AUTOMATIC RECTIFICATION OF LONG IMAGE SEQUENCES

Kenji Okuma, James J. Little, David G. Lowe

The Laboratory of Computational Intelligence
The University of British Columbia
Vancouver, British Columbia, V6T 1Z4

Abstract

This paper addresses the problem of automatically computing homographies between successive frames in image sequences and compensating for the panning, tilting and zooming of the cameras. A homography is a projective mapping between two image planes and describes the transformation created by a fixed camera as it pans, tilts, rotates, and zooms around its optical centre. Our algorithm achieves improved robustness for large motions by combining elements of two previous approaches: it first computes the local displacements of image features using the Kanade-Lucas-Tomasi (KLT) tracker and determines local matches. The majority of these features are selected by RANSAC and give the initial estimate of the homography. Our model-based correction system then compensates for remaining projection errors in the image to rink mapping. The system is demonstrated on a digitized sequence of an NHL hockey game, and it is capable of analyzing long sequences of consecutive frames from broadcast video by mapping them into the rink coordinates.

1. INTRODUCTION

With the advance of information technologies and the increasing demand for managing the vast amount of visual data in video, there is a great potential for developing reliable and efficient systems that are capable of understanding and analyzing scenes. In order to design such systems that describe scenes in video, it is essential to compensate for camera motions by estimating a planar projective transformation (i.e., homography) [3, 5, 6, 9, 12]. This paper has two major contributions. One is to present an algorithm for automatically computing homographies by combining the KLT tracking system [1, 8, 10], RANSAC [2] and the normailzed Direct Linear Transformation (DLT) algorithm [3]. The other is to describe a new model-based correction system that fits projected images to the model and reduces projection errors produced by automatically computed homography. Our system detects features that lie on line segments of projected images and minimize the difference between projected images and the model using the normalized DLT algorithm. Similarly, Koller *et. al* [7] uses line segments of moving vehicles to track them from road traffic scenes monitored by a stationary camera. Yamada *et. al* [11] uses line segments and circle segments of the soccer field to estimate camera parameters and mosaic a short sequence of video images in order to track players and a ball in the sequence.

In the subsequent section, the theoretical background of the homography (also known as a plane projective transformation, or collineation) is described. The third section describes our algorithm for automatically computing homography between successive frames in image sequences. The fourth section explains our model-based correction system for compensating projection errors produced by automatic computation of homography. In the fifth section, the result of our experiments is presented. The final section concludes this paper and indicates future directions of our research.

2. HOMOGRAPHY

The definition of a homography (or more generally *projectivity*) in [3] is an invertible mapping of points and lines on the projective plane $\mathbb{P}^2$. This gives a homography two useful properties. For a stationary camera with its fixed centre of projection, it does not depend on the scene structure (i.e., depth of the scene points) and it applies even if the camera "pans and zooms", which means to change the focal length of the camera while it is rotating about its centre. With these properties, a homography is applied under the circumstance which the camera pans, tilts, rotates, and zooms about its centre.

2.1. Representation of Homography

Homogeneous representation is used for a point $\mathbf{x} = (x, y, w)^\top$, which is a 3-vector, representing a point $(x/w, y/w)^\top$ in Euclidean 2-space $\mathbb{R}^2$. As homogeneous vectors, points are also elements of the projective space $\mathbb{P}^2$. It is helpful to consider the inhomogeneous coordinates of a pair of matching points in the world and image plane as $(x/w, y/w)^\top$ and

$(x'/w', y'/w')^\top$, because points are measured in the inhomogeneous coordinates directly from the world plane. According to [12], a homography is a linear transformation of $\mathbb{P}^2$, which is expressed in inhomogeneous form as:

$$x'/w' = \frac{Ax + By + C}{Px + Qy + R}, \; y'/w' = \frac{Dx + Ey + F}{Px + Qy + R} \quad (1)$$

where vectors $\mathbf{x}$ and $\mathbf{x}'$ are defined in homogeneous form, and a transformation matrix $\mathbf{M}$ as:

$$\mathbf{x} = \begin{pmatrix} x \\ y \\ w \end{pmatrix} \qquad \mathbf{x}' = \begin{pmatrix} x' \\ y' \\ w' \end{pmatrix} \qquad \mathbf{M} = \begin{bmatrix} A & B & C \\ D & E & F \\ P & Q & R \end{bmatrix}$$

where $\mathbf{x} \leftrightarrow \mathbf{x}'$ denotes a pair of 2D point correspondences. Normally the scale factor w is chosen in such a way that x/w and y/w have order of 1, so that numerical instability is avoided.

Now, Eq. (1) can be written as:

$$\mathbf{x}' = c\mathbf{M}\mathbf{x} \quad (2)$$

where c is an arbitrary nonzero constant. Homographies and points are defined up to a nonzero scalar c, and thus there are 8 degrees of freedom for homography. Often, $R = 1$ and the scale factor is set as $w = 1$. Eq. (2) can now be written simply as:

$$\mathbf{x}' = \mathbf{H}\mathbf{x}$$

where $\mathbf{H}$ is 3×3 matrix called a homography. Every correspondence ($\mathbf{x}$, $\mathbf{x}'$) gives two equations. Therefore, computing a homography with this algorithm requires at least four correspondences. The normalized DLT algorithm [3] is used to compute frame-to-frame homographies. Figure 1 shows the result of homography transformation by the normalized DLT algorithm based on manually selected correspondences.

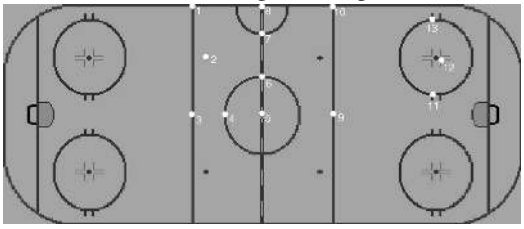
3. COMPUTATION OF THE HOMOGRAPHY

Given a sequence of images acquired by a broadcast camera, the objective is to specify a point-to-point planar homography map in order to remove the camera motion in images. Our algorithm has four major steps to automatically compute homographies.

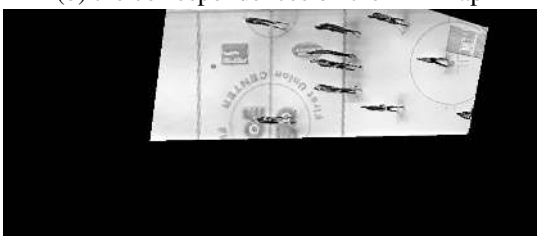
3.1. Reduce vision challenges

Since the source of our data is video clips of broadcast hockey games, there are various vision problems to deal with, namely camera flashes that cause a large increase of image intensities and rapid motions of broadcast cameras for capturing highly dynamic hockey scenes.

(a) The original image



(b) the correspondences on the rink map



(c) The transformation result

Fig. 1. **Homography Transformation.** (a) shows the original image (320×240) to be transformed. (b) shows manually selected points that are corresponding to those on the rink in the image, which are used only for the initial frame in a video sequence. The correspondences are paired up by the numeric number. (c) is the result (1000×425) of the transformation.

3.1.1. Flash Detection

In order to deal with camera flashes in digitized hockey sequences, automatic detection of those flashes is necessary. Figure 2 shows the average intensity of over 2300 consecutive frames.

In the graph, there are several sudden spikes which indicate that there is a camera flash for that particular frame. With our observation of camera flashes, a simple flash detection method is derived by taking the difference of the average intensity from two successive frames.

3.1.2. Prediction

Broadcast cameras often make rapid motions to capture dynamic hockey scenes during a game. The amount of motion, however, can be reduced by *predicting* the current camera motion based on the previous camera motion. For instance, given a frame-to-frame homography $\mathbf{H}_{1,2}$ that represents the camera motion from Frame 1 to Frame 2, $\mathbf{H}_{1,2}$ is used

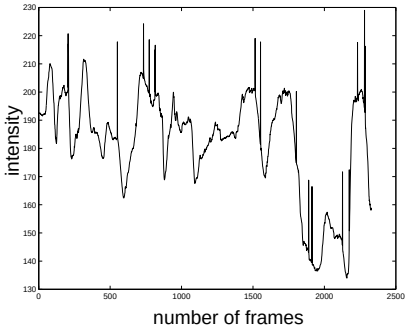


Fig. 2. **The average intensities over 2300 frames.** The vertical axis indicates the number of the intensity ranging from 130 to 230 where 130 indicates a darker pixel and 230 is a brighter pixel. The horizontal axis is the number of the frame.

as the estimation of $\mathbf{H}_{2,3}$ to transform Frame 2 so that we can minimize the amount of motion between Frame 2 and Frame 3. That is, to have the following assumption:

$$\mathbf{H}_{\mathrm{n,n-1}} \approx \mathbf{H}_{\mathrm{n+1,n}}$$

where every successive frame is processed and $\mathbf{H}_{n-1,n}$ means a homography from Frame $n-1$ to Frame n. This assumption holds only without skipping too many frames. In our experiments, our system processes every fourth frame of data sampled at 30 frames per second, and shows that it is capable of compensating a large motion of a camera.

3.2. Acquisition of correspondences

For successful homography computation, it is crucial to have a reliable set of point correspondences that gives an accurate homography. KLT gives those correspondences automatically by extracting features and tracking them. That is, those features that are successfully tracked by KLT between images are ones that are corresponding to each other.

3.3. RANSAC: Elimination of outliers

Correspondences gained by KLT are yet imperfect to estimate a correct homography because they also include outliers. Though an initial set of correspondences selected by KLT contains a good proportion of correct matches, RANSAC is used to identify consistent subsets of correspondences and obtain a better homography. In RANSAC, a putative set of correspondences is produced by a homography based on a random set of four correspondences, and outliers are eliminated by the homography.

3.3.1. Sample Selection

Distributed spatial sampling is used to avoid choosing too many collinear points to produce degenerate homography. In the sampling, a whole image is divided into four sub-regions of an equal size so that each correspondence is sampled from a different sub-region. Once four point correspondences are sampled with a good spatial distribution, a homography is computed based on those correspondences and use the homography to select an initial set of inliers. For inlier classification, we use the symmetric transfer error $d^2_{transfer}$, defined in [3]:

Let $\mathbf{x} \leftrightarrow \mathbf{x}'$ be the point correspondence and $\mathbf{H}$ be a homography such that $\mathbf{x}' = \mathbf{H}\mathbf{x}$, then

$$d^2_{transfer} = d(\mathbf{x}, \mathbf{H^{-1}}\mathbf{x}')^2 + d(\mathbf{x}', \mathbf{H}\mathbf{x})^2 \qquad (3)$$

where $d(\mathbf{x}, \mathbf{H^{-1}}\mathbf{x}')$ represents the distance between $\mathbf{x}$ and $\mathbf{H^{-1}}\mathbf{x}'$. After the symmetric transfer error is estimated from each point correspondence, we then calculate the standard deviation of the sum of the symmetric errors from all correspondences, which is denoted by σ_{error} and defined as follows:

Suppose there are N point correspondences and each one of them has the symmetric transfer error $\{d^2_{transfer}\}_{i=1\ldots N}$, then:

$$\sigma_{error} = \sqrt{\frac{\sum_{1\le i\le N}(\{d^2_{transfer}\}_i - \mu)^2}{N-1}} \qquad (4)$$

where μ is the mean of the symmetric errors. Now we can classify an outlier as any point $\mathbf{x}_i$ that satisfies the following condition:

$$\gamma(\mathbf{x}_i) = \begin{cases} 0 & \{d^2_{transfer}\}_i \ge \sqrt{5.99} * \sigma_{error} \quad \text{(outlier)} \\ 1 & \text{Otherwise} \quad \text{(inlier)} \end{cases} \qquad (5)$$

where γ is a simple binary function that determines whether the point $\mathbf{x}_i$ is an outlier. The distance threshold is chosen based on a probability of the point being an inlier. The constant real number, $\sqrt{5.99}$, is, therefore, derived by computing the probability distribution for the distance of an inlier based on the model of which this case is the homography matrix [3].

3.3.2. Adaptive termination of sampling

After sampling four spatially distributed correspondences and classifying inliers and outliers, the termination of sampling needs to be determined in order to save unnecessary computation. An adaptive algorithm [3] for determining the number of RANSAC samples is implemented for that purpose. The adaptive algorithm gives us a homography that produces the largest number of inliers by adaptively determining the termination of the algorithm with respect to the

probability of at least one of the random samples being free from outliers and that of any selected data point being an outlier.

3.4. Selection of best inliers

The set of inliers selected by RANSAC sometimes contains a large number of matches. This set is further refined by eliminating points with a large amount of the symmetric transfer error in Eq.(3) and making a set of better inlying matches. The aim of this further estimation is, therefore, to obtain an improved estimate of a homography with better inliers selected by randomly selected 100 point correspondences, instead of being selected by only randomly selected four point correspondences in RANSAC. The number, 100, is chosen because a least square solution of more than 100 point correspondences requires inefficient amount of computation. If the set of inliers contains less than 100 matches, then this process is skipped.

The process of the further estimation is that at each iteration, a homography is estimated with a set of 100 randomly selected point correspondences that are considered to be inliers, classify a set of all correspondences based on our simple classifier in Eq.(5) and update a set of inliers. the process is repeated until the symmetric error of all the inlier becomes less than $\sqrt{5.99} * \sigma_{error}$. An important remark of this estimation process is to take an initial set of correspondences into account without eliminating any one of them, and to consider some outliers being re-designated as inlilers.

4. MODEL FITTING

In order to reduce projection errors from automatic computation of the homography, model fitting is applied to the result of the homography transformation. The rink dimensions and our model are strictly based on the official measurement presented in [4]. Our model consists of features on lines and circles of the rink. There are 296 features in total: 178 features on four End-Zone circles, 4 features on centre ice face-off spots around the centre circle, and 114 features on lines.

4.1. Edge search

This section describes how to fit projected images to our model of the rink and reduce projection errors produced by automatic computation of homography. In order to fit the projected images to the model, a local search is performed on each model point appearing within the region of each projected image. The local search is conducted to find the nearest edge pixel in the image. Figure 3 shows how to fit the projected image to our rink model.

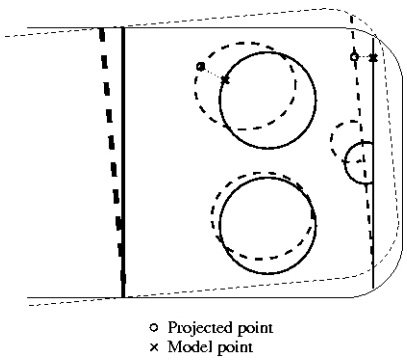


Fig. 3. **Fitting a projected image to our model of the rink.** Dotted lines represent the projected image and solid lines represent the model. Although only two examples of matching a projected point to a model point are presented in this image, a local search is performed for finding the nearest edge pixel (i.e., a projected point) from all model points appearing within the projected image.

For edge detection, the search is performed locally only on high gradient regions in the original sequence where there are most likely edges in order to save on the computational time. In the search on high gradient regions, edge orientation is considered to find a most likely edge pixel. Given an image, I, the image gradient vector $\mathbf{g}$ is represented as:

$$\mathbf{g}(\mathbf{x}) = \begin{pmatrix} \frac{\partial}{\partial x}(I) \\ \frac{\partial}{\partial y}(I) \end{pmatrix}$$

The gradient vector represents the orientation of the edge. The orientation is perpendicular to the direction of the local line or circle segment. Figure 4 shows the orientation of two edges that form a thick line in the image. Since lines and circles of the hockey rink are not single edges but thick lines, they give two peaks of gradients. The image gradient vector $\mathbf{g}$ is computed from the original image because the projected image may not give accurate gradients due to resampling effects. Figure 5 shows how the edge search is conducted.

As it is shown in the figure, the edge search does not perfectly detect all the edge pixels on the rink surface. For instance, in (b) of Figure 5, there is one edge pixel that does not belong to any lines in the left bottom face-off circle. Furthermore, there are not many edge points detected on the centre circle since there are many gradient peaks detected on the line of the circle, the edges of the logo, and the edges of the letters. In order to avoid finding edge points that are not on the edge of the circles or lines on the rink, our edge search ignores ambiguous regions with many edges by detecting multiple gradient peaks in the search region. Given n edge points found by our edge search, these points can be used to compute a transformation, $\mathbf{H}_{corr}$, to

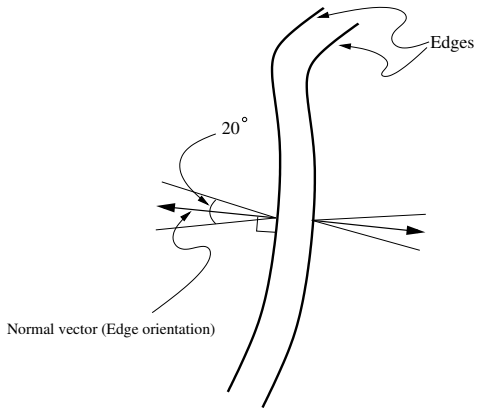
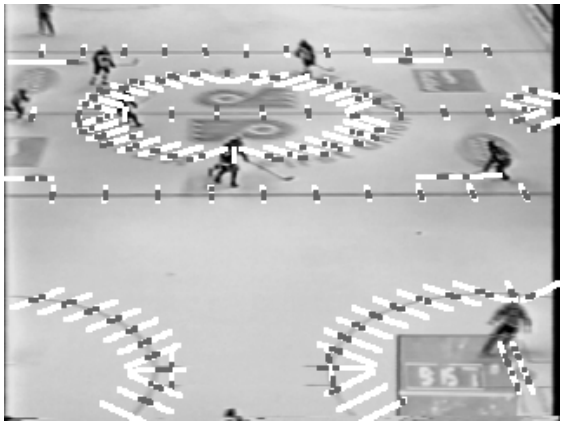
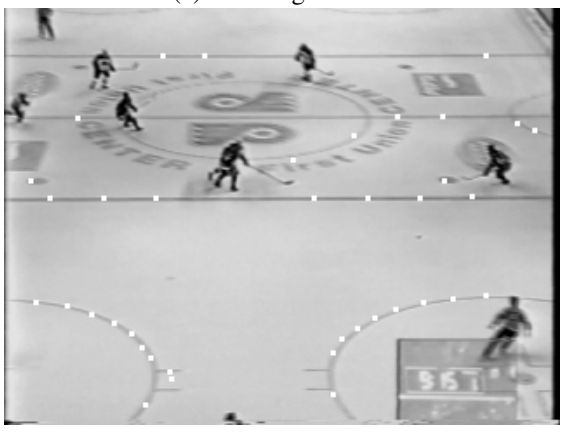


Fig. 4. **Edge orientation.** The orientation of the edge is represented as the normal vector (i.e., gradient vector) that is perpendicular to the edge. The threshold is set as 20° to match the orientation.

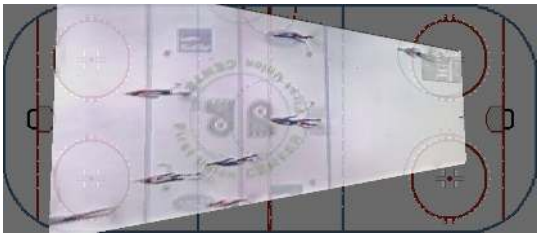


(a) local edge search



(b) Edge points found by the search

Fig. 5. **Searching edges.** (a) shows the search regions (lighter points) and high gradient regions (darker points). It is shown that edges lie on high gradient regions. (b) is the result of the edge search. It shows how successfully our search detects edge points for each model points.



(a) The result without fitting the model

(b) The result with fitting the model

Fig. 6. **The result of our model fitting.** (a) is the result after 323 frames without using the model fitting. (b) is the result after 323 frames with the model fitting. (b) clearly shows a more accurate projection over 300 frames.

rectify a projected image to the model. The normalized DLT algorithm is used to compute $\mathbf{H}_{corr}$ based on 2D to 2D point correspondences $\{\mathbf{x}_i^{Edge} \leftrightarrow \mathbf{x}_i^{Model}\}_{i=1\ldots n}$ where $\{\mathbf{x}_i^{Edge}\}_{i=1\ldots n}$ denote n edge points detected by our edge search and $\{\mathbf{x}_i^{Model}\}_{i=1\ldots n}$ are n corresponding model points. Overall, our edge search gives us reliable performance and can prove that our model fitting system works well. Figure 6 shows how effective our model fitting is for reducing accumulative projection errors over a sequence of frames.

5. EXPERIMENTS

This section presents the result of our experiments. In Figure 7, our system is demonstrated on a sequence of 1900 frames that is digitized from a video clip of NHL hockey games on TV. The system processes every fourth frame and rectifies them by computing 1200 KLT features from which the best inliers are selected. Once a set of correspondences are manually selected only on the very first frame of the sequence to compute the transformation between the image and rink mapping, homographies between the rest of the sequence and rink mapping are automatically computed by our algorithm. Our non-optimized implementation in C on a 2.8 GHz Pentium IV takes about an hour to process 1900 frames of data. Figure 7 shows the successful automatic rectification. Although our system is demonstrated on Hockey data at this time, our algorithm is also applicable to other domains of sports such as soccer and football or any other planar surface scenes with identifiable features.

6. CONCLUSION

This paper describes an automatic system of computing homographies over a long image sequence and rectifying the sequence by compensating for the panning, tilting and zooming of the cameras. Since our model-based correction system performs a local search of both straight and circular line segments and distinguishes them by their orientation, it does not require direct methods of conic detection or line detection. It achieves robustness by combining a number of different methods that would not be sufficient on their own.

Our system is easily applicable to different scenes such as soccer, football, or many other scenes that have a planer surfaces with identifiable features and line segments. Among many directions and improvements considered in future, the speed up of computation is primarily required to make our system a practical application.

7. REFERENCES

[1] S. Birchfield. *Depth and motion discontinuities*. PhD thesis, Stanford University, 1999.

[2] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981.

[3] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, June 2000.

[4] U. H. Inc. The official rules of ice hockey, 2001.

[5] M. Irani, P. Anandan, J. Bergen, R. Kumar, and S. Hsu. Efficient representations of video sequences and their applications, 1996.

[6] K. Kanatani and N. Ohta. Accuracy bounds and optimal computation of homography for image mosaicing applications. In *Proceedings of the 7th IEEE International Conference on Computer Vision (ICCV-99)*, volume I, pages 73–79, Los Alamitos, CA, Sept. 20–27 1999. IEEE.

[7] D. Koller, K. Daniilidis, and H. Nagel. Model-based object tracking in monocular image sequences of road traffic scenes. *IJCV*, 10(3):257–281, June 1993.

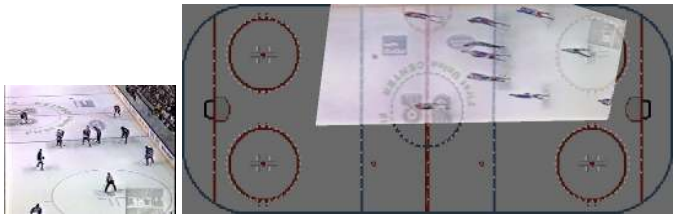
[8] J. Shi and C. Tomasi. Good features to track. Technical Report TR93-1399, Cornell University, Computer Science Department, Nov. 1993.

[9] R. Szeliski. Image mosaicing for tele-reality applications. In *WACV94*, pages 44–53, 1994.

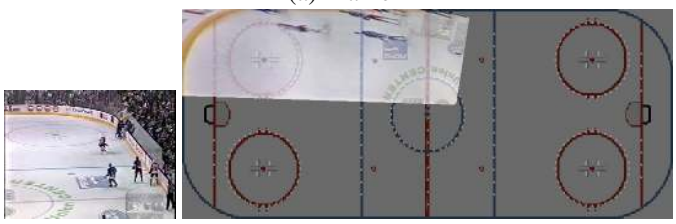
[10] C. Tomasi and T. Kanade. Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon University, Computer Science Department, 1991.

[11] A. Yamada, Y. Shirai, and J. Miura. Tracking players and a ball in video image sequence and estimating camera parameters for 3D interpretation of soccer games. In *ICPR02 VOL I*, pages 303–306. IEEE, 2002.

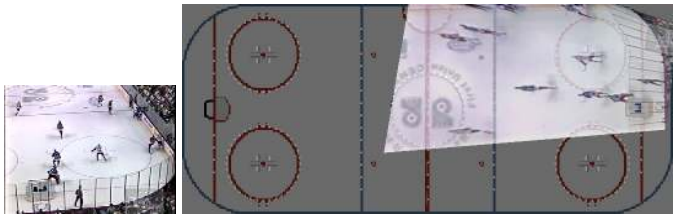
[12] I. Zoghiami, O. Faugeras, and R. Deriche. Using geometric corners to build a 2d mosaic from a set of images. In *CVPR97*, pages 420–425, 1997.



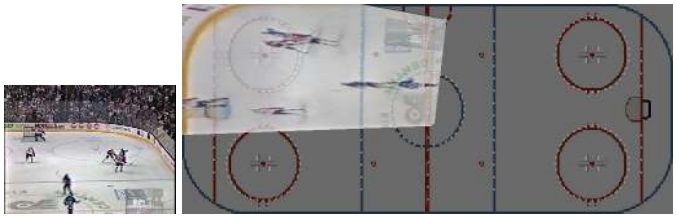
(a) Frame 1



(b) Frame 600



(c) Frame 1200



(d) Frame 1850

Fig. 7. **Automatic rectification result.** The figure shows the result of our algorithm on over 1800 frames on hockey data. The left column shows the original image (320×240) to be transformed and on the right, it shows a rectified image that is superimposed on the model of the rink map.