Robust, Real-Time Motion analysis *

Moshe Ben-Ezra Shmuel Peleg Michael Werman

Institute of Computer Science
The Hebrew University of Jerusalem
91904 Jerusalem, Israel
Email: {moshe, peleg, werman}@cs.huji.ac.il

Abstract

An algorithm is presented for a robust real-time motion recovery. The algorithm uses point to line matches and L_1 error metric to reduce outliers and aperture effects.
The line-to-point match is implemented using weighted hough transform over a normalized correlation matrix. The motion parameters minimize the L_1 norm and are computed using linear programming.

1 Introduction

Real-time motion recovery is essential for time critical applications such as robotics and vision based military applications. Real-time performance is also beneficial for less critical tasks such as panoramic image where real-time enables better user interface (the user can see the panoramic image as it forms) and simpler application (there is no need for mass storage device to save the captured movie for off-line processing). The problem of robust motion recovery is a a difficult problem due to noise and outliers. Real-time requirement makes the problem even more difficult. This paper presents a new approach towards real-time performance where robustness is the key to performance by allowing fast single iteration algorithm.

1.1 Previous Work

Many methods have been developed for motion recovery from an images sequence, among them:

Probabilistic algorithm [8] - Algorithms that calculate the motion parameters from randomly selected pairs of matched points until they reach the desired accuracy. These algorithms may have several hundreds of iterations to achieve low error probability (less then 10^{-3} which corresponds to one-two errors a minute).

Direct computation from grey level [4, 5] - Algorithms are based on the constant brightness assumption and the optical flow constraint. These algorithms require good motion segmentation and are usually combined with motion segmentation methods in an iterative manner. These algorithms are usually CPU intensive. The application of direct computation from grey level to complex motion models (projective, fundamental matrix, tensor) is complicated.

Global alignment of local measures [3] - This algorithm is a generalization of the direct grey level algorithms in the sense that it is not restricted to grey level minimization. The algorithm defines *match-measure surface* over the local match field and uses Newton iterations to maximize (or minimize) the sum of the local measures.

Probabilistic matching [6, 7] - This method describe motion by normalized correlation matrices which enables direct recovery of the translation vector with maximum likelihood. Other parameters (rotation, scale) are recovered by extensive search over the parameter space.

Motion from Fuzzy Correspondences [2] - This method expresses uncertainty by convex polygons. The motion parameters are recovered by minimizing the L_1 norm. The method is limited to affine and pseudo-projective models.

The algorithm presented in this paper combines and extends the last two approaches, normalized correlation matrices are approximated by lines and City-Block distance metric, and the motion parameters minimizes the L_1 metric. The algorithm can express complex motion models as well.

*This research was funded by DARPA through the U.S. Army Research Labs under grant DAAL0197R9291.

2 Motion Computation

In order to gain real-time performance the algorithm relies heavily on the robustness properties of point to line matches and L_1 metric. The number of selected points that are used for motion recovery is extremely low (20-30 points) and no iteration are used. The algorithm is composed of the following steps:

1. Point Selection.
2. Point to line matching.
3. Alignment using L_1 metric.

2.1 Point Selection

Good selection of points should be spread across the image for good geometrical stability. Points should be located on strong features such as edges, and they should have balanced X-direction and Y-direction information. The point selection algorithm is:

1. Evenly spread N points across the image in a chess board grid.
2. Allow "black" points to move slightly horizontally to find strong vertical edges. Allow "white" points to move slightly vertically to find strong horizontal edges.
3. Select the best K "black" points and the best K "white" points ($2K < N$).

2.2 Point to Line matching

The correlation matrix of each points is transformed into $(\theta, distance)$ Hough spaces using weighted Hough transform. For each matrix, the algorithm selects the line with the maximum likelihood (the global maximum at the Hough space). A second line is optionally selected which is the second local maximum.

These two lines approximate the correlation matrix. The approximation is given by the sum of Euclidian distances of each matrix cell from these two lines. This approximation can express several common types of matches:

point-to-point match point $(x, y) \rightarrow (u, v)$ is expressed by the point (x, y) and the lines $(x = u, y = v)$.

point-to-line match is trivially expressed by the point and the detected line.

point-to-ellipse match is expressed by aligning the lines with the prime axes of the ellipse and assigning weights according to the eccentricity of the ellipse.

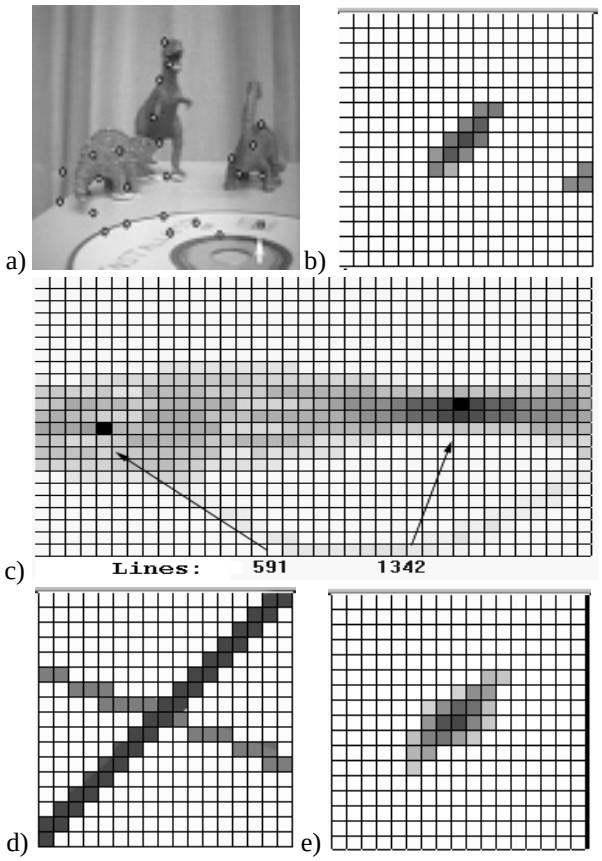

Figure 1. a) Point selection. b) Original correlation matrix for the point that is marked with a white arrow. c) Hough space of the correlation matrix. Arrows point to the peak locations. Peak values (591, 1342) are used as weights. d) Lines that correspond to the peak location. These lines are used to approximate the correlation matrix. e) Weighted City-Block distance approximation of the correlation matrix.

The selection of lines *and weights* is given by weighted Hough transform - no addition processing is required.

Figure 1 demonstrate the approximation of the correlation matrix by two lines.

2.3 Alignment using L_1 metric

For this section we use a 2D, eight parameter projective motion model.
A point $p = (x, y, 1)^t$ located at image-1 is mapped by the 2D projective transformation H into the (unknown) point $p' = \left(\frac{(H_1 \cdot p)}{(H_3 \cdot p)}, \frac{(H_2 \cdot p)}{(H_3 \cdot p)}, 1\right)^t$. The Euclidean distance of the point p' from a line $l = (Ax + By + C)$ located at image-2, is given by:

$$dist(p', l) = \frac{1}{\sqrt{A^2 + B^2}} \left(\frac{A(H_1 \cdot p)}{(H_3 \cdot p)} + \frac{B(H_2 \cdot p)}{(H_3 \cdot p)} + C \right) \quad (1)$$

Multiplying (1) by $(H_3 \cdot p)$ which is non-zero for finite size image and setting the distance to zero results with the following linear constraint:

$$\frac{1}{\sqrt{A^2 + B^2}} \left(A(H_1 \cdot p) + B(H_2 \cdot p) + C(H_3 \cdot p) \right) = 0 \quad (2)$$

In order to recover the eight parameters 2D projective transformation at least eight such point-to-line matches are required.

Recovering H from an over-constraint problem using L_1 is done by representing the equation set as the following linear problem:

$$min: \ C^t |Z_i|$$
$$s.t.$$

$$A'_i(H_1 \cdot p_i) \ + \ B'_i(H_2 \cdot p_i) \ + \ C'_i(H_3 \cdot p_i) \ + \ Z_i = 0$$

$$where:$$
$$A'_i = \frac{A_i}{\sqrt{A_i^2 + B_i^2}}, \quad B'_i = \frac{B_i}{\sqrt{A_i^2 + B_i^2}}, \quad C'_i = \frac{C_i}{\sqrt{A_i^2 + B_i^2}}$$
$$C \ is \ the \ weight \ vector \ (likelihood) \ of \ each \ line.$$

Notes:

1. This type of linear program can be used to solve any non-homogeneous linear equation set $Ax = b$.
2. If A is a (M, N) matrix $(M < N)$ then the linear program will have a total of $2(M+N)$ variables: $2N$ variables are required for the variable vector x, and $2M$ variables are requires for the slack variables z. The factor of two is needed since each unconstraint variable is represented by two non-negative variables.
3. For this particular type of linear program, where the constraints are of the form $Ax - b + z = 0$, a basic feasible solution is given by setting $x = 0, z = b$. This enables the use of an efficient one-phase simplex algorithm to solve the problem.
4. The slack variables z contains the error measure for each point and can be used for segmentation [2].

3 Experiments

3.1 Panoramic Image Creation

A panoramic image was created in real-time (10-12 frames / sec) using off-the-shelf PC. While the camera was scanning the scene a pendulum was swinging across the scene. The size of the pendulum was large enough to create 10 to 15 percent of outliers during the panorama forming. Since the stabilization algorithm used only frame to frame motion recovery *any* error will cause the panorama to fail. Figure 2 shows the pendulum (and its shadow) appear/disappear several times due to the swinging motion. However *all* frames were correctly aligned (by similarity model) as can be seen by the objects that were not occluded by the pendulum.

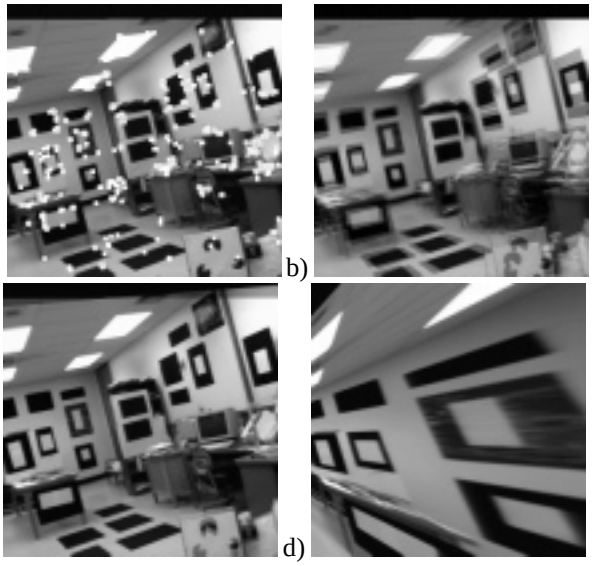

Figure 3. a) Selection points. b) Unregistered blending of image1 and image2. c) Projective registration using point to two lines formulation and L_1 registration. Blending of image1 and wrapped image2. d) Warping of image2 *using the same input data points* and a projective transformation recovered by least-square metric.

3.2 Projective Registration

In this experiment we have compared the L_1 registration to least-square point-to-point registration. The input points were selected from a bi-directional optical-flow field. Only points that were located on strong edges and agree on both directions were selected. The L_1 algorithm converted each point-to-point match as point into two point-to-line matches. The L_1 alignment has aligned most of the image (The box in the from of the image was considered an outlier and was not aligned). The least square alignment has failed.

a)  b) 

Figure 2. a) Point selection. The pendulum occupies four point of thirty. b) Panoramic image that was created while a pendulum was swinging. The similarity alignment was not affected by the outliers.

4 Concluding Remarks

This paper presents a new approach for real-time motion analysis based on: weighted Hough transform, point-to-line match and L_1 metric computed by linear programming. The robustness of the tools used allow the algorithm to be single resolution, single iteration which gives it real-time performance.

A linear programming L_1 equation has been introduced. This solver can be used in other computer vision problems that are sensitive to outliers (the equation system must be non-homogeneous).

References

[1] M. Ben-Ezra, S. Peleg, and B. Rousso. Motion segmentation using convergence properties. In *Image Understanding Workshop (ARPA94)*, pages II:1233–1235, 1994.

[2] M. Ben-Ezra, P. S., and M. Werman. Efficient computation of the most probable motion from fuzzy correspondences. In *IEEE Workshop on Applications of Computer Vision (WACV98)*, 1998.

[3] M. Irani and P. Anandan. Robust multi-sensor image alignment. In *ICCV98*, pages 959–966, 1998.

[4] M. Irani, B. Rousso, and S. Peleg. Recovery of ego-motion using region alignment. In *PAMI*, volume 19, pages 268–272, March 1997.

[5] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI81*, pages 674–679, 1981.

[6] Y. Rosenberg and M. Werman. Representing local motion as a probability distribution matrix and object tracking. In *DARPA Image Undersading Workshop*, pages 153–158, 1997.

[7] Y. Rosenberg and M. Werman. Real-time tracking from a moving camera: A software approach. In *IEEE Workshop on Applications of Computer Vision (WACV98)*, 1998.

[8] P. Torr and D. Murray. Outlier detection and motion segmentation. In *SPIE*, volume 2059, pages 432–443, 1993.