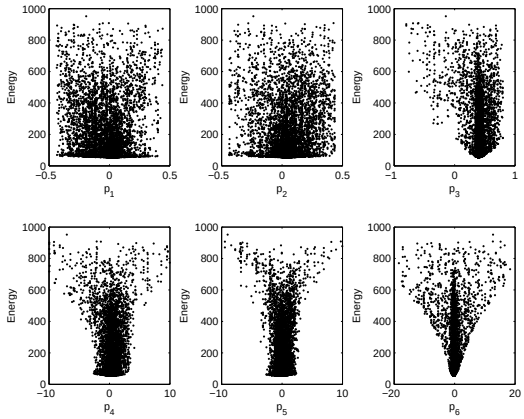

Figure 2: parameter value and its energy

The results are tested for four different image sequences: Klaver, Apedoorn, lokseq, and Ouderijn. Klaver has less structural information around the road. In contrast, Apeldoorn has more complicated structure surrounding the road. Lokseq and Ouderijn are selected as an opposite examples of simple and sophisticated road shape. The result of our method is compared with corresponding points that are identified manually. The parameters are estimated using Levenberg-Marquardt algorithm (LM) for the following energy function:

$$F(p) = \min \sum (X_1 - T(p)X_2)^2 \quad (12)$$

where X_1, X_2, and T are respectively first, second image coordinates and transformation matrix. X_1 and X_2 for projective model should be transformed to the image center.

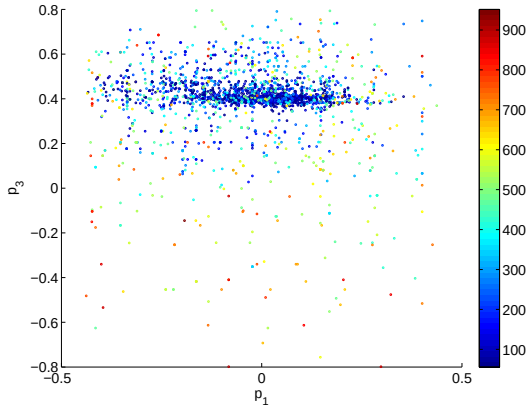

Figure 3: Two parameter values and their energy

The results are given in Tables 1, 2, 3, and 4 respectively for Klaver, Apeldoorn, lokseq, and Ouderijn images. The first row of each transformation model is assigned to our method and the second one for the manual method. The $1-6^{th}$ column shows the parameters. 7^{th} and 8^{th} are the energy value and amount of moving pixels smaller than 5 pixels to the whole number of image pixels of the results.

In 3D Euclidean model, first three parameters are rotational and the last are translational parameters. In our projective model, the parameters are ordered as scale, rotation, translation, and two projective parameters.

The values in the table are rounded to the precision of each parameter. We got better results in our method than in the manual way in all the images. Lower quality of manual results is due to inaccuracy in corresponding point selection because of slight image burring. Finding corresponding points are more difficult in Klaver and Apeldoorn images.

There is a small correlation between rotation around x axis and translation in y direction and also for rotation around y axis and translation in x direction (see right Figure 1) in 3D Euclidean model which results in having different results for these parameters in a approximate calculation. But this problem is not occurring in accurate calculation. That is why the parameters from manual and automatic methods in 3D Euclidean model are a little bit different in the $2-5^{th}$ parameters.

Figures 4, 5, 6, and 7 are representing the potential of our method in image registration. The first row, from left to right, is respectively the first, second image and the difference between them. The second row is the first and the transformed of the second image with the results from our method using 3D Euclidean model and their difference.

5 CONCLUSIONS

We have proposed an approach for automatic registration of consecutive images in an image sequence without using any feature. The parameters have been estimated by optimizing MSE between a reference image and a transformed target image. Our method has shown robustness in different image sequences with various contents. The images are precisely co-registered. This method will be extended to registration of an entire image sequence. Although in principle, this approach is able to tackle the large transformation, but more subtle idea is required to reduce computation cost. We will further study the possibility of using different energy functions such as correlation coefficient or more expensive ones such as mutual information, and investigate their behavior and limitation. In addition, the further study will be on possibility of using randomly selected image pixels instead of entire image pixels to speeding up the process.

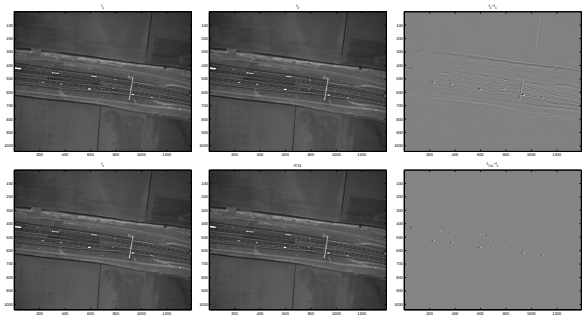

Figure 4: Klaver images: The first row, from left to right, is the first, second and their difference images. The second row, from left to right, is the first image and the transformed of the second image based on the estimated parameters using our method and their difference.

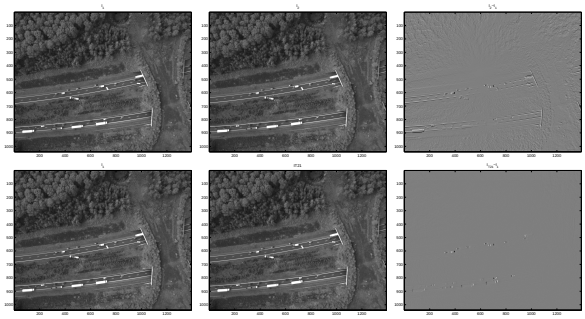

Figure 5: Apeldoorn images: The first row, from left to right, is the first, second and their difference images. The second row, from left to right, is the first image and the transformed of the second image based on the estimated parameters using our method and their difference.

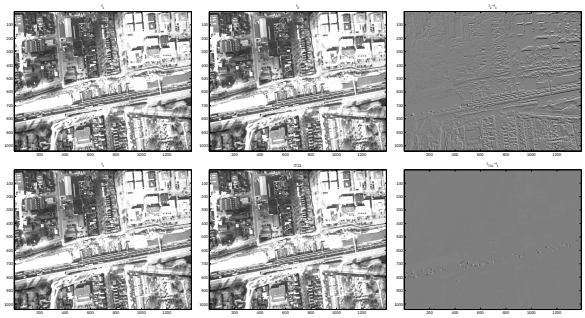

Figure 6: lokseq images: The first row, from left to right, is the first, second and their difference images. The second row, from left to right, is the first image and the transformed of the second image based on the estimated parameters using our method and their difference.

ACKNOWLEDGEMENTS

The research presented in this paper is part of the research program "Tracing Congestion Dynamics with Innovative Traffic Data to a better Theory", sponsored by the Dutch Foundation of Scientific Research MaGW-NWO.

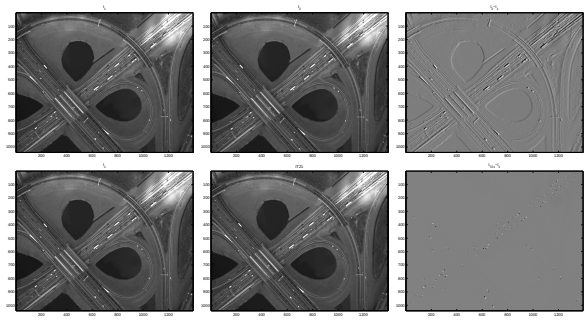

Figure 7: Ouderijn images: The first row, from left to right, is the first, second and their difference images. The second row, from left to right, is the first image and the transformed of the second image based on the estimated parameters using our method and their difference.

Table 1: Klaver: The first row of each model is our method and the second row is the manual method. The 1-6th columns are parameters. The 7th is the energy value and the last is percentage of moving pixels less than 5 pixels.

3D Euclidean parameters							
p_1 10^{-1}	p_2 10^{-1}	p_3 10^{-1}	p_4	p_5	p_6	e	m_5
0.13	-1.63	1.64	-5.9	-3.3	-0.1	19.6	0.981
0.01	1.02	1.83	0.5	-3.3	-0.0	24.6	0.967
Projective without affine parameters							
p_1	p_2 10^{-1}	p_3	p_4	p_5 10^{-6}	p_6 10^{-6}	e	m_5
1.000	-1.72	-2.3	-3.0	-2.04	0.61	19.6	0.981
1.000	-1.81	-1.3	-3.3	1.36	0.05	24.6	0.967

Table 2: Apeldoorn: The first row of each model is our method and the second row is the manual method. The 1-6th columns are parameters. The 7th is the energy value and the last is percentage of moving pixels less than 5 pixels.

3D Euclidean parameters							
p_1 10^{-1}	p_2 10^{-1}	p_3 10^{-1}	p_4	p_5	p_6	e	m_5
0.27	0.53	4.05	1.9	-1.4	-0.9	23.5	0.975
-1.04	-0.05	4.18	0.7	1.5	-0.4	26.8	0.972
Projective without affine parameters							
p_1	p_2 10^{-1}	p_3	p_4	p_5 10^{-6}	p_6 10^{-6}	e	m_5
1.001	-4.03	-2.3	-3.0	0.38	-0.31	23.3	0.979
1.000	-4.18	-1.3	-3.3	-0.11	1.41	26.8	0.972

3D Euclidean parameters							
p_1 10^{-1}	p_2 10^{-1}	p_3 10^{-1}	p_4	p_5	p_6	e	m_5
-1.49	-0.16	2.10	0.3	0.7	-0.0	27.8	0.933
-1.15	0.28	2.05	1.4	-0.3	-0.2	35.8	0.874
Projective without affine parameters							
p_1	p_2 10^{-1}	p_3	p_4	p_5 10^{-6}	p_6 10^{-6}	e	m_5
1.000	-2.11	-2.3	-3.0	-0.31	1.67	27.5	0.936
9.999	-2.05	-1.3	-3.3	0.37	1.56	36.0	0.872

Table 3: lokseq: The first row of each model is our method and the second row is the manual method. The 1-6th columns are parameters. The 7th is the energy value and the last is percentage of moving pixels less than 5 pixels.

3D Euclidean parameters							
p_1 10^{-1}	p_2 10^{-1}	p_3 10^{-1}	p_4	p_5	p_6	e	m_5
-0.68	3.82	0.27	-0.7	-0.7	0.5	34.1	0.957
0.34	2.90	0.37	-2.7	-3.4	-0.5	48.1	0.895
Projective without affine parameters							
p_1	p_2 10^{-1}	p_3	p_4	p_5 10^{-6}	p_6 10^{-6}	e	m_5
1.000	-1.7	-0.20	-3.0	4.42	-0.94	36.1	0.940
1.000	-1.9	-0.33	-3.3	3.94	-0.46	48.0	0.896

Table 4: Ouderijn: The first row of each model is our method and the second row is the manual method. The 1-6th columns are parameters. The 7th is the energy value and the last is percentage of moving pixels less than 5 pixels.

References

Bouguet, J.-Y., 2007. Camera calibration toolbox for matlab. http://www.vision.caltech.edu/bouguetj/calib_doc/.

Hartley, R. I. and Zisserman, A., 2004. Multiple View Geometry in Computer Vision. Second edn, Cambridge University Press.

Kang, J., Cohen, I., Medioni and Chang Yuan, G., 2005. Detection and tracking of moving objects from a moving platform in presence of strong parallax. In: IEEE International Conference on Computer Vision, Vol. 1, pp. 10–17. Link.

Lagarias, J. C., Reeds, J. A., Wright, M. H. and Wright, P. E., 1998. Convergence properties of the nelder-mead simplex method in low dimensions. In: SIAM Journal of Optimization, Vol. 9number 1, pp. 112–147.

Lowe, D. G., 2004. Distinctive image features from scale-invariant keypoints. In: International Journal of Computer Vision.

McKay, M. D., Conover, W. J. and Beckman, R. J., 1979. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. In: Technometrics, Vol. 21, pp. 239–245.

Pless, R., Brodsky, T. and Aloimonos, Y., 2000. Detecting independent motion: The statistics of temporal continuity. In: IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22number 8, pp. 768–773. Link.

Price, K. V., Storn, R. M. and Lampinen, J. A., 2005. Differential Evolution: A Practical Approach to Global Optimization. First edn, Springer.

Snellen, M. and Simons, D. G., 2007. An assessment of the performance of global optimisation methods for geoacoustic inversion. In: accepted for publication in the Journal of Computational Acoustics.