Fast Motion Consistency through Matrix Quantization

Pyry Matikainen[1]; Rahul Sukthankar[2,1]; Martial Hebert[1]; Yan Ke[3]
[1]The Robotics Institute, Carnegie Mellon University
[2]Intel Research Pittsburgh
[3]Microsoft Corporation
{pmatikai,rahuls,hebert,yke}@cs.cmu.edu

Abstract

Determining the motion consistency between two video clips is a key component for many applications such as video event detection and human pose estimation. Shechtman and Irani recently proposed a method for measuring the motion consistency between two videos by representing the motion about each point with a space-time Harris matrix of spatial and temporal derivatives. A motion-consistency measure can be accurately estimated without explicitly calculating the optical flow from the videos, which could be noisy. However, the motion consistency calculation is computationally expensive and it must be evaluated between all possible pairs of points between the two videos. We propose a novel quantization method for the space-time Harris matrices that reduces the consistency calculation to a fast table lookup for any arbitrary consistency measure. We demonstrate that for the continuous rank drop consistency measure used by Shechtman and Irani, our quantization method is much faster and achieves the same accuracy as the existing approximation.

1 Introduction

The motion field is a fundamental feature that many algorithms use to analyze video. Determining whether two videos exhibit the same motion field is a key component in a wide range of applications such as event detection [7,8,10,15], action classification [3,4], human pose estimation [5], and occlusion boundary detection [18]. Motion consistency between videos has traditionally been computed by first calculating the optical flow of the two videos and then determining whether the flows match. Optical flow is difficult to compute accurately and is notoriously noisy on motion boundaries. Shechtman and Irani (S-I) recently demonstrated the feasibility of determining whether two videos could have been generated by the same motion field without explicitly calculating the optical flow [15]. However, many optimizations are required for the algorithm to run in real time because one must correlate a spatio-temporal volume through all locations in the video. Instead of representing the motion field as a real valued matrix as S-I do, we propose a method for quantizing the motion-based feature matrices into a set of discrete prototypes. By using these "motion words" our method dramatically reduces computation needed to calculate motion consistency with minimal loss in accuracy. Motion consistency calculations between two space-time patches are reduced to table lookups. Since naive quantizations of

space-time Harris matrices could introduce unacceptable errors, we perform a principled analysis of the space of these matrices to derive our quantization method.

Vector quantization is commonly employed to reduce the computation time required to search for the nearest neighbors of a vector or to directly look up the result of a function on that vector [11, 13, 14, 19]. The mapping of continuous feature spaces into a dictionary of prototypes results in more than just a computational speedup; it changes the way we view the problem and enables us to use fundamentally different approaches to solving the problem [6, 20]. For example, by quantizing SIFT features [12] into visual words, Sivic *et al.* were able to use text retrieval algorithms for image and video retrieval [16]. We observe that similar techniques can be applied to the motion domain and more specifically to the motion consistency measure proposed by S-I. Choosing the appropriate values to quantize and the right quantization function is a challenging process that demands a deeper understanding of both the algorithm and the space of matrices.

The original algorithm proposed by S-I computes the motion consistency between every point in the three dimensional template volume and the target video. While this is the most accurate method, it is also the slowest because the motion consistency needs to be recalculated at every location in the target video. Processing a $50 \times 25 \times 20$ query on a $144 \times 180 \times 200$ video using a 3 GHz Intel® Pentium®4 machine is 270 times slower than real-time [15]. By reducing the number of points at which to compute motion consistency, such as employing a hierarchical decomposition, processing only every other frame, and scanning only a few points in the template, S-I were able to reduce the computation time to near real-time (20 frames a second). However, we argue that orders of magnitude increases in the speed are still needed for the algorithm to work in production systems. We present a fast multi-level approach and demonstrate that it is sufficiently flexible to approximate S-I's method with a dramatic speedup.

2 Flow Consistency

Shechtman and Irani showed that by calculating the motion consistency between a template video and a test video, one can find actions such as diving, clapping, and spinning in real-world videos. This is done by finding the motion consistency between all points in the template video and the test video. The template is then scanned across all locations in both space and time, and all peaks where the consistency is greater than a specified threshold are flagged as containing the action of interest. Using a similar notation as S-I, we review the details of their algorithm. Let P be a small, *e.g.*, $7 \times 7 \times 3$ space-time patch in the video. We define the space-time gradient as $\Delta P_i = (P_{x_i}, P_{y_i}, P_{t_i})$ for each point in $P(i = 1 \ldots n)$. and we similarly define the space-time Harris matrix M as follows (see S-I [15] for further details):

$$M = \begin{bmatrix} \sum P_x^2 & \sum P_x P_y & \sum P_x P_t \\ \sum P_y P_x & \sum P_y^2 & \sum P_y P_t \\ \sum P_t P_x & \sum P_t P_y & \sum P_t^2 \end{bmatrix}. \quad (1)$$

We further define $M^{\diamondsuit}$ to be the upper left minor on M:

$$M^{\diamondsuit} = \begin{bmatrix} \sum P_x^2 & \sum P_x P_y \\ \sum P_y P_x & \sum P_y^2 \end{bmatrix}. \quad (2)$$

A space-time patch P contains multiple motions if there is a rank increase between $M^{\diamondsuit}$ and M, or a single motion if there is no rank increase. Because the local space-time

patches are small, we can assume that most patches have only one motion. Consider two space-time patches P_1 and P_2, where M_1 and M_2 denote their space-time Harris matrices, respectively. Determining whether the two patches have inconsistent motion is equivalent to determining whether the concatenated patch P_{12} has multiple motions. This is straightforward since $M_{12} = M_1 + M_2$. Ideally, one would like to calculate the rank-increase measure Δr, where

$$\Delta r = \text{rank}(M) - \text{rank}(M^{\diamondsuit}), \tag{3}$$

which one can do by calculating the number of non-zero eigenvalues of the matrices. Due to noise, the eigenvalues are never exactly zero, and therefore S-I defined a continuous rank-increase measure $\Delta\tilde{r}$ where

$$\Delta\tilde{r} = \frac{\lambda_2 \cdot \lambda_3}{\lambda_1^{\diamondsuit} \cdot \lambda_2^{\diamondsuit}} = \frac{\det(M)}{\det(M^{\diamondsuit}) \cdot \lambda_1} \approx \frac{\det(M)}{\det(M^{\diamondsuit}) \cdot \|M\|_F}, \tag{4}$$

and λ_i is the i^{th} largest eigenvalue of M. Since finding the eigenvalues of a matrix is time consuming, S-I approximate λ_1 by the Frobenius norm of M. The local *in*consistency measure is defined as

$$m_{12} = \frac{\Delta r_{12}}{\min(\Delta r_1, \Delta r_2) + \varepsilon}, \tag{5}$$

and its inverse, the consistency measure is defined as

$$c_{12} = 1/m_{12}. \tag{6}$$

To calculate the consistency for a template at a specific location in a video, the template is translated to that location and the pairwise consistencies between the template and video are summed over the entire template volume. To produce a consistency map, the consistency between the template and video is calculated for every possible location of the template. As calculating the full consistency map requires computing consistencies between all points in the template and video, this calculation is very expensive. Quantization of these matrices would enable us to avoid the calculations and look up the results directly. Additionally, precomputation allows us to use *any* flow consistency measure that is a function on M, rather than only those that can be efficiently computed.

3 Space of S-T Harris Matrices

We now analyze the space of S-T Harris matrices to gain a better understanding of its structure. This matrix has been studied in other contexts, for example in computing optical flow [1, 17] and space-time interest points [9]. As seen from Eqn. 1, S-T Harris matrices (M) are 3×3 symmetric positive-semidefinite matrices embedded within a six dimensional space. The basic structure of the matrices within this space is a polygonal cone in six dimensions, which we visualize using a cone in a three dimensional space (Figure 1). The positive semi-definite cone can be seen as a polygonal cone whose interior is composed of rank-3 matrices, whose faces are rank-2 matrices, and whose edges are rank-1 matrices. The cone interior represents matrices that are inconsistent by the rank increase measure, since the upper left minor must have rank at most two while the full matrix has rank three. The space outside the cone is composed of matrices which, while symmetric, are not positive semi-definite. The faces and edges may or may not be consistent, depending upon the ranks of their upper left minors ($M^{\diamondsuit}$). However, since the space of rank-1 and rank-0 2×2 matrices is a lower dimensional space than rank-2 3×3 matrices, inconsistent matrices occupy zero "volume" on the faces and edges of the cone.

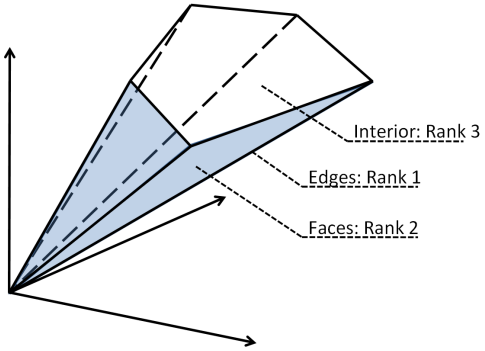


Figure 1: Space-time Harris matrices are 3×3 symmetric positive-semidefinite matrices, which we visualize using a cone. This conic structure suggests that the matrices could be quantized with predictable error bounds.

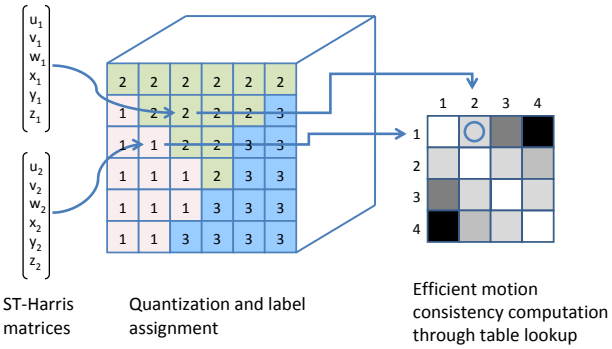


Figure 2: System overview. Given two patches and their corresponding ST-Harris matrices, we first quantize them and then assign a label according to the nearest prototype matrix. The labels are used to index into a pre-computed table that enables us to efficiently look up the motion consistency between the matrices.

4 Approach

Our approach to making the continuous rank-increase measure more efficient to compute is to look up $\Delta\tilde{r}$ directly given the space-time Harris matrices M. Given two space-time patches P_1 and P_2 and their corresponding space-time Harris matrices M_1 and M_2, we propose a two-level lookup table to find the motion consistency between the patches. We first convert the matrices into a small, finite set of prototype matrices so that that space of matrices can be represented using discrete labels. We then use the labels to look up the motion consistency between the patches. The overview of the algorithm is illustrated in Figure 2.

4.1 Matrix Quantization

The simplest approach to the problem would be to simply quantize matrices according to a uniform grid. Using n divisions on each axis of the six-dimensional grid, this results in n^6 prototypes. As the ultimate goal is to generate a lookup table for all possible *pairs* of prototypes, the consistency table would need n^{12} entries, which is impractically large for all but the coarsest grids. However, any more sophisticated method of assigning prototypes could be itself very expensive. To resolve both these issues, we propose a two-level quantization method in which a uniform grid is used to precompute the prototype assignments according to any arbitrary scheme and consistencies are precalculated between all pairs of the greatly reduced number of prototypes.

We first describe how we convert the space-time Harris matrices to their prototype matrices. Since the cone of symmetric positive semi-definite matrices is relatively small proportion of the entire matrix space, we can represent this space using a sparse set of prototypes.

Consider two space-time Harris matrices M_1 and M_2. The first lookup table is a finely-divided quantization table that takes M_i and returns a label l_i for its prototype matrix $\bar{M}_i$. The sparse set of prototype matrices $S = \{\bar{M}_1, \ldots, \bar{M}_p\}$ is chosen to maximize the coverage of the matrix space. Note that there is a one-to-one correspondence between l and $\bar{M}$ and there are p possible labels. We use l_1 and l_2 to index into the second consistency table to look up the precomputed motion consistency c_{12} between $\bar{M}_1$ and $\bar{M}_2$.

Consider a symmetric semi-positive definite matrix M. We first normalize it so that the L_∞ norm, $|M|_\infty$, has a maximum of 1.0 (described in Section 4.3) and call it $\hat{M}$:

$$\hat{M} = \begin{pmatrix} d_0 & f_0 & f_1 \\ f_0 & d_1 & f_2 \\ f_1 & f_2 & d_2 \end{pmatrix}. \tag{7}$$

After normalization, the entries in the matrix are in specific ranges such that the diagonals are in $d_i \in [0,1)$ and the non-diagonals are in $f_i \in (-1,1)$. The space of the matrix lies in a $(-1,1)^6$ hypercube. We quantize this space using D divisions per dimension. The quantization table Q is of size D^6 and each entry contains the prototype label for $\hat{M}$. The larger the number of divisions D, the more accurate the quantization is at the expense of a larger lookup table Q. The table can be indexed by quantizing the elements of $\hat{M}$ to D levels in the standard way ($\dot{d}_i = \lfloor d_i \cdot D \rfloor$, $\dot{f}_i = \lfloor \frac{f_i+1}{2} \cdot D \rfloor$). The label assigned to $\hat{M}$ is given by a simple table lookup

$$l = Q[\dot{d}_0, \dot{d}_1, \dot{d}_2, \dot{f}_0, \dot{f}_1, \dot{f}_2]. \tag{8}$$

We construct the quantization table as follows. Suppose $f(M)$ is some arbitrary assignment function from matrices to (labels of) prototype matrices. Recall that the quantization table Q has D^6 cells and each cell represents a small volume in the quantized matrix space $(-1,1)^6$. We build the quantization table so that, for a given cell center M_c, the cell's value is $f(M_c)$. In our implementation, the prototype matrices are randomly and uniformly distributed in the matrix space and we assign the label of M_c to its nearest neighbor in the set of of prototypes. The connected nature and relatively small dimensionality (6) of the space suggest that a uniform random distribution is likely to be near optimal. An alternative method of selecting prototypes would be to sample matrices from a set of training videos, but experiments suggest that the distribution of prototypes has only a minor effect.

4.2 Computing Motion Consistencies

We use the labels l_1 and l_2 to efficiently look up the motion consistencies between $\dot{M}_1$ and $\dot{M}_2$ in a precomputed table. This is an approximation of the motion consistency measure for the space-time patches P_1 and P_2 and their corresponding space-time Harris matrices M_1 and M_2. The precomputed consistency table C is a $p \times p$ matrix, where p is the number of prototype matrices. The motion consistency between M_1 and M_2 is given by

$$c_{12}(M_1, M_2) = C[l_1, l_2]. \tag{9}$$

We populate this table by calculating the consistency measure $c(\dot{M}_i, \dot{M}_j) = 1/m(\dot{M}_i, \dot{M}_j)$ (Eqn. 6) between all pairs of prototype matrices $(\dot{M}_i, \dot{M}_j) \in S^2$.

A further optimization could be used: if a given prototype produces the same (or nearly the same) consistency for any given comparison prototype, then these comparisons could be premptively avoided. For our selected consistency measure, however, the vast majority of prototypes do not have constant consistencies, so the improvement in a small minority of cases is outweighed by the additional overhead of testing for them.

4.3 Bounding Quantization Error

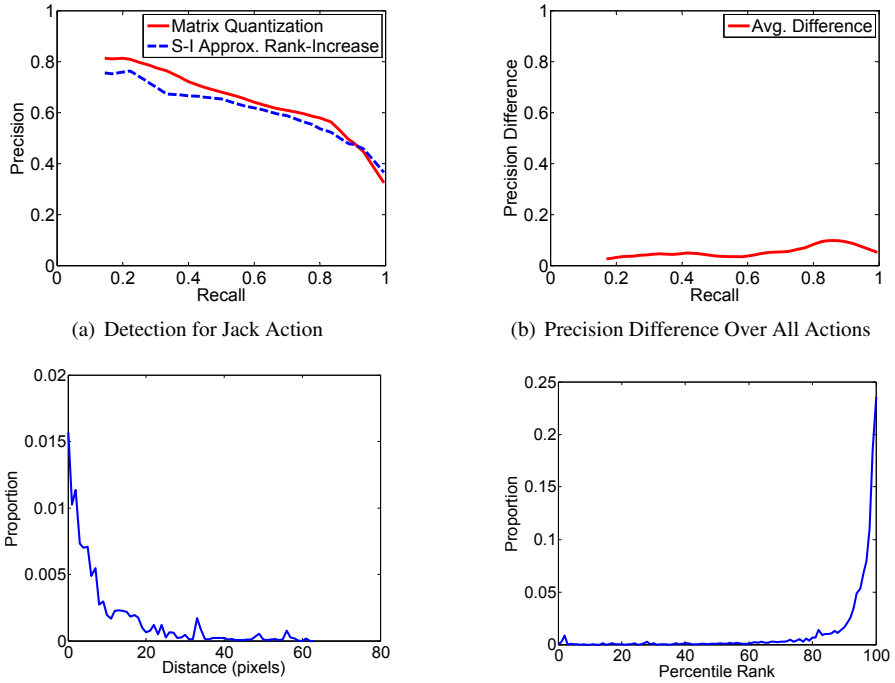
When a matrix M is discretized for lookup in the quantization table Q, we introduce an error that is proportional to the discretization step size. This may not be a problem if the dynamic range of the matrices in the data is small. Empirically, however, we observe over five orders of magnitude in difference between the largest and smallest matrices in our video data. To minimize the quantization error for small-normed matrices, a naive approach of finely discretizing the space would require 10^5 divisions per axis and a corresponding table with 10^{30} entries. A better approach is to use a hierarchy of nested quantization tables. This allows us to maintain a constant relative quantization error with only a linear (as opposed to exponential) increase in table size for the expected dynamic range of the matrices.

In practice, the hierarchical quantization table can be represented with only one table with a uniform grid size. Due to the recursive definition of the table, a sub-table at level i in the hierarchy is equal to the sub-table at level $i-1$ scaled by a ratio r. We normalize an input matrix M as follows. First, we find the hierarchy level of M, given by

$$k = \lceil \log_r |M|_\infty \rceil. \tag{10}$$

$\hat{M}$ is defined as $\hat{M} = M/r^k$, which ensures that $\frac{1}{r} < |\hat{M}|_\infty < 1.0$. Similarly, all prototype matrices would have norms between $\frac{1}{r}$ and 1.0 as well. The only exception is the zeroth prototype matrix representing the zero matrix. Once we have found the level k and computed $\hat{M}$, we look up label $\hat{l}$ from the quantization table Q as described in Section 4.1. Since the nearest neighbor to $\hat{M}$ could be from a prototype in a different scale, we also search the adjacent levels of $\hat{M}$. The assigned label l is the tuple $(\hat{l}, k)$. As the assignment of matrices to their prototypes is template-independent, this only needs to be done once as a preprocessing step.

We augment the motion consistency table C described in Section 4.2 as follows. Instead of storing motion consistencies between two prototype matrices of the same scale, it must be expanded for all pairs of prototype matrices in all scales. Given p prototype matrices and n levels, the expanded consistency table C would contain $(p \times n)^2$ entries.



(a) Detection for Jack Action

(b) Precision Difference Over All Actions

(c) Histogram of Distances for Films

(d) Histogram of Percentile Ranks for Films

Figure 3: Graphs in (a) and (b) compare matrix quantization (MQ) to S-I in an action recognition task. (c) compares the Euclidean distance between the per-frame most consistent locations of MQ and S-I. (d) considers how close the location of S-I's maximum is to a maximum in MQ; e.g., a percentile rank of 90 would indicate that the location of S-I's maximum in MQ is larger (more consistent) than 90 percent of the locations within the frame. In the ideal case, the histogram would have a single peak at 100.

5 Evaluation

Using the following set experiments, we show that the quantized S-T Harris matrix representation is equivalent to its continuous form when used to measure motion consistency. We also show that the computation time can be dramatically improved. The discrete representation can be used in conjunction with the optimizations that S-I proposed for further performance improvements. Note that we are not performing a full system evaluation for action recognition; we are instead studying the methods and effects of quantizing S-T Harris matrices in the context of computing motion consistency. We evaluate our method against S-I by both final results in a recognition task and by the raw consistency values both methods produce (see Figure 3).

For evaluation in a recognition task we evaluate our method on a subset of the Weizmann actions dataset [2]. Since template matching against flow does not produce recognition comparable to state of the art algorithms, we have chosen the simple Weizmann dataset in order to produce recognition rates high enough to effectively detect discrepancies between S-I and our method. The dataset contains 9 people performing 9 different actions. So as to be able to cross-validate across all nine individuals for a given template,

Figure 4: Sample frames from some of the test videos

we only use the five actions (jumping jacks, bend, one-handed wave, two-handed wave, and jumping in place) in which all individuals perform the action in the same direction (e.g., left or right). Each video clip contains one person performing one or more instances of an action, and we individually annotated all instances of the actions.

The relative performance of the quantization method and the S-I baseline are compared through cross-validation and averaging over all the trials. For each of the five actions, one instance of one person's action is used as a template to calculate the motion consistency on the remaining videos (from different people). The motion consistency maps are then used to generate precision-recall curves by computing a histogram of positive and negative motion consistency measures. The positive samples are computed by taking the local maximum around the manually labelled event, and the negative samples are chosen from random locations elsewhere. We plot the precision-recall curve for the "jumping jacks" action in Figure 3(a) and observe that the performance of the two methods are very similar. By averaging the difference in precision between the two methods for all actions, we summarize the results in Figure 3(b), where it can be seen that the difference in precision between the two methods is less than 0.10 for any recall rate.

In addition to the effect of quantization on the final detection rate, we investigate the effect of quantization on the raw template matching consistencies. We compare our method against S-I when run on approximately five hours of video, comprising three live-action films, stills of which can be seen in Figure 4. This represents over a trillion individual matrix-matrix consistency comparisons. In Figure 3(c) we compare the locations of the per-frame best consistencies by measuring the distance in pixels between the two locations. Depending on the choice of threshold, the mean distance ranges from 7 to 12 pixels. If we assume that S-I per-frame maximum is correct, we can also measure how close the same location is to a maximum when computed with our method. In particular, let the percentile rank of a point be the percentage of locations in the frame which are less consistent. If our method produced numerically identical results to S-I, the rank of the location of S-I's maximum in our calculation would always be exactly one. We consider the distribution of ranks in Figure 3(d).

The previous results were generated using a quantization table Q with a fixed set of parameters. Specifically, we quantized Q into 20 divisions per dimension and used a 16-level hierarchy with a scaling ratio of 2.1 between the levels. At each level, there are 100 prototypes for a total of (only) 1600 prototypes spanning the space of all S-T Harris matrices. We now study the effects of these parameters have on the performance. For the jumping in place action, Figure 5 shows how the performance across all actors varies as we increase the number of levels in the hierarchy. In general, more levels results in better performance, with diminishing returns after 8 levels. Performance varies little with the number of divisions per dimension or number of prototypes. Indeed, little degradation is seen until as few as 48 prototypes or 5 divisions per dimension are used, a surprising observation we plan to investigate in future work.

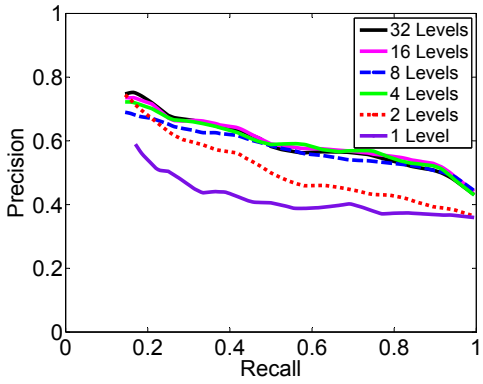


Figure 5: Increasing the number of levels in the quantization table hierarchy improves performance. We see diminishing returns after 8 levels.

Table 1: Timing results on a 4s video for different parts of the consistency measure computation. Our consistency measure (c) is over 30 times faster than S-I's baseline (a).

		Timing (sec.)	(σ)
(a)	S-I approx. consistency	92.70	(0.024)
(b)	Matrix quantization	2.07	(0.039)
(c)	Consistency lookup	2.75	(0.0087)

Finally, we compare the run-time performance of S-I's and our matrix quantized motion consistency measure. We calculate a consistency map for a $24 \times 74 \times 13$ template on a $180 \times 144 \times 124$ video (at half scale in both space and time) three times and measure the execution time of the algorithms (summarized in Table 1). A preprocessing step that is necessary for both algorithms is the spatio-temporal gradient computation needed to generate the elements of M. Because it is not dependent on the templates, it must only be computed once per video and thus we exclude it from our timing results. Our matrix quantization (line b) is also template-independent, and thus can be included in the preprocessing step. The part that is dependent and therefore scales linearly with the number of templates is S-I's approximate consistency measure (line a) and our consistency lookup (line c). As suggested in Section 1, a typical system might need to scan over 100,000 templates, and therefore these steps dominate the overall computation time. Thus, the 30-fold acceleration in motion consistency enabled by our method is a significant contribution for real-world video processing applications.

6 Conclusion

This paper proposes a method for quantizing space-time Harris matrices in the context of efficiently computing motion consistency in video. First proposed by Shechtman and Irani for event detection, correlating motion consistency has been incorporated as key components in a variety of applications in video processing. By analyzing the space of space-time Harris matrices, we observe that the space forms a hypercone that is amenable to quantization and reduction to a small set of prototypes. Our two-level quantization of

the matrices reduces the motion consistency calculations to a fast table lookup, resulting in a thirty-fold speedup over the baseline S-I baseline algorithm. Our basic matrix quantization approach generalizes to other domains but requires a deep understanding of the matrix space. In future work, we plan to analyze feature spaces in other popular computer vision algorithms to see whether our technique can provide similar benefits.

Acknowledgements

This research was supported in part by NSF Grant IIS0534962 and by the NSF Quality of Life Technologies ERC, Grant EEEC-540865.

References

[1] J. Bigün and G. Granlund. Optical flow based on the inertia matrix of the frequency domain. In *Proc. SSAB Symposium on Picture Processing*, 1988.

[2] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. In *ICCV*, 2005.

[3] A. F. Bobick and J. W. Davis. The recognition of human movement using temporal templates. *PAMI*, 23(3), 2001.

[4] A. Efros, A. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *ICCV*, 2003.

[5] A. Fathi and G. Mori. Human pose estimation using motion exemplars. In *ICCV*, 2007.

[6] B. Julesz. Textons, the elements of texture perception, and their interactions. *Nature*, 290:91–97, 1981.

[7] Y. Ke, R. Sukthankar, and M. Hebert. Efficient visual event detection using volumetric features. In *ICCV*, 2005.

[8] Y. Ke, R. Sukthankar, and M. Hebert. Event detection in crowded videos. In *ICCV*, 2007.

[9] I. Laptev and T. Lindeberg. Space-time interest points. In *ICCV*, 2003.

[10] I. Laptev and P. Perez. Retrieving actions in movies. In *ICCV*, 2007.

[11] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[12] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2), 2004.

[13] J. C. Niebles, H. Wang, and L. Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. In *BMVC*, 2006.

[14] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *CVPR*, 2006.

[15] E. Shechtman and M. Irani. Space-time behavior based correlation -or- how to tell if two underlying motion fields are similar without computing them? *PAMI*, 29(11), 2007.

[16] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, 2003.

[17] H. Spies and H. Scharr. Accurate optical flow in noisy image sequences. In *ICCV*, 2001.

[18] A. N. Stein and M. Hebert. Local detection of occlusion boundaries in video. In *BMVC*, 2006.

[19] T. Tuytelaars and C. Schmid. Vector quantizing feature space with a regular lattice. In *ICCV*, 2007.

[20] S.-C. Zhu, C.-E. Guo, Y. Wang, and Z. Xu. What are textons? *IJCV*, 62(1–2), 2005.