20

Camera Motion Estimation Based on Edge Structure Analysis

Andrey Vavilin and Kang-Hyun Jo
University of Ulsan,
Korea

1. Introduction

The estimation of camera motion is important for several video analysis tasks such as indexing and retrieval purposes, motion compensation and for scientific film analysis. From an aesthetical point of view, camera motion is often used as an expressive element in film production. Motion content can be used as a powerful cue for structuring video data, similarity-based video retrieval, and video abstraction

Motion estimation and motion pattern classification problem have been extensively investigated by the scientific community for semantic characterization and discrimination of video streams. Moving object trajectories have been used for video retrieval [1–3]. Camera motion pattern characterization has been efficiently applied to video indexing and retrieval [4–7]. However, the main limitation of the latter methods is that they deal only with the characterization of the detected camera motion patterns, without explicit measurement of the camera motion parameters. As a result, the acquired information is of limited interest, since it can be used primarily for video indexing and retrieval.

There are different types of camera motion: rotation around one of the three axes and translation along the x and y-axis. Furthermore, zoom in and out can be considered as equivalent to translation along the z-axis. Existing methods can be classified as optical flow methods and feature correspondences based approaches. Let us also mention recursive techniques based on extended Kalman filters [8] which track camera motion and estimate the structure of the scene. In the case of an uncalibrated camera, interesting approaches are described in [9, 10]. The use of optical flow avoids the choice of “good features”. In [11] differential approaches of the epipolar constraint are described. In [12], the optical flow computed between two adjacent images in a video sequence is linearly decomposed on a database of optical flow models. The authors of [13] propose a comparison of algorithms which only use optical flow for estimating camera.

In this work we present an approach for recovering graph-based structures from images. This structure is then used for estimating 3D camera motion. Our approach is based on detecting straight line segments. After several prefiltering operations such as bilateral filtering and Hough transform, preceding by the edge detection is applied. Then a result of transformation analyses in order to detect local maximums. Reverse transform of these maximums gives us a several straight lines presented in the image. We use an intersection of these analytical lines with detected edges in order to find straight line segments. On this step we also detect intersections between line segments. For each intersection point we compute rank based on number of connected points. After transforming image into graph we search for similar structural elements in the graph of previous frame. This process is based on searching subgraphs consists of vertices with similar ranks. After finding correspondence points camera motion is estimated as a combination of translation and rotation.

2. Camera motion model

In this work we considered eight-parameter perspective model defined as follows:

$$\begin{aligned} x_i^2 &= \frac{a_0 + a_2 x_i^1 + a_3 y_i^1}{a_6 x_i^1 + a_7 y_i^1 + 1} \\ y_i^2 &= \frac{a_1 + a_4 x_i^1 + a_5 y_i^1}{a_6 x_i^1 + a_7 y_i^1 + 1} \end{aligned} \tag{1}$$

where $\left(x_1^1, y_1^1\right)$ and $\left(x_i^2, y_i^2\right)$ are the coordinates of the same point in the consequent frames at t_1 and t_2 respectively and $\left(a_0, \ldots, a_7\right)$ are the motion parameters. Various motion models can be derived from this mode. For example, in case of $a_6 = a_7 = 0$ it is reduced to affine model, and setting $a_2 = a_5$, $a_3 = -a_4$ and $a_6 = a_7 = 0$ will give us a translation-zoom-rotation model.

In [14] any vector field is approximated by a linear combination of a divergent field, a rotation field and two hyperbolic fields. The relationship between motion model parameters and symbol-level interpretation is established as:

$$\begin{aligned} &Pan = a_0 \\ &Tilt = a_1 \\ &Zoom = \frac{1}{2}\left(a_2 + a_5\right) \\ &Rotation = \frac{1}{2}\left(a_3 - a_4\right) \end{aligned} \tag{4}$$

Error in esti mation parameters is defined as:

$$\varepsilon(a) = \sum_{i=1}^{N} \left\| p_i^2 - f\left(p_i^1, a\right) \right\| \tag{2}$$

where N is the number of corresponding points, $p_i^1 = \left(x_i^1, y_i^1\right)$ and $p_i^2 = \left(x_i^2, y_i^2\right)$ are the corresponding points in first and second frames, $a = \left(a_0, \ldots, a_7\right)$ is a transformation parameters vector and *f()* is a transformation function defined by (1).

Using this model definition, problem of camera motion estimation could be formalized as the error minimization problem:

$$M = \arg\min_{a} \varepsilon\left(a\right) \tag{3}$$

Where *M* defines estimated camera motion parameters.

It is well known that, by taking some particular point *p* as the origin of the coordinate system with coordinates *z*, any infinitely differentiable function *f*(x) could be approximated using Taylor series:

$$f\left(z\right) = f\left(p\right) + \sum_{i} \frac{\partial f}{\partial z_i} z_i + \frac{1}{2} + \sum_{i} \frac{\partial^2 f}{\partial z_i \partial z_j} z_i z_j + \ldots \approx c - b \cdot z + \frac{1}{2} z \cdot A \cdot z \tag{4}$$

where

$$c \equiv f\left(p\right) \qquad b \equiv -\nabla f \mid p \quad \left[A\right]_{ij} = \frac{\partial^2 f}{\partial z_i \partial z_j} \mid p \tag{5}$$

The matrix *A* which consists of a second partial derivatives of the function is also called Hessian matrix of the function at *p* [15]

In approximation of (4) the gradient of *f* is easily calculated as

$$\nabla f\left(z\right) = Az - b \tag{6}$$

In Newton’s method gradient is set to 0 to determine the next iteration point.

The gradient of error function ε with respect to parameters *a* has components

$$\frac{\partial \varepsilon}{\partial a_k} = -2\sum_{i=1}^{N} \frac{\left[p_i^2 - f\left(p_i^1, a\right)\right] \partial f\left(p_i^1, a\right)}{\partial a_k}, \qquad k = 0,1,\ldots,7 \tag{7}$$

Taking second order partial derivatives gives

$$\frac{\partial^2 \varepsilon^2}{\partial a_k \partial a_j} = 2\sum_{i=1}^{N} \left[\frac{\partial f\left(p_i^1, a\right)}{\partial a_k} \frac{\partial f\left(p_i^1, a\right)}{\partial a_j} - \left[p_i^2 - f\left(p_i^1, a\right)\right] \frac{\partial^2 f\left(p_i^1, a\right)}{\partial a_k \partial a_j} \right] \tag{8}$$

It is conventional to remove the factors of 2 by defining

$$\alpha_{kj} = \frac{1}{2} \frac{\partial^2 \varepsilon^2}{\partial a_k \partial a_j} \tag{9}$$

$$\beta_k = \frac{1}{2} \frac{\partial \varepsilon}{\partial a_k} \tag{10}$$

Making [a]=1/2A in equation (6), in terms of which that equation can be rewritten as the set of linear equations

$$\sum_{i=1}^{N} \alpha_{ki} \partial a_i = \beta_k \tag{11}$$

SVD is used to compute transformation parameters form overdetermined set of linear equations (11).

In the proposed work initial translation was estimated prior to pan-tilt-zoom estimation based on center of gravity of corresponding feature points in consistent frames. To remove outliers voting idea was used. After finding correspondences between frames each pair of matching points "votes" for its offset. Then points with small number of offset votes are discarded.

3. Image to graph conversion

However, the most challenging part in camera motion estimation is finding correspondences between two frames. In proposed work this process is based on searching similar edge structures in the frames. Input image is represented as a graph based on edges and their intersections. Intersection points are ranked according to the number of connections. Algorithm for transforming image into graph is presented in Fig.1.

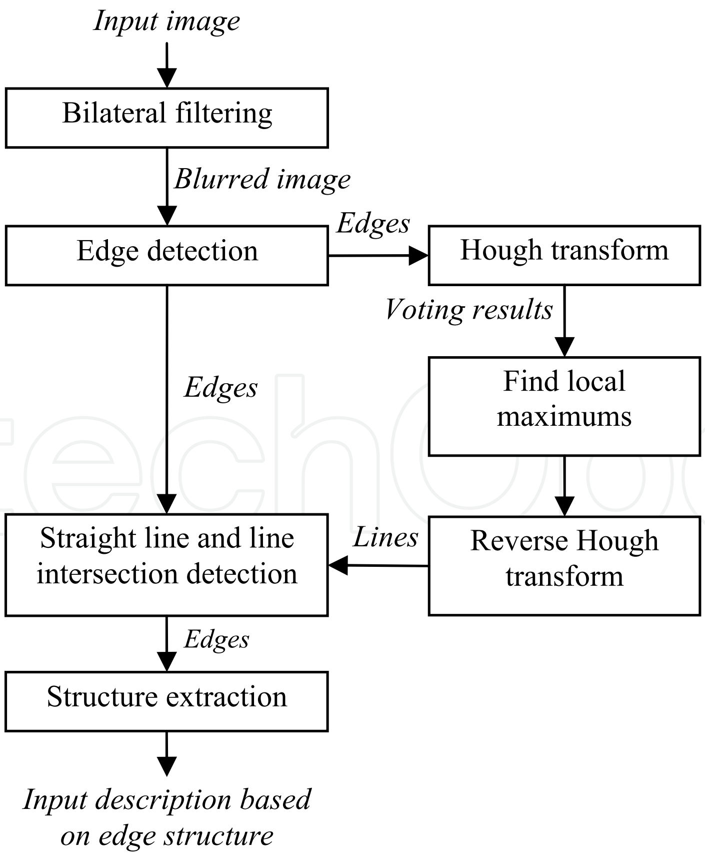


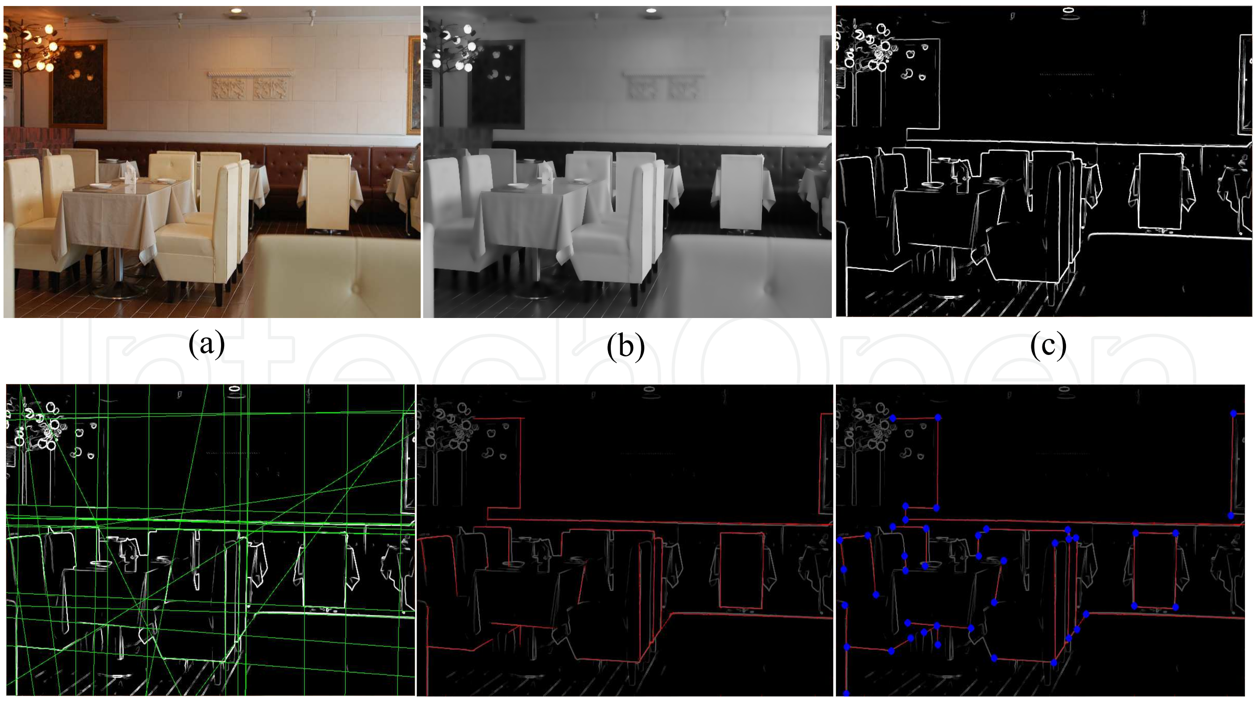
Fig. 1. Transforming image into graph

The proposed method is based on matching edge structures inside the images. Robustness of the algorithm depends on the quality of detected edges. Thus, we try to discard small edges and edges with small magnitude before recovering structures. One of the most effective way to remove such edges is a bilateral filtering [18]. We used bilateral filtering to blur image while preserving strong edges (Fig.2(b)). This allows us to reduce number of small edges. With a Gaussian function $g(x,\sigma)=\exp\left(-x^2/\sigma^2\right)$, bilateral filter of input image I at pixel p is defined as

$$bf(I)_p = \frac{1}{k}\sum_{q\in I}\left[G\left(\|p-q\|,\sigma_s\right)\cdot G\left(\left|I(p)-I(q),\sigma_I\right|\right)\cdot I(q)\right] \tag{5}$$

where σ_s controls the influence of spatial neighbourhood, σ_I the influence of the intensity difference and k is a normalizing coefficient defined as

$$bf(I)_p = \frac{1}{k}\sum_{q\in I}\left[G\left(\|p-q\|,\sigma_s\right)\cdot G\left(\left|I(p)-I(q),\sigma_I\right|\right)\right] \tag{6}$$


(a) (b) (c)

(d) (e) (f)

Fig. 2. Example of image to graph transformation. Refer text for details.

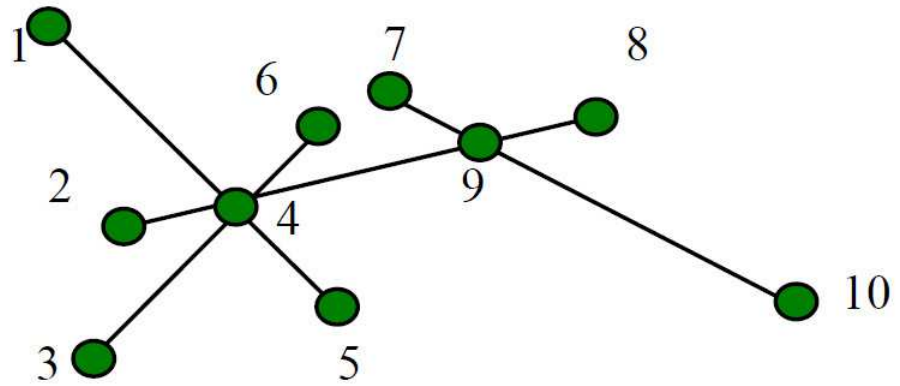


Fig. 3. Example of graph structure.

Next step is edge detection (Fig.2(c)). In this work Sobel edge detector was used. Edges with magnitude less than 50 were filtered out during this step. Remained edges were mapped into ρ,φ space using Hough transform. As a result 2-dimensional voting array was generated. Local maximums of this array correspond to strongest edges in the image. Using reverse Hough transform for these maximums we obtain set of lines (Fig.2(d)). Intersection of these lines gives us straight line segments (Fig.2(e)) and their intersections (Fig.2(f)). The purpose of these steps is to obtained graph-like structure. In further processing we use information about edge intersections (vertices) and their connections (edges). The computational time of the proposed algorithm might be reduced by removing Hough transform. It is not very important to detect exactly straight lines between vertices. The important information is their connectivity. However, using straight lines instead of edges allows more robust detection of edge intersection.

Information about edges could be represented in the form of a graph $<E,V>$ where $v=\{v_1,...,v_m\}$ is a set of vertices obtained from intersections and $E=\{e_1,...,e_n\}$ is a set of edges. Each vertex is described by its coordinates and rank which shows how many connections this vertex has. Edges of graph show connections between vertices. To compute rank of the vertex we used the following idea. Let's consider the structure presented in the Fig.3 as a graph, obtained after edge detection. It has nine edges $e_1 \ldots e_9$, ten vertices $v_1 \ldots v_{10}$ and could be described by the following connectivity matrix :

$$C=\begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Let's define *j*-th order rank of the *i*-th vertex as a number of vertices which could be reached from *i*-th vertex by *j* steps. First order rank of the *i*-th vertex shows how many vertices are directly connected to it. Thus, the v_1 will have first order rank $r_1^1 = 1$ and the v_4 will have $r_4^1 = 6$. Second order rank shows how many vertices could be reached from the *i*-th vertex in two steps. Thus, $r_1^2 = 6$ and $r_4^2 = 9$. Finally, *n*-th order rank shows how many vertices could be reached from *i*-th vertex in *n* steps. This ranking could be used to evaluate complexity and size of the substructures. Rank table for the graph presented in Fig.3 is shown in Table 1. Let's define the highest order of the rank *rMax* as the minimal order which satisfies the following condition:

$$rMax = j \mid r_i^j = n \forall i \geq rMax \tag{7}$$

where *n* is the number of edges in graph.

The highest order for this graph is 3. Using higher order rank is useless for that graph as long as it would have same values. However, in common case, using higher order ranks allows to make matching process more effective by detecting more complex structures inside the image.

	Vertices									
	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}
r_i^1	1	1	1	6	1	1	1	1	4	1
r_i^2	6	6	6	9	6	6	4	4	9	4
r_i^3	9	9	9	9	9	9	9	9	9	9

Table 1. Vertex ranks

These ranks have the following properties:

- Maximum rank could not be higher than number of edges.
- Maximum rank is equal to the number of edges if and only if the graph does not contain loops.
- Highest order of the rank shows diameter of the graph: maximum number of steps required to reach any vertex from any other.

After transforming two frames into graphs problem of finding correspondences between them could be reformulated as a problem of finding sub-graphs with similar structure. We start our search with finding correspondences for vertices with high rank. During this process spatial information could also be considered. Thus we try to find corresponding vertex which would have the same rank and will be located close to the reference vertex. In the future work we this step may be improved by searching correspondences for groups of vertices instead of matching them one by one. As the result of this step we obtain two set of points *P* and *Q* which are used to estimate camera motion as it was described.

Matching idea is based on searching similar substructures in the image graphs. Typically, images contain many simple substructures. These substructures have a small maximum rank order *rMax*. Furthermore, most of the vertices have small rank of first and second order, while very few of them are ranked higher than 6. Thus we start searching of correspondences from the structures with highest ranks. In the graph presented on Fig.3 such vertices are v_4 and v_9. They have ranks 6 and 4 respectively. As long as they are connected we will try to search for two connected vertices with ranks 4 and 6 in the next frame.

To simplify matching process we try to find substructures with more complex structure. This problem is solved by computing rank of vertices of higher orders.

Some edges could be lost in image sequence due to noise, camera movement and movement of objects in scene. It may cause differences in the structure of image graphs. To solve this problem the following ideas were used:

1. We tried to match structures located closer to the center of frame first in order to avoid loosing edges due to camera motion.
2. We consider that vertex i in one frame may correspond to vertex j in other frame even if their ranks are not exactly same. Thus, for each vertex in first frame we select several candidates which ranks are differs less than 30%. After candidates are selected for all vertices we try to choose best corresponding pairs using spatial information.

To decrease number of incorrect matches and to increase the preciseness of matching algorithm matched points were additionally compared using Cross-Hexagon Search (CHS) algorithm [19]. Coordinates of matched graph vertices were used a block centers and offset between matched points in consistent frames was used as initial offset. As long as CHS is used for verification of matched feature points, it can be changed to any block-based feature matching algorithm such as Diamond Search [20] or Three-Step Search [21].

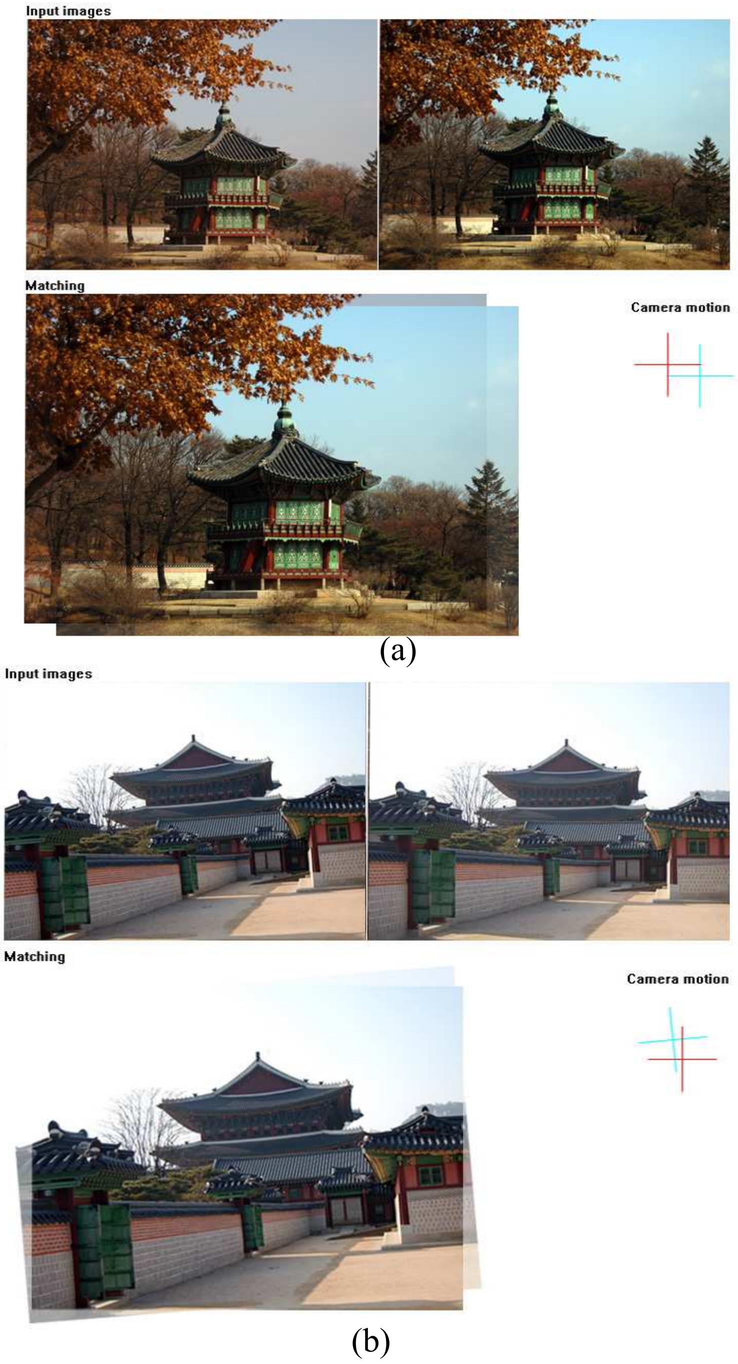
4. Experimental results

All experiments were done on Intel Core 2 Duo with 2Gb memory under Borland Builder environment. Program was not optimized for the maximum performance, thus, computational time, shown in Table 2, could be decreased.

Stage	Average time (sec)
Bilateral filtering	0,31
Edge detection and Hough transform	0,2
Representing image as a graph	0,03
Ranking	Less than 0,01
Graph matching	0,17
Total	0,72

Table 2. Average computational time for 200 vertices.

To evaluate matching quality several types of experiments were done. First group of tests considered planar camera motion (translation and rotation around camera optical axis). Image sequences with camera smoothly by shifted by 20 cm in different directions and rotated by 20 degrees around its optical axis were made (see Fig.4 for example). To evaluate quality of camera motion estimation average and maximum absolute errors were computed. Results for all groups of tests are shown in Table 3.



(a)

(b)

Fig. 4. Planar camera motion estimation for translation (a) and translation with rotation (b).

Second group of tests was used to evaluate algorithm performance for full 3D camera motion with known real camera trajectory. Three kinds of scenes were used: static scenes, scenes with moving objects and scenes with high amount of natural objects (trees, grass etc). Example of camera trajectory estimation is shown at Fig.5~6.

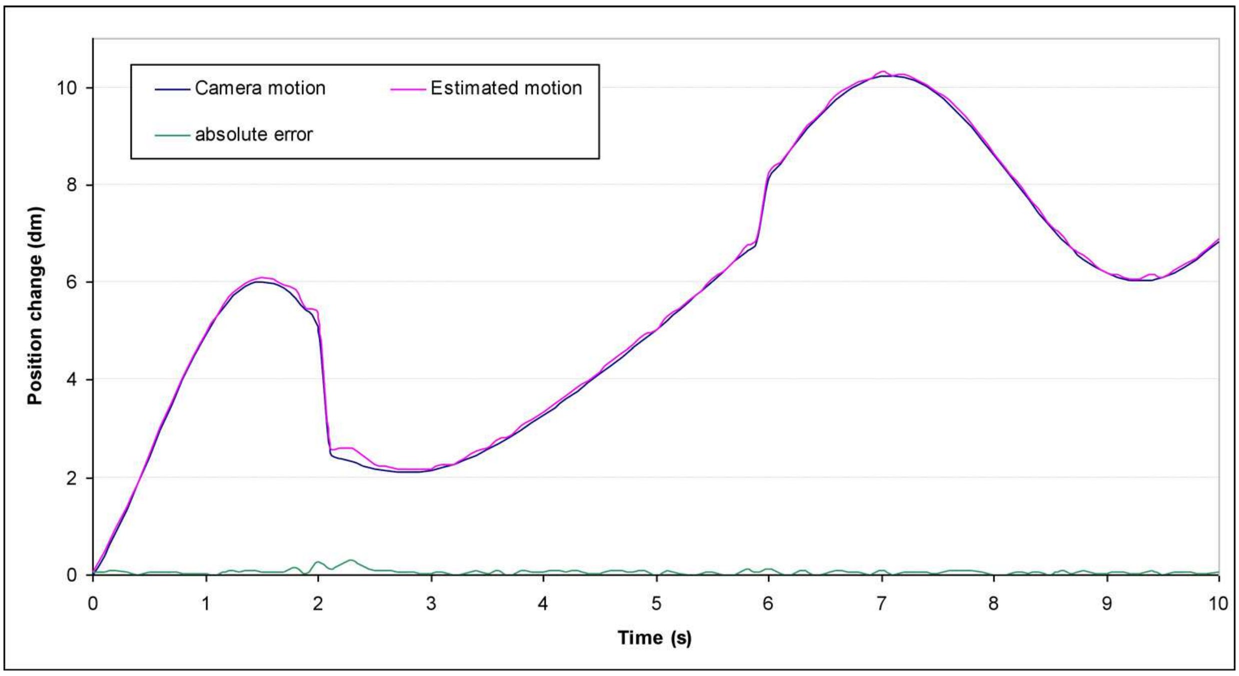


Fig. 5. Estimated trajectory for translation and zoom.

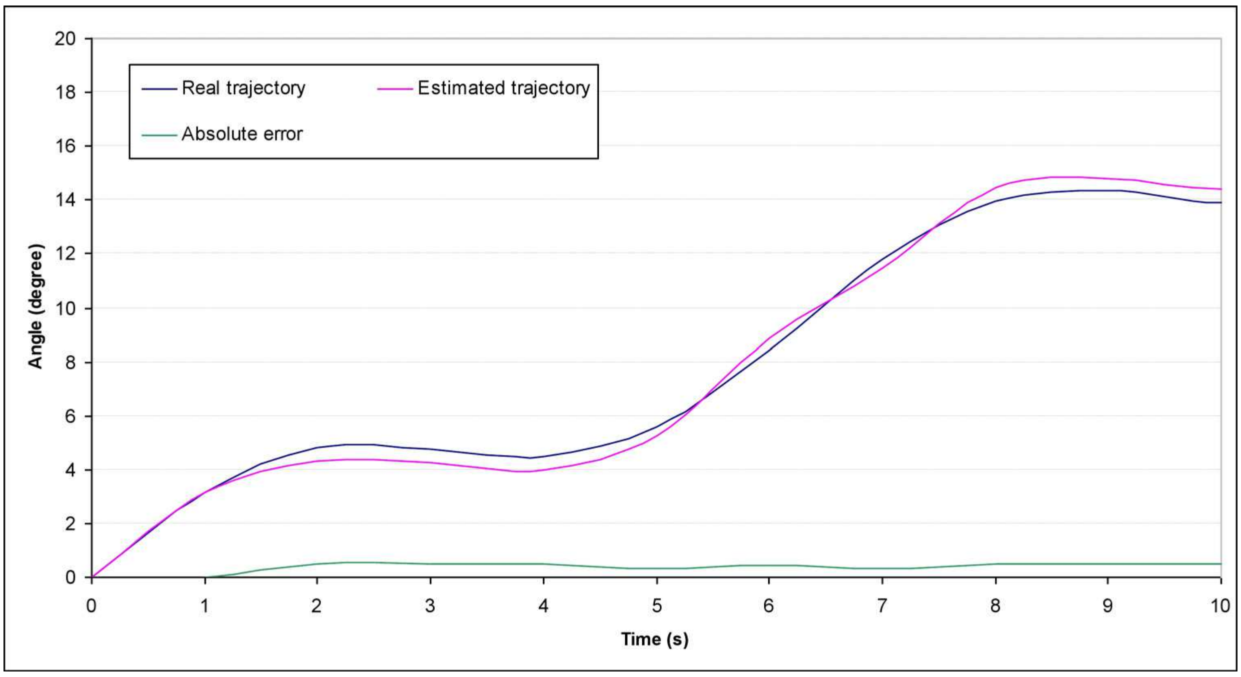


Fig. 6. Estimated trajectory for rotation.

Additionally, these tests were used to evaluate performance of algorithm with and without CHS correction. (Fig.7,8). It is easy to see, that using additional verification step for correspondent points could decrease estimation error, with relatively small increasing of computational time (about 0,07 sec).

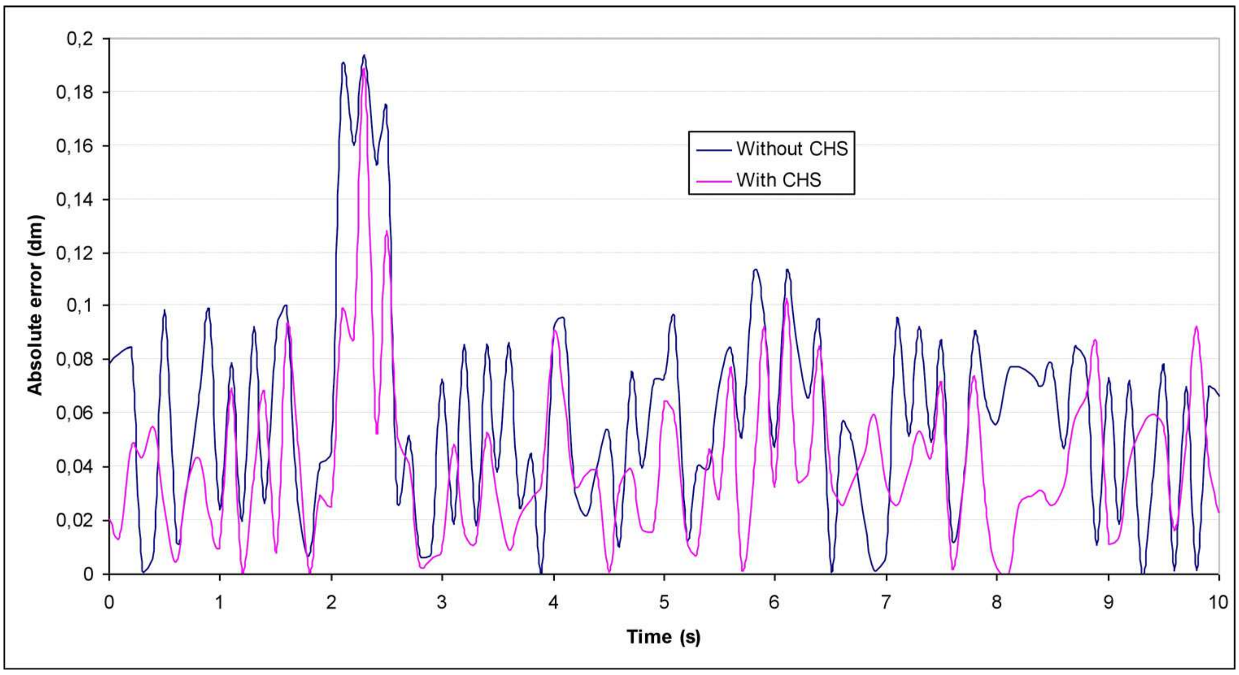


Fig. 7. Absolute error for translation and zoom.

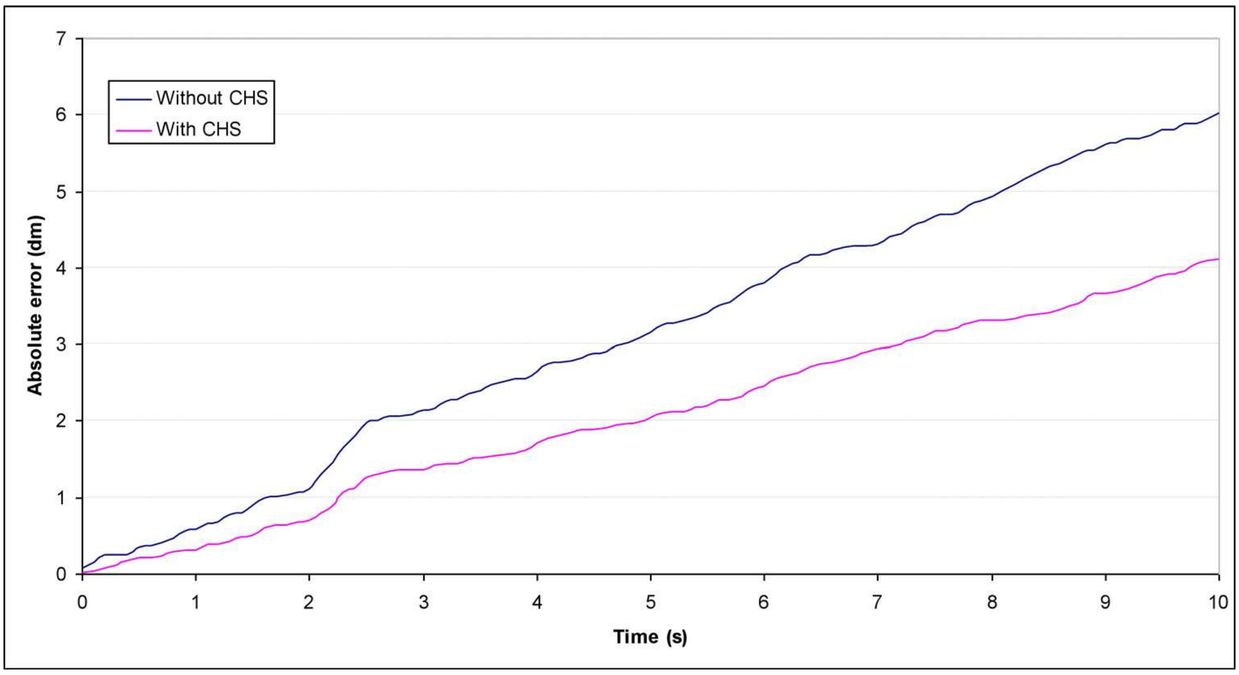


Fig. 8. Accumulated absolute error for translation and zoom.

In last group of tests image sequence with predefined camera motion trajectory was used to evaluate error depending on number of graph vertices used for matching result is shown in Fig.9.

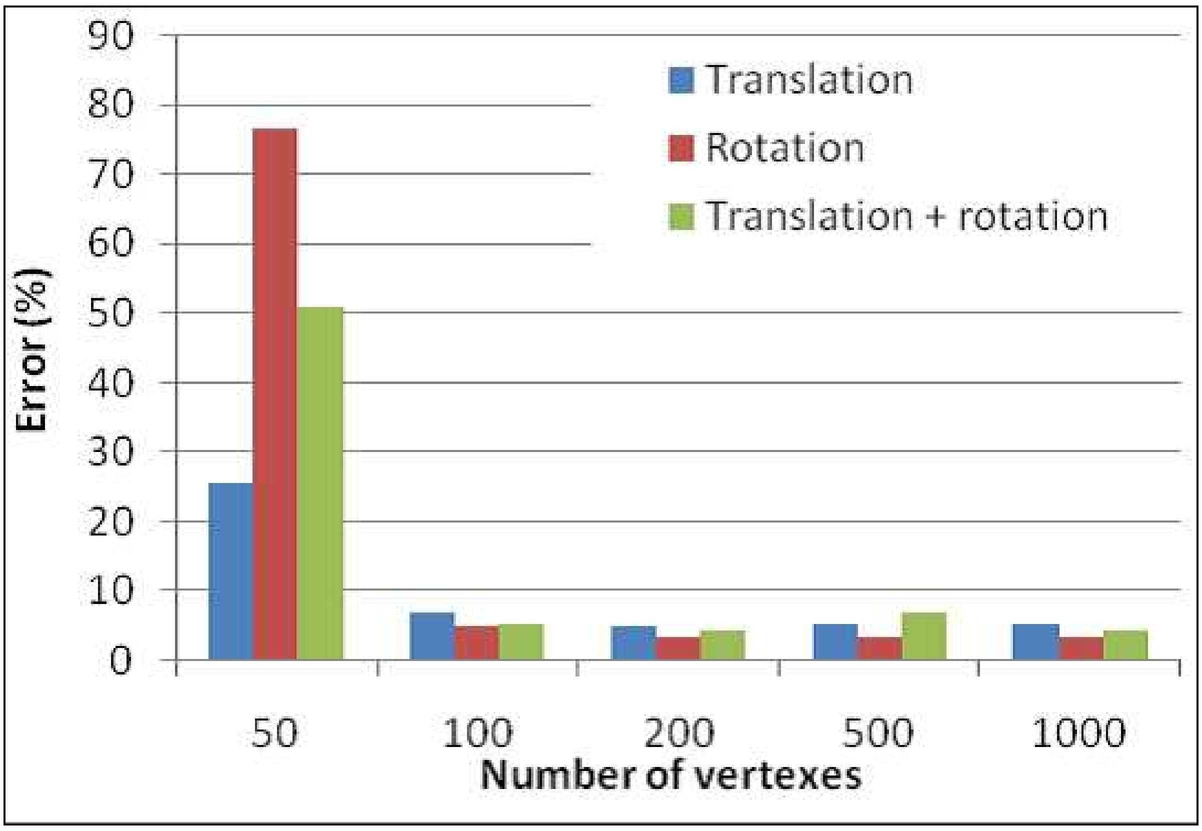


Fig. 9. Computational error depending on number of vertices.

	Planar motion	3D motion			3D motion with CHS		
		Static scene	Moving objects	Natural scene	Static scene	Moving objects	Natural scene
Average absolute error	0,59	0,44	0,59	4,61	0,33	0,37	4,3
Maximum absolute error	0,6	3,7	3,93	9,02	2,21	2,47	8.98

Table 3. Computational error (cm)

Table 3 shows that the proposed method can effectively estimate camera motion for scene with moving objects. However, its weak point is natural scenes with small number of geometrical objects.

5. Conclusion

In this paper we introduce an algorithm for converting images into graphs based on edges structure and graph matching algorithm based on vertex ranking. Proposed method could be used in different applications such as camera motion estimation, stereo matching, motion compensation, background model generation and many others. The proposed method provides efficient image-to-graph mapping for urban scenes. In case of natural scenes additional prefiltering may be required for removing unimportant information from edge image. In this paper we also presented simple algorithm for camera motion estimation based on parametric motion model. In future works we would like to improve the performance of the proposed algorithm to work in a real-time applications. Furthermore, algorithm could be improved for natural scenes by using different type of features.

6. Acknowledgments

This research was supported by the MKE(The Ministry of Knowledge Economy), Korea, under the Human Resources Development Program for Convergence Robot Specialists support program supervised by the NIPA(National IT Industry Promotion Agency) (NIPA-2010-C7000-1001-0007) and post-BK21 at University of Ulsan.

7. References

[1] W. Lie and W. Hsiao, "Content-based video retrieval based on object motion trajectory," in Proc. IEEEWorkshop on Multimedia Signal Processing, Dec. 2002, pp. 237–240.

[2] W. Hu, D. Xie, Z. Fu, W. Zeng, and S. Maybank, "Semantic-based surveillance video retrieval," IEEE Trans. Image Process., vol. 16, no. 4, pp. 1168–1181, Apr. 2007.

[3] Y. Jianfeng and L. Zhanhuai, "Modeling of moving objects and querying videos by trajectories," in Proc. 10th Int. Multimedia Modelling Conf., Washington, DC, 2004, pp. 373-380.

[4] Y.-P. Tan, D. D. Saur, S. R. Kulkarni, and P. J. Ramadge, "Rapid estimation of camera motion from compressed video with application to video annotation," IEEE Trans. Circuits Syst. Video Technol., vol. 10, no. 1, pp. 133–145, Jan. 2000.

[5] L.-Y. Duan, J. S. Jin, Q. Tian, and C.-S. Xu, "Nonparametric motion characterization for robust classification of camera motion patterns," IEEE Trans. Multimedia, vol. 8, no. 2, pp. 323–340, Apr. 2006.

[6] X. Zhu, A. K. Elmagarmid, X. Xue, L. Wu, and A. C. Catlin, "InsightVideo: Toward hierarchical video content organization for efficient browsing, summarization and retrieval," IEEE Trans. Multimedia, vol. 7, no. 4, pp. 648–666, Aug. 2005.

[7] T. Lertrusdachakul, T. Aoki, and H. Yasuda, "Camera motion characterization through image feature analysis," in Proc. ICCIMA'05, Aug. 16–18, 2005, pp. 186–190.

[8] A. Yao and A. Calway, "Robust estimation of 3-d camera motion for uncalibrated augmented reality", Technical Report CSTR-02-001, Dept of Computer Science, University of Bristol, 2002.

[9] O. Faugeras, Q.T. Luong, and T. Papadopoulo. "The Geometry of Multiple Images". MIT Press, 2000.

[10] M. Pollefeys, M Vergauwen, K Cornelis, J. Tops, F. Verbiest, and L. Van Gool "Structure and motion from image sequences" Proceedings of Conference on Optical 3-D Measurement Techniques V, pp. 251-258, 2001.

[11] Y. Ma, J. Koseck_a, and S. Sastry. "Linear differential algorithm for motion recovery: A geometric approach", IJCV, vol.36(1), pp.71-89, 2000.

[12] S.C. Park, H.S. Lee, and S.W. Lee, "Qualitative estimation of camera motion parameters from the linear composition of optical flow", PR(37), vol. 4, pp.767-779, 2004.

[13] Y. Tian, C. Tomasi, and D.J. Heeger, “Comparison of approaches to egomotion computation” IEEE Computer Society, Conference on Computer Vision and Pattern Recognition, pp.315-320, 1996.

[14] E. Francois and P.Bouthemy, “Derivation of qualitative information in motion analysis,” Image Vi-sion Computing, vol.8, no.4, pp. 279-287, Nov.1990.

[15] W.H. Press, B.P. Flannery, S. A. Teukolsky, and W.T. Vetterling, Numerical Recipes in C: The Art of Scientific Computing. Cambridge, U.K.: Cambridge Univ. Press, 1988, pp.59-70, pp.656-706.

[16] R. Grompone von Gioi, J. Jakubowicz, J-M.Morel and G.Randall, “On Straight Line Segment Detection”, Journal of Mathematical Imaging and Vision, vol.32(3), pp.313-347, November 2008.

[17] C. Beumier, “Straight-Line Detection Using Moment of Inertia”, IEEE International Conference on Industrial Technology, pp.1753-1756, 15-17 Dec. 2006.

[18] C. Tomasi and R. Manduchi, "Bilateral Filtering for Gray and Color Images", Proceedings of the 1998 IEEE International Conference on Computer Vision, Bombay, India, 1998.

[19] S S. Zhu, J. Tian, X. Shen, K. Belloulata,“A new cross-diamond search algorithm for fast block motion estimation”, the IEEE Int. Conf. Image Processing, *ICIP2009*, Cairo, Egypt, 07-11 Nov. 2009, Vol. I, pp. 1581-1584.

[20] Belloulata, K., Shiping Zhu, Jun Tian, Xiaodong Shen, "A novel cross-hexagon search algorithm for fast block motion estimation", Systems, Signal Processing and their Applications (WOSSPA), 2011 7th International Workshop on, On page(s): 1 - 4, Volume: Issue: , 9-11 May 2011

[21] Li Ren-xiang, Zeng Bing, Liou M. L, “A new three-step search algorithm for block motion estimation,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 4, pp. 438-442, April 1994.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821