Dense Matching by Conservative Search of Transformation Space

Seth Teller Eric Amram Michael Werman

Abstract

A simple method to find a dense match of a planar object is presented. The basic idea is structured recursive search of transformation space using a conservative predicate. The predicate *excludes* regions known not to contain a matching transformation; the remaining set of candidate transformations are examined for matches by traditional methods. Our technique is demonstrated to match planar regions even under changes in lighting, and extreme geometric transformations.

1 Introduction

Matching images is a long-standing problem in computer vision, used in stereo, motion, object recognition, compression (e.g., MPEG, fractal [ABT97]), and other contexts. Here we consider the case in which the search region is known (or hypothesized) to form the image of some planar surface in the world.

Matching algorithms have often been classified as either *area-based* or *feature-based*. Feature-based methods require robust operators to find the features and give rise to sparse correspondences. Area-based methods produce dense depth maps without requiring special features to be present, but usually assume local planarity of the 3D object.

Area-based methods, though they yield a dense correspondence, can fail in the presence of depth discontinuities, occlusion, specularities and noise. The main ideas proposed in area-based matching algorithms have to do with designing distance measures between image patches, so that patches separated by a small distance are considered a match. The patches to compare are either searched for exhaustively (e.g. with a multi-resolution representation) or through anchoring to some set of features. This limits the search to a small class of transformations, or ends up being essentially feature-based.

This paper proposes a new way to identify patches to test for matching. Our method excludes large portions of the transformation space; remaining *candidate* transformations are checked for a match using traditional methods.

1.1 Related work and existing methods

We can divide existing methods into two categories: feature-based and featureless methods (minimizing mean squared error, or based on spatiotemporal derivatives and optical flow).

Representatives of feature-based methods are for example [Wol90, DHU90, AKM$^+$92].

There are two kinds of featureless methods, those based on optical flow for example [Sze94, SK95, KAI$^+$95, IAH95, MP94, MP95]) or those based on parameter estimation such as [McM95, Sze94, MP95].

The closest methods to ours in that they try to perform hierarchichal search of transformation space are [HKR93, CGH$^+$93, Ruc97, HV97, DMML97]. These methods recursively divide transformation space in order to find candidate transformations. However they all try to do point matching up to a transformation; that is, they are feature-based and work on a set of points. Points act specially under these transformations as they remain the same shape. They also use variations on beam search, by computing estimates of the error function of all possible matches, and deleting nodes all of whose children have an error function provably larger than the minimum value already found. Our algorithm, in contrast, is much faster; it *excludes* regions of transformation space known not to contain the sought transformation, producing a set of candidate regions which are then inspected by traditional means.

2 The Matching Algorithm

Given: a region p of some image, and an image P.

The Problem: Find a transformed instance of p in P. That is, find the transformation $t \in T$ which maps p to its match in P, where T is a set of transformations (Figure 1).

In order to find the transformed p in P we make use of $Q(p, P, T)$, a *predicate* which returns FALSE only when there is no transformation $t \in T$ such that t applied to p is in P. (Note that this predicate is "conservative;" it may return TRUE even when there is no matching transformation in T.)

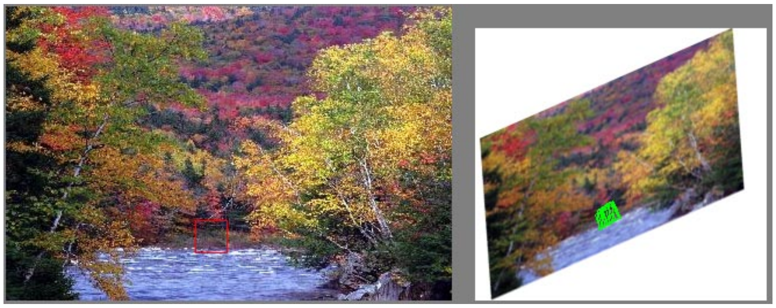

Figure 1: Matching a quadrilateral subregion between two images.

Given an image region p, an image P, and a set T of transformation space, the predicate $Q(p, P, T)$ is false only when T does not contain a transformation mapping p to a matching region of P. The predicate Q must be a monotone Boolean function, in the sense that $\neg Q(p, P, T_2) \Rightarrow \neg Q(p, P, T_1)$ whenever $T_1 \subset T_2$. That is, exclusion of a matching transformation from a set implies exclusion from all its subsets.

Given a set T of possible transformations, the algorithm applies Q to determine whether T might contain a match. If not, the algorithm terminates. Otherwise, if T is sufficiently small, it is returned as a "candidate match", to be examined for a match through traditional means (e.g., sampling and correlation). Otherwise, T is divided into subsets, and each is examined recursively. This can be done either using a "quadtree" strategy (divide a d-dimensional region into 2^d subregions by bisecting along each dimension) or a "$k - d$" tree strategy (loop through the dimensions, bisecting each in turn [Ben75]).

Here is pseudocode for the algorithm:

```
**Find** $(p, P, T)$
    **if** $|T| < \epsilon$
        **output** $T$
    **else if** $Q(p, P, T)$ **then**
        $\forall T_i \in T$  // $(\bigcup_i T_i = T)$
            **Find**$(p, P, T_i)$
```

2.1 Families of Transformations

We have studied the following types of transformations: Translations (2 scalar degrees of freedom); Stereo Group (4 DOF); Affine (6 DOF), and Projective (8 DOF).

2.1.1 Translation

The simplest interesting transformation is translation; that is, $x', y' = x + A, y + B$. Clearly the space of translations is two-dimensional. Suppose the images are each coordinatized as $[0, 1]^2$; then the region of "translation space" to search is $[-1, 1]^2$. Our implementation represents this region as a square.

2.1.2 Stereo Group

When the epipolar constraint relating the image pair is known, searching for a polygon region match involves one scalar parameter per region vertex. The simplest instance of this is to match a quadrilateral, thus searching a four-dimensional transformation space. Equivalently, one might imagine each vertex of the source quadrilateral projecting to an epipolar line in the destination image; our problem is to deduce the correct (1-D) position of each matched vertex on its respective epipole. Our implementation represents this region as a 4D hypercube. (Note that matching the vertices of a triangular region would not be sufficient, as four points are needed to fix a projective transformation.)

2.1.3 Affine

Affine transformations are often used as approximations to the full projective group, when the imaged objects are not close to the camera. In this case, our implementation represents the search region as a six-dimensional hypercube.

2.1.4 Projective

The complete projective group. The simplest parametrization of this group, especially when p is a quadrilateral, are the coordinates of the image of p in P, Initially, they can be anywhere in P; their ranges are reduced at each step of the algorithm by subdivision of an eight-dimensional hypercube.

3 The Conservative Predicate Q

The above description of the algorithm describes the matching predicate generically: it need only be

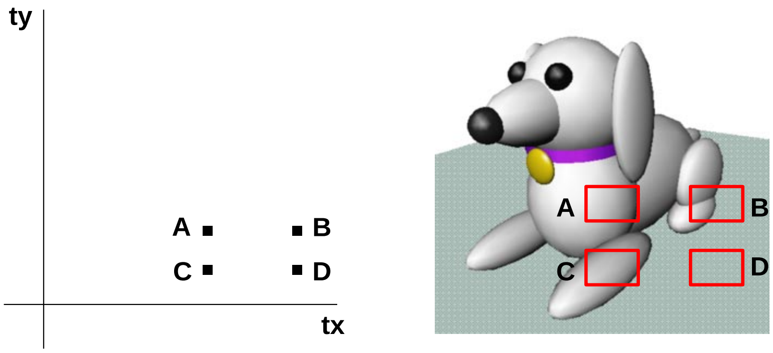

Figure 2: Each point in transformation space (at left) defines a quadrilateral in the target image (at right).

monotonic and conservative, as described. However, implementing the algorithm requires a specific choice of predicate. We define two pictures as possible matches if they contain roughly the same set of colors (or greyscale levels for unsaturated colors).

When searching for matches in some set T of transformation space, we examine the mapping of p under *all* $t \in T$: we denote as P' this subregion of image P. Now, in order for T to contain a matching transformation, it must be true that the colors (grey scales) in P' include those in $t(p)$, for some $t \in T$. To determine inclusion, we use a *histogram* of colors and grey scales. The following subsections describe our method for computing histograms.

3.1 Histogram Computation

The fact that the world-space surface is geometrically transformed between the two images implies that it or the camera has moved; therefore, in general we can expect some degree of view-dependent lighting effects to alter the appearance of the surface. We found that RGB histograms were not stable under changes in illumination. In order to achieve some degree of color constancy, we adopted the following scheme. Each image is converted to HSV (hue, saturation, value). Each pixel is then classified as saturated ($S > s_0$) or unsaturated ($S \leq s_0$), for some threshold value s_0. Saturated pixels are histogrammed according to their hue (H), while unsaturated pixels are histogrammed according to their value (V).

To allow a color that is near a border to vary, we use overlapping intervals: a color belongs to two intervals if it is close enough to another cluster.

For each region of T to be examined, two histograms must be compared: a histogram of the pixels in the image region p, under all transformations $t \in T$; and a histogram of the pixels in the subregion P'. For efficiency, we compute a "quadtree" of histograms in P; that is, the image is decomposed as a quadtree, and the histogram of all pixels contained in a quadtree node is stored with that region. Thus, to find the histogram of an arbitrary polygonal region P', we simply sum the histograms of all nodes contained in P' whose parents are not contained in P'.

The histogram of $t(p)$ (that is, of p under the action of all transformations in set t) contains, in each bin, the *minimum* of all histogram values for that bin. We compute this minimum differently for each of the four cases of transformation. Under translation, the histogram in invariant. Under affine transformation, the histogram simply scales by the determinant of the transformation. Under the stereo and full projective groups, we sample over the space of transformations, and take a minimum of the histograms generated from each individual transformation of p.

4 Implementation and Results

We implemented the search method, using the conservative predicates described above, on an SGI workstation. We parallelized the algorithm implementation straightforwardly using a producer-consumer model of work generation and execution.

4.1 Book Cover Matching

Figure 4 shows the algorithm run on a pair of images of a course bulletin with a colored cover. A subregion of the image at left is manually selected. The algorithm finds the best instance of the selected region in the target image.

4.2 Building Facade Matching

Figure 5 shows the algorithm run on a pair of images of an architectural scene. The images were acquired from two cameras tens of meters apart. Part of one building facade is manually selected in the image at left. The algorithm identifies the matching section of the facade, in the image at right.

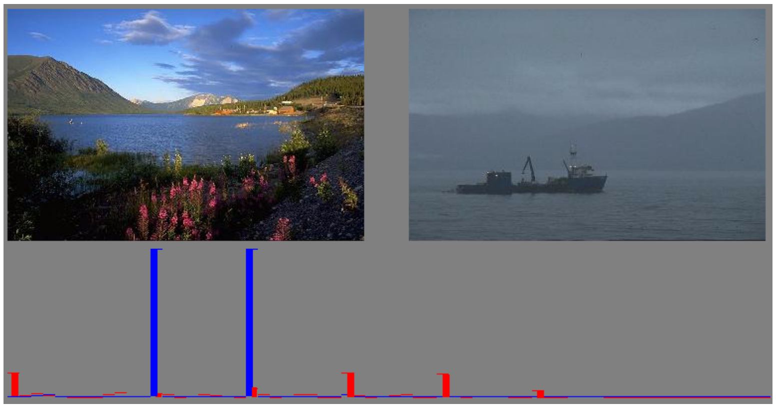

Figure 3: Histograms of images are very often different (red bars represent the histogram of the image at left; blue bars of the image at right.

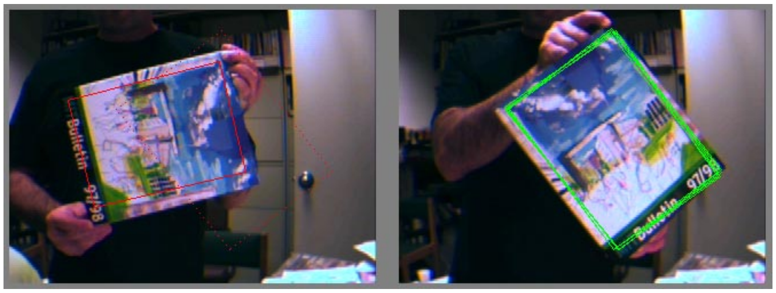

Figure 4: A match between a book cover in two images.

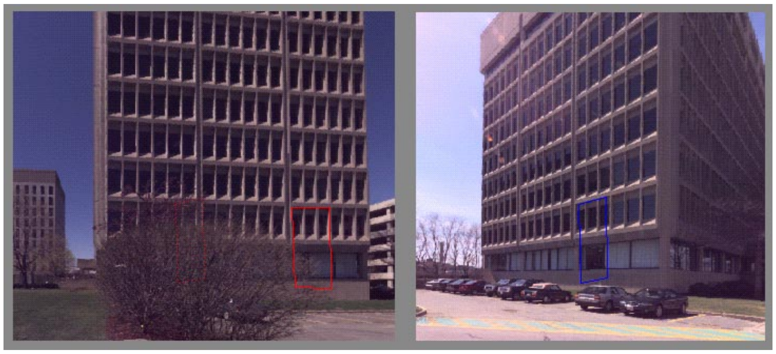

Figure 5: Part of a building facade, matched across two images.

5 Conclusion

We have described a recursive, exclusive search method which operates by excluding large regions of transformation space. Our method can identify extreme transformations between image regions. The algorithm has been implemented and used to match regions of actual images.

References

[ABT97] E. Amram and J.-J. Blanc-Talon. Fractal image compression: an efficient acceleration technique. In *CISST*, 1997. http://graphics.lcs.mit.edu/amram/fractals.ps.gz.

[AKM+92] E. M. Arkin, K. Kedem, J. S. B. Mitchell, J. Sprinzak, and M. Werman. Matching points into pairwise disjoint noise regions: Combinatorial bounds and algorithms. *ORSA Journal on Computing, special issue on computational geometry*, pages 27–52, 1992.

[Ben75] J.L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18:509–517, 1975.

[CGH+93] L. Paul Chew, Michael T. Goodrich, Daniel P. Huttenlocher, Klara Kedem, Jon M. Kleinberg, and Dina Kravets. Geometric pattern matching under Euclidean motion. In *Proceedings of the 5th Canadian Conference on Computational Geometry*, pages 151–156, Waterloo, ON, Canada, August 1993. University of Waterloo.

[DHU90] D.P. D.P. Huttenlocher and S. Ullman. Recognizing solid objects by alignment with an image. In *IJCV*, pages 195–212, 1990.

[DMML97] N. S. Netanyahu D. M. Mount and J. Le Moigne. Efficient algorithms for robust feature matching. In *Proceedings of the CESDIS Image Registration Workshop, NASA Goddard Space Flight Center (GSFC)*, pages 247–256, Greenbelt, MD, 1997. NASA Publication CP-1998-206853.

[HKR93] Huttenlocher, Klanderman, and Rucklidge. Comparing images using the hausdorff distance. *IEEETPAMI: IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15, 1993.

[HV97] M. Hagedoorn and R. Veltkamp. A general method for partial point set matching. In *Proceedings of the 13th International Annual Symposium on Computational Geometry (SCG-97)*, pages 406–408, New York, Jun 1997. ACM Press.

[IAH95] M. Irani, P. Anandan, and S. Hsu. Mosaic based representations of video sequences and their applications. In *International Conference on Computer Vision*, pages 605–611, 1995.

[KAI+95] R. Kumar, P. Anandan, M. Irani, J. Bergen, and K. Hanna. Representation of scenes from collections of images. In *Workshop on Representation of Visual Scenes*, March 1995.

[McM95] L. McMillan. Acquiring immersive visual environments with an uncalibrated camera. Technical Report TR95-006, University of North Carolina, 1995.

[MP94] S. Mann and R. W. Picard. Virtual bellows : Constructing high quality stills from video. Technical Report TR-259, M.I.T Media Laboratory, 1994.

[MP95] S. Mann and R. W. Picard. Video orbits of the projective group : A new perspective on image mosaicing. Technical Report TR-338, M.I.T Media Laboratory, 1995.

[Ruc97] W.J. Rucklidge. Efficiently locating objects using the hausdorff distance. *International Journal of Computer Vision*, 24:251–270, 1997.

[SK95] R. Szeliski and S.B. Kang. Direct methods for visual scene reconstruction. In *IEEE Workshop on Representation of Visual Scenes*, June 1995.

[Sze94] R. Szeliski. Image mosaicing for telereality applications. Technical Report CRL94/2, DEC-CRL, May 1994.

[Wol90] H.J. Wolfson. Model-based object recognition by geometric hashing. In *ECCV*, 1990.