Interframe Coding of Canonical Patches for Mobile Augmented Reality

Mina Makar, Sam S. Tsai, Vijay Chandrasekhar, David Chen, and Bernd Girod
Information Systems Laboratory, Department of Electrical Engineering
Stanford University, Stanford, USA
{*mamakar, sstsai, vijayc, dmchen, bgirod*}*@stanford.edu*

***Abstract*—Local features are widely used for content-based image retrieval and augmented reality applications. Typically, feature descriptors are calculated from the gradients of a canonical patch around a repeatable keypoint in the image. In previous work, we showed that one can alternatively transmit the compressed canonical patch and perform descriptor computation at the receiving end with comparable performance. In this paper, we propose a temporally coherent keypoint detector in order to allow efficient interframe coding of canonical patches. In inter-patch compression, one strives to transmit each patch with as few bits as possible by simply modifying a previously transmitted patch. This enables server-based mobile augmented reality where a continuous stream of salient information, sufficient for the image-based retrieval, can be sent over a wireless link at the smallest possible bit-rate. Experimental results show that our technique achieves a similar image matching performance at $1/10$ of the bit-rate when compared to detecting keypoints independently frame-by-frame.**

***Keywords*-image matching; local features; augmented reality;**

I. Introduction

Many content-based image retrieval and augmented reality applications require the use of local image features. Examples of these robust local features include Scale-Invariant Feature Transform (SIFT) [1], Speeded Up Robust Features (SURF) [2] and Compressed Histogram of Gradients (CHoG) [3].

In an image matching framework, keypoints are detected from database and query images. These keypoints should be shift, scale and rotation invariant and also should be repeatably detectable in different images of the same scene. After keypoint detection, feature descriptors are calculated for every keypoint. Similarity of two descriptors is evaluated using a suitable distance metric such as the L_2 norm [4]. Many feature descriptors [1]–[4] share the common framework that the descriptor consists of histograms of gradients in a patch located around the detected keypoint. For scale and rotation invariance, patches are oriented with their maximum gradient along the same direction and resized according to the scale of the detected feature. We refer to these oriented and scaled patches as canonical patches.

For applications where data are transmitted over a network for feature detection and image matching, it is desirable that the amount of data sent is as low as possible to reduce the latency and the cost of the system. The emerging MPEG standard, Compact Descriptors for Visual Search (CDVS) [5] targets designing low bit-rate descriptors that achieve state-of-the-art image matching performance. In our previous work [6], [7], we present an initial study showing that one can alternatively transmit the compressed canonical patch and perform descriptor computation at the receiving end with comparable performance. This has the advantage of allowing interoperability between systems using different feature descriptors.

For mobile augmented reality applications, we extend the patch encoding idea to motion video. To exploit the temporal correlation in a video, we design a temporally coherent keypoint detector that allows efficient inter-patch compression. In inter-patch compression, we strive to extract canonical patches from the video that are temporally coherent and transmit each patch with as few bits as possible by simply modifying a previously transmitted patch. The goal is to enable server-based mobile augmented reality where a continuous stream of salient information, sufficient for image-based retrieval, can be sent over a wireless link at the smallest possible bit-rate.

The remainder of the paper is organized as follows. Section II explains the proposed temporally coherent keypoint detector including the patch matching technique and the improvements we achieve through using a similarity transform and an adaptive detection interval. Section III presents different patch encoding modes that exploit temporal correlation. In Section IV, we explain the idea of differential location coding. Finally, in Section V, we present experimental results showing the performance of the temporally coherent keypoint detector and predictive coding of patches and keypoint locations in terms of achieving a close image matching performance at more than $10\times$ bit-rate reduction compared to detecting keypoints independently frame-by-frame.

II. Temporally Coherent Keypoint Detector

As mentioned above, the first step in the extraction of local feature descriptors [1]–[4] on a still image is keypoint detection. These keypoints are represented by their location, orientation and scale and each keypoint corresponds to a canonical patch. We use either a 16×16 or 8×8 sampling grid to sample each canonical patch. To achieve temporal coherence, we need to track the patches over consecutive

978-0-7695-4875-3/12 $26.00 © 2012 IEEE
DOI 10.1109/ISM.2012.18

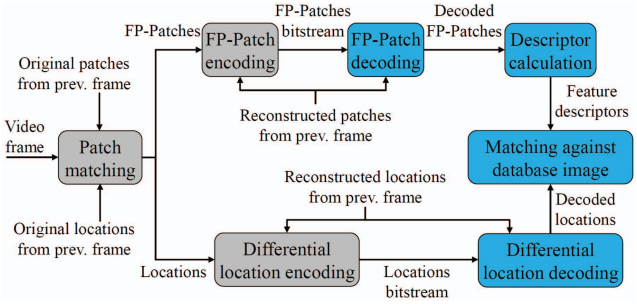

Figure 1. Image matching pipeline for FP-Patches. Blocks highlighted in gray represent the transmitter (camera phone) side and blocks highlighted in cyan represent the receiver (server) side.

video frames. Tracking of interest points in a video has been studied extensively, e.g., KLT tracker and Normalized Cross Correlation (NCC) tracker [8]. Our work introduces the idea of performing the tracking on the canonical patch level. We propose performing canonical *patch matching* for efficient patch tracking.

A. Patch Matching with Displacement Model

We divide the video frames into two categories: *Detection frames* or *D-frames* and *Forward Propagation frames* or *FP-frames*. In D-frames, we run a keypoint detection algorithm to detect the interest points in the video frame. In this paper, we use the SIFT keypoint detector [1] which relies on maxima detection in a difference of Gaussian (DoG) scale space representation of the frame. The patches corresponding to the keypoints detected by running a conventional detection algorithm are referred to as D-Patches.

In FP-frames, patch tracking is performed and each patch is connected with a patch from the previous frame. We refer to these patches as Forward Propagation Patches or FP-Patches because they result from the propagation of the patches of the previous frame. Fig. 1 presents a block diagram of the proposed image matching pipeline for FP-Patches. To detect FP-Patches, we do not run our keypoint detection algorithm again. Instead, for each patch in the previous video frame, we search for a corresponding patch in the current frame that minimizes a distortion measure. In this paper, we use the Sum of Absolute Differences (SAD) as a distortion measure. To search for the minimum distortion patches, we apply patch matching as follows.

Each patch in the previous frame is a function of its location, orientation and scale. We start from the location of the patch in the previous frame and then define a search range to search for the most correlated patch in the current frame. Fixing the orientation and scale, we vary the x and y values in small steps around the starting location until we cover the whole search range. For each value of x and y, we extract the corresponding patch in the current video frame and measure its SAD distortion from the patch of the previous frame. The patch with the minimum SAD is selected to be the tracked FP-Patch in the current video frame. In this version of the algorithm, an FP-Patch simply inherits orientation and scale from the initial D-Patch.

Fig. 2 illustrates the patch matching technique. It is similar to the idea of block matching used in video coding but applied on canonical patches. In Fig. 2(a), the magenta circle near the letter **A** corresponds to the x and y locations of a patch in frame $n-1$ in *Reba* test sequence. The size and rotation angle of the square define the scale and orientation of this patch respectively. In Fig. 2(b), we search among candidate patches in frame n, with the same scale and orientation and varying x and y in a defined search range, for the patch with minimum SAD. The patch defined with a green square minimizes the distortion and thus selected as the FP-Patch corresponding to the patch in Fig. 2(a).

In our experiments, we vary the x and y values from -24 to 24 in steps of 2. We find that this search range is sufficient for video resolutions with good image matching. Please note that patch matching searches in the scale space representation with the same scale of the keypoint from the previous frame. This results in scale-adaptive search accuracy since the variation in x and y is multiplied by the scale factor when choosing the layer in the DoG pyramid corresponding to the scale of the keypoint under search.

A finer search stage with smaller variations in x and y can follow the initial search stage once we obtain an initial good estimate of the matching patch. This is similar to how sub-pixel motion estimation is defined in H.264 video coding standard [9]. After settling on the best patch in varying steps of 2, we test more patches around the initial matching patch estimate, these patches correspond to varying x and y locations between -1.75 and 1.75 in steps of 0.25 around the x and y locations of the best initial guess to fine-tune the matching patch to quarter pixel accuracy. Although this finer search only improves the image matching performance by little, it reduces the prediction residual between the tracked patches which results in bit-rate reduction for inter-patch coding as described in Section III.

In some cases, we cannot find a good FP-Patch to track a patch from the previous frame. This happens when a keypoint in the previous frame is no longer relevant and results in a large value for the minimum SAD during patch matching. We solve this problem by terminating the patch in the previous frame and reducing the number of patches in the current FP-frame. If the SAD value of a matching patch is larger than a threshold SAD_{term}, we signal a termination flag for the patch in the previous frame. Patch termination reduces the patch transmission bit-rate. It also helps to remove unnecessary patches which may lead to more outliers during the geometric consistency check of the matching features.

After tracking all patches from frame $n-1$ to frame n, the keypoint locations in frame n are used as a starting point and are tracked to frame $n+1$ and so on. Every

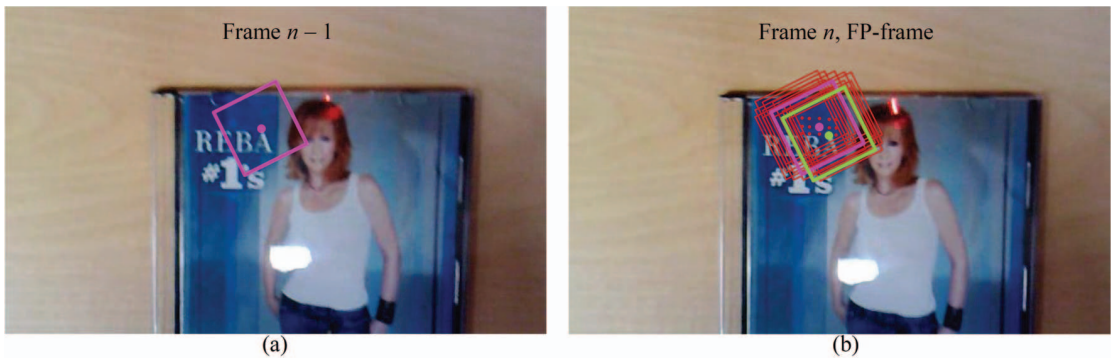

Figure 2. Illustration of patch matching. (a) Example patch in frame $n-1$, to be tracked in frame n. (b) Patch matching searches for the patch of minimum SAD with respect to the patch in (a). The patches are defined by fixing the orientation and scale and varying the location to cover the search range. The patch highlighted with a green square represents the result of patch matching.

few frames, we find that the tracked patches start to deviate from the actual interest points that they correspond to. This is because of the assumption that the patches only move with translational motion in a specific search range, while practical videos may contain more complex motion. Thus, after a pre-defined number of frames, we insert a D-frame by running the keypoint detector instead of tracking patches from the previous frame. We refer to the number of frames between consecutive D-frames as the *detection interval* (DI).

B. Patch Matching with Similarity Transform

In Section II-A, we only vary the x and y locations of the patches during patch matching. In the later image matching stage, we use keypoint locations for a geometric consistency check and the update in the locations of the keypoints between frames is needed to accurately localize the object of interest in a particular video frame. We observe that the tracking fails where there is complex motion in the video, and the result is a large drop in the number of feature matches. This problem can be mitigated by using a short detection interval but this in turn causes a large increase in the patch transmission bit-rate.

We introduce a more sophisticated motion model to accurately track over longer detection intervals. Since each patch is a function of its location, orientation and scale, we vary orientation and scale along with location during patch matching. Thus, we assume a similarity transform motion model with four degrees of freedom between corresponding patches in consecutive video frames. Affine motion models have been explored before in motion estimation for video coding, see, e.g., [10]. Although these models only result in a minor bit-rate reduction in video coding, we observe that for our problem, they introduce a significant gain in the tracking and image matching performance especially when using a large detection interval and during periods of camera rotation around the optical axis, looming motion and zoom.

To decide the search range for orientation and scale values, we perform the following experiment: We extract D-Patches from all the video frames in a training set and try to match the descriptors for each two consecutive video frames. We study the changes in orientation and scale between each two matching keypoints to indicate a reasonable search range for orientation and scale in patch matching. We generally find that the changes in orientation and scale are small between consecutive frames and a small search range with few search values is enough.

We apply a brute force search for all different combinations of varying x, y, orientation and scale. Varying orientation and scale during patch matching generally improves the performance. However, we observe that using the 4-parameter model during the initial search phase, that includes the whole search range for locations, sometimes hurts the geometric consistency check which only considers keypoint locations. We achieve the best image matching results when we include the orientation and scale search only during the fine location search which considers a 2×2 search range for x and y.

C. Adaptive Detection Interval

An advantage of the proposed patch matching technique is avoiding the computational complexity of performing keypoint detection on FP-frames. However, a fixed detection interval may be inefficient when the tracked FP-Patches are no longer well representing the objects of interest in a particular video frame. This happens when there are occlusions or scene-cuts or when objects of interest are entering or leaving the view, which results in wrong matching decisions until the next D-frame. To solve this problem we propose an adaptive detection interval where we insert a D-frame whenever patch matching deteriorates.

We acquire both types of patches for each video frame. D-Patches are acquired through running the keypoint detector and FP-Patches are acquired through patch matching. We measure the deviation between the D-Patches and the FP-Patches and refer to the deviation measure as *D-FP*$_{SAD}$. A small deviation means that the FP-Patches are well representing the scene; thus, we keep the FP-Patches and decide an FP-frame. A large deviation indicates that the

D-Patches contain information that is not well captured by the FP-Patches and hence, we decide a D-frame. The deviation measure is calculated as follows.

The first frame in the video is always a D-frame. For each new frame, we calculate both FP-Patches and D-Patches. For each patch in the FP-Patches we search for a corresponding patch in the D-Patches with the minimum SAD. The deviation measure *D-FP*$_{SAD}$ for a certain frame equals the average SAD between the FP-Patches and their closest corresponding D-Patches. A threshold *D-FP*$_{th}$ is defined. If *D-FP*$_{SAD}$ is smaller than the threshold, we decide an FP-frame; otherwise, we decide a D-frame. Once the frame type is decided, the other type of patches is discarded. To avoid the build-up of small deviations, we define another threshold *FP*$_{max}$ for the maximum number of consecutive FP-frames. If the number of consecutive FP-frames reaches this threshold, the next frame is decided to be a D-frame.

III. Patch Encoding

We use our own Patch ENCoder (PENC) [7] to encode both D-Patches and FP-Patches. PENC has a very similar structure to JPEG [11]. However, PENC applies a preprocessing step of Gaussian blurring and mean removal on the patches and always uses a 2D DCT transform with the same size of the patch in order to avoid producing blocking artifacts within the patch. Depending on the patch size, there are two variants of PENC for 8×8 patches and 16×16 patches. PENC uses Huffman tables that are trained for patch statistics which are different from natural image statistics. Huffman tables are trained using the statistics of the patches of 5000 images from the distractor dataset used for MPEG CDVS [12] evaluation.

A. Coding of FP-Patches

Instead of encoding the FP-Patches independently, we apply predictive coding to lower the bit-rate. From the patch matching stage we know the patch correspondences between consecutive video frames and hence, each patch is predicted from the corresponding patch in the previous frame and only the residual signal between patches is encoded using PENC. In Section II-A, we mention that the original patches are always used in patch matching. However, during the encoding stage, the patches of the current frame are predicted from the reconstructed patches of the previous frame (see Fig. 1) to avoid any drift between the encoder and the decoder. We refer to this encoding mode as *Predictive Coding* or *P* mode.

We observe that the residual energy between the FP-Patches of two consecutive frames is usually less than the residual energy between a patch in a video frame and the matching patch in the clean image containing the object of interest. This suggests that in many mobile augmented reality applications, it is not necessary to transmit the prediction residuals in FP-frames. We update the keypoint locations, orientation and scale and still use the descriptors extracted from the patches of the previous D-frame after removing the descriptors corresponding to terminated patches. This has the advantage of significantly lowering the transmission bit-rate with a minor effect on the image matching performance. We refer to this encoding mode as *Skip* or *S* mode.

B. Coding of D-Patches

The first D-frame is always encoded independently which means that the actual patches are passed to PENC and the bitstream that is generated is decodable without the need of any extra information. We refer to this encoding mode as *Intra Coding* or *I* mode. For the following D-frames, we compare two encoding methods. First, these D-frames can also be encoded independently using *I* mode.

The second mode for encoding D-frames is *Predictive Coding with Patch Selection* or *PS* mode. Although the D-Patches are generated by running the keypoint detector independently and there is no one-to-one correspondence to the patches of the previous frame, we still can search for a patch in the previous reconstructed frames that minimizes some distortion measure (SAD). For correct decoding, the residual of the D-Patch is transmitted along with the patch number that is used for the prediction. There are two different options in the *PS* mode depending on the frame we use for patch selection. In the first option, D-Patches are predicted from the reconstructed FP-Patches of the previous frame. We refer to this mode as *PS*$_P$ mode where the subscript *P* highlights the use of the *previous* frame for patch selection. The second option involves using the previous D-frame for patch selection. This enables the application of the *Skip* mode for encoding the FP-Patches (Section III-A) as there is no need to reconstruct the previous FP-Patches in order to decode the current D-Patches. We refer to this mode as *PS*$_D$ mode where the subscript *D* highlights the use of the previous *D-frame* for patch selection.

Different encoding schemes are constructed by combining the proposed encoding modes for D-Patches and FP-Patches. The most bit-rate efficient encoding scheme is combining the *PS*$_D$ mode for D-Patches with the *S* mode for FP-Patches. In our experiments, we compare the two encoding schemes presented in Fig. 3 to illustrate the fact that a good image matching performance is still achieved when using the *Skip* mode and sending no residuals for FP-Patches.

IV. Differential Location Coding

In *Skip* mode (Section III-A), we avoid the transmission of the patch residuals of FP-Patches and use the descriptors extracted from the patches of the previous D-frame. However, in augmented reality applications, we need to track the object of interest and localize it in all video frames. Hence, we need to transmit the new locations for the keypoints detected in the FP-frames. Some applications may also need the transmission of the orientation and scale of these keypoints.

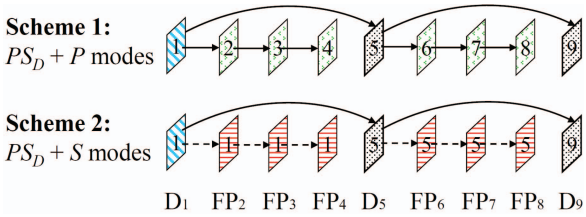

Figure 3. Example patch encoding schemes. Each scheme combines an encoding mode for D-Patches and another for FP-Patches. The first D-frame is always encoded using *I* mode. We use solid or dashed arrows to indicate whether residuals are encoded or not. Curved arrows indicate D-frame prediction with patch selection.

For the keypoint locations of the D-Patches, we use the location coder developed by Tsai *et al.* [13], [14]. Since the keypoint locations for FP-Patches are predicted from the corresponding keypoint locations of the previous frame, it is more efficient to only transmit the difference between the keypoint locations of corresponding patches in consecutive frames. Measuring the statistics of the location differences between consecutive video frames, we sometimes observe a global motion for the object of interest. This results in a probability distribution of the difference values that follows a Laplacian distribution centered on the global shift value. We refer to the global shift values in x and y directions as $G_{sh,x}$ and $G_{sh,y}$ respectively. We propose a differential location coder that works as follows.

First, we measure the differences between the original keypoint locations of the current FP-frame and the reconstructed corresponding keypoint locations of the previous frame. These location differences are quantized using a quantization step Q_{loc} in both x and y directions. Second, we measure the histograms of the quantized location differences in x and y in the current frame, and estimate the values of $G_{sh,x}$ and $G_{sh,y}$ from the peaks of these histograms. The values of $G_{sh,x}$ and $G_{sh,y}$ are encoded separately using a signed exponential Golomb code where the symbols 0, 1, -1, 2, -2, 3, $-3\ldots$ are encoded using the codewords 1, 010, 011, 00100, 00101, 00110, $00111\ldots$ and so on.

The last step in the differential coding of locations is the subtraction of $G_{sh,x}$ from the quantized location differences in x direction and $G_{sh,y}$ from the quantized location differences in y direction. The quantized location differences after removing the global shift are encoded using an arithmetic encoder similar to the encoder used in [13], [14]. The encoded locations need to be decoded back at the transmitter side in order to be used in the predictive coding of the keypoint locations of the following FP-frame.

To measure the effect of location quantization on the image matching performance, we use RANSAC [15] for geometric consistency check and investigate two different metrics: the number of feature matches post-RANSAC and the localization accuracy. The localization accuracy is measured by calculating the *Jaccard* index between the ground truth location of the object of interest and the projected location that is inferred after matching. We manually generate the ground truth location by defining a bounding quadrilateral around the object of interest in each video frame. From RANSAC results, we project the vertices of the clean image on each video frame. The Jaccard index is defined as the ratio between the area of overlap between the ground truth and the projected quadrilaterals, and the area of the union of the two quadrilaterals. Hence, Jaccard index values close to 1 indicate better localization.

For applications where we need to transmit orientation and scale, we also apply differential coding by transmitting differences from the previous frame. Since we search few values for orientation and scale (Section II-B), we do not apply further quantization on the encoded values. The statistics of the difference values in orientation and scale usually follow a Laplacian distribution and thus, we use a signed exponential Golomb code to encode them. For example, if we allow the five values -0.1, -0.03, 0, 0.03 and 0.1 for the differences in orientations, then the corresponding codewords are 00101, 011, 1, 010 and 00100 respectively.

V. Experimental Results

We perform experiments on four VGA size videos: *Fogelberg*, *AliciaKeys*, *Reba* and *AnneMurray*. All these videos contain a CD cover recorded with a hand-held mobile phone with different amounts of camera motion, glare, blur, zoom, rotation and perspective changes. Each video contains 200 frames recorded at 25 fps. The frames of each video sequence are matched against a still image of the corresponding clean CD cover with dimensions 500×500. Fig. 4 shows an example frame of each video and the matching clean CD cover. The videos and the clean images can be downloaded from [16].

A. Performance of Temporally Coherent Detection

To minimize the bit-rate, we only transmit patches corresponding to features that are more likely to match. We use the same feature selection algorithm as in the CDVS Test Model (TM) [17] and we set the maximum number of extracted patches from a video frame to 200. This feature selection algorithm chooses the features with higher matching probabilities based on their scale, orientation, output of the DoG pyramid and distance from the image center. First, we study the feature matching performance of 16×16 patches and 8×8 patches without applying any quantization to the patches or the keypoint locations. This is to illustrate the performance of the proposed patch matching technique and its improvements as explained in Section II.

Fig. 5 compares the image matching performance for different proposed techniques and patch sizes. The figure shows the plots for *Fogelberg* and *AliciaKeys* video sequences where the x-axis represents the frame number and

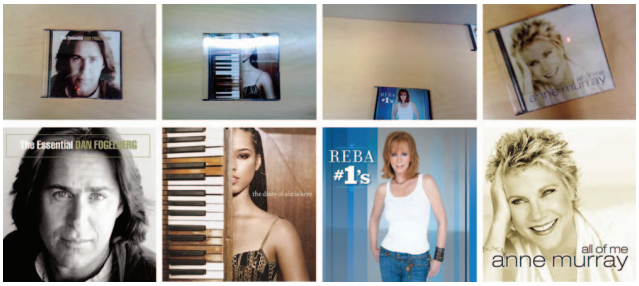

Figure 4. Example frames from the video sequences used in the experiments (upper row) and their corresponding matching clean CD cover (lower row). Sequences from left to right are *Fogelberg*, *AliciaKeys*, *Reba* and *AnneMurray*.

the y-axis represents the number of feature matches post-RANSAC. Figs. 5(a) and 5(b) show the results for 16×16 patches and Figs. 5(c) and 5(d) show the results for 8×8 patches. Note that to perform matching, we first upsample the patches to size 32×32 using bilinear interpolation and then calculate SIFT descriptors on the upsampled patches.

Each plot compares four different curves. First, we plot the number of matches if we run the SIFT keypoint detector every frame; i.e., all frames are D-frames. This serves as a benchmark for other techniques as our goal is to obtain a matching performance close to this curve when applying the temporally coherent detector. The second curve uses a short DI of 5 frames and only searches keypoint locations as in Section II-A. The third curve uses a longer DI of 20 frames and searches the same locations as the second curve. The last curve includes 5 orientation differences (-0.1, -0.03, 0, 0.03 and 0.1) and 5 scale differences (-0.5, -0.15, 0, 0.15 and 0.5) in the search range (Section II-B). For patch termination, we use $SAD_{term} = 1800$ for 16×16 patches and $SAD_{term} = 450$ for 8×8 patches.

If only searching keypoint locations, we find that a short DI maintains a good matching performance compared to the DI $= 1$ case. However, a large drop in matching is observed when increasing DI to 20. Adding orientation and scale search for a similarity transform motion model helps reducing most of the performance drop and results in an acceptable matching performance. An example drawback of using a fixed DI is observed at frames 22 to 40 from *AliciaKeys*. Although the performance of DI $= 1$ improves at frame 26, the performance of the other curves does not improve until the next D-frame.

Comparing different patch sizes, we see that reducing the size from 16×16 to 8×8 causes about 15% drop in performance for the uncompressed patches case. However, using a smaller patch size helps maintaining a better patch quality when applying a bit-rate constraint. We also observe that nearly no patches are terminated in *Fogelberg* due to simple motion, while *Reba* is the sequence with the largest number of terminated patches because of more complex motion and a part of the CD cover sometimes leaves the scene as well.

Fig 6 uses *Reba* and *AnneMurray* video sequences and presents the results of the adaptive detection interval idea described in Section II-C with *D-FP*$_{th} = 1800$ for 16×16 patches or 450 for 8×8 patches, and $FP_{max} = 29$. These parameters are chosen to obtain a reasonable number of D-frames that significantly reduces the bit-rate but still maintains a good matching performance. Comparing the performance to the case of fixed DI, we find that the performance of the adaptive DI technique better follows the DI $= 1$ case. During periods when the use of a fixed DI causes a large drop in matching (frames 71 to 80 and 169 to 180 from *Reba* and frames 165 to 173 and 195 to 200 from *AnneMurray*), we observe that the adaptive DI technique solves this problem by inserting D-frames and results in a larger number of feature matches.

B. Effect of Patch Encoding

In Fig. 7, we start from the matching results of the uncompressed patches for the adaptive DI case and use PENC to compress the patches with different schemes illustrated in Fig. 3. We compare Schemes 1 and 2 and use a medium value of QP $= 32$ as a compromise between the matching accuracy and bit-rate. We find that there is a consistent drop in the number of feature matches due to compression; however, this drop is small at the chosen compression quality. Scheme 2 performs very close to Scheme 1 and it is characterized by having less fluctuations in the number of matches between consecutive frames since it is using the exact same descriptors over the whole detection interval and only changing the keypoint locations during RANSAC.

Table I shows the bit-rates for all the video sequences and the average number of feature matches post-RANSAC for each scheme. We compare the adaptive DI case with encoding schemes 1 and 2 to the DI $= 1$ case where the D-Patches are encoded using the PS_D mode. We also state the number of D-frames for each video sequence using the above-mentioned values for *D-FP*$_{th}$ and FP_{max}. Scheme 2 results in a very large bit-rate reduction with a matching performance comparable to both Scheme 1 and DI $= 1$ case. With Scheme 2, we obtain bit-rate savings of about $7\times$ compared to Scheme 1 or more than $10\times$ compared to DI $= 1$ case. We target a bit-rate of few tens of kbps, reasonable for practical transmission systems. We achieve acceptable matching results at this bit-rate by using patch termination, small size patches, adaptive DI with large FP_{max} and combining PS_D and S encoding modes.

C. Effect of Location Coding

The results reported in the previous subsections use unquantized locations. We quantize and encode locations as described in Section IV. Statistics from *JanetJackson* sequence are used to train the arithmetic coder for differential

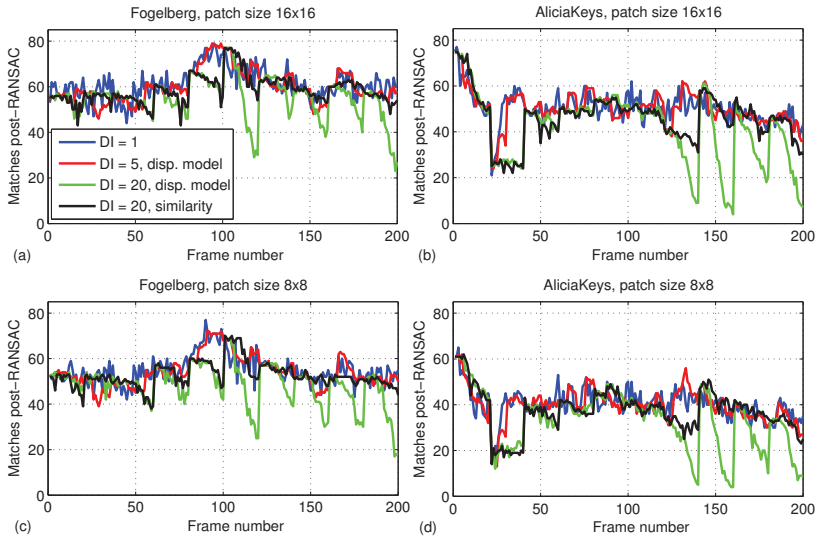

Figure 5. Matching performance of the proposed patch matching technique. We compare running the keypoint detector every frame (DI $= 1$) to performing patch matching with different detection intervals. '**disp. model**' refers to only varying x and y in the search range while '**similarity**' refers to including the keypoint orientation and scale in search as described in Section II-B.

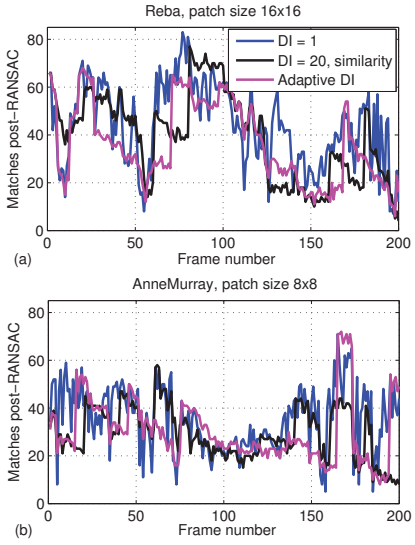

Figure 6. Matching performance of patch matching technique when using a fixed versus an adaptive detection interval. Using an adaptive DI better follows the performance of DI $= 1$ case.

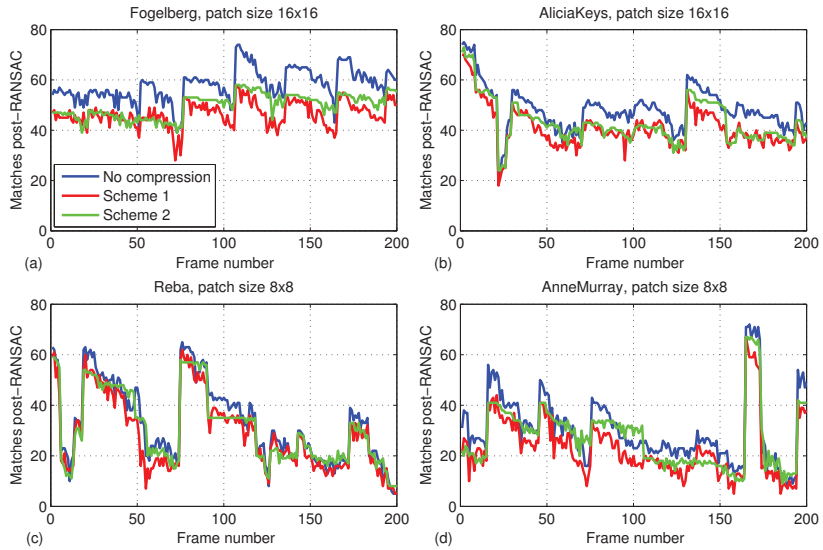

Figure 7. Matching performance after inter-patch encoding. The figure uses an adaptive DI and compares using descriptors extracted from original patches against those extracted from encoded patches. Encoding schemes 1 and 2 are compared. Scheme 2 achieves a large bit-rate reduction while keeping a reasonable matching performance.

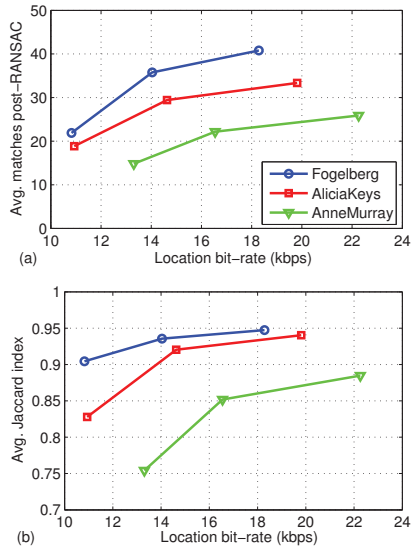

Figure 8. Matching performance and localization accuracy after location coding. $Q_{loc} = 2, 4$ and 8 values are used and a large drop at $Q_{loc} = 8$ is observed.

location coding. This sequence is available in [16] and is not used for testing. We fix all the patch encoding parameters to obtain the results in Table I (Scheme 2, 8×8 patches) and vary Q_{loc} for different location bit-rates. Fig. 8(a) presents the relation between the location bit-rates and the average number of feature matches post-RANSAC for *Fogelberg*, *AliciaKeys* and *AnneMurray* sequences. We use the same Q_{loc} for both D-frames and FP-frames and vary the bit-rates by varying $Q_{loc} = 2, 4$ and 8. $Q_{loc} = 4$ results in good matching accuracy at an acceptable location bit-rate.

Finally, in Fig. 8(b), we compare the Jaccard index at different location bit-rates for the same video sequences. The Jaccard index is averaged over all the frames in the video sequence. The localization accuracy drops with coarser

Sequence, patch size	DI = 1		Adaptive DI				
	Bit-rate (kbps)	Avg. matches	Bit-rate (kbps) Scheme 1	Avg. matches Scheme 1	Bit-rate (kbps) Scheme 2	Avg. matches Scheme 2	Number of D-frames
Fogelberg, 16×16	376.1	50.1	186.3	46.48	21.13	49.66	8
Fogelberg, 8×8	318.57	47.38	172.97	42.1	15.04	42.1	7
AliciaKeys, 16×16	478.7	42.7	263.9	40.84	51.5	42.94	15
AliciaKeys, 8×8	370.4	36.23	211.54	32.8	27.78	34.35	11
Reba, 16×16	429.74	35.6	227.26	27.4	39.7	28.7	14
Reba, 8×8	343.64	34.15	186.88	28.8	25.25	30.87	11
AnneMurray, 16×16	394.9	29.6	209.35	23.6	23.68	26.9	9
AnneMurray, 8×8	346.3	28.5	197.3	23.1	17.46	27.0	8

Table I
BIT-RATE AND MATCHING PERFORMANCE FOR PATCH ENCODING.

quantization of keypoint locations and $Q_{loc} = 4$ results in acceptable compromise between bit-rate and accuracy. At this quantization step, the average bit-rate used for encoding a keypoint location in a D-frame $= 6.85$ bits/location. However, the average bit-rate for a location in an FP-frame $= 2.886$ bits/location. This indicates that differential encoding of keypoint locations results in 58% bit-rate reduction when compared to non-differential methods [14].

VI. CONCLUSIONS

We present an efficient method for obtaining canonical image patches that are temporally coherent to exploit the temporal correlation in videos used in mobile augmented reality applications. We also propose methods for encoding these canonical patches and their corresponding keypoint locations using efficient predictive coding techniques.

Experimental results show that the proposed patch extraction mechanism results in a matching performance similar to the oblivious detection of new patches every video frame. Our patch encoding and differential location coding techniques achieve a substantial bit-rate reduction when compared to the bit-rates of encoding non-coherent patches. We achieve a $10\times$ bit-rate reduction and still maintain an acceptable matching performance and this facilitates building practical streaming mobile augmented reality systems.

REFERENCES

[1] D. Lowe, "Distinctive Image Features From Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91-110, Nov. 2004.

[2] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded Up Robust Features," *European Conference on Computer Vision*, Graz, Austria, May 2006.

[3] V. Chandrasekhar *et al.*, "CHoG: Compressed Histogram of Gradients," *IEEE International Conference on Computer Vision and Pattern Recognition*, Florida, USA, June 2009.

[4] B. Girod *et al.*, "Mobile Visual Search," *IEEE Signal Processing Magazine*, vol. 28, no. 4, pp. 61-76, July 2011.

[5] B. Girod, V. Chandrasekhar, R. Grzeszczuk, and Y. Reznik, "Mobile Visual Search: Architectures, Technologies, and the Emerging MPEG Standard," *IEEE Multimedia*, vol. 18, no. 3, pp. 86-94, July-September 2011.

[6] M. Makar, C.-L. Chang, D. Chen, S. Tsai, and B. Girod, "Compression of Image Patches for Local Feature Extraction," *IEEE International Conference on Acoustics, Speech and Signal Processing*, Taipei, Taiwan, April 2009.

[7] M. Makar, H. Lakshman, V. Chandrasekhar, and B. Girod, "Gradient Preserving Quantization," *IEEE International Conference on Image Processing*, Florida, USA, Sept. 2012.

[8] G. Takacs, "Unified Tracking and Recognition with Rotation-Invariant Fast Features (chapter 2)," *Ph.D. dissertation, EE Dept., Stanford University*, Stanford, USA, Feb. 2012.

[9] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC Video Coding Standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol.13, no.7, pp. 560-576, July 2003.

[10] H. Lakshman, H. Schwarz, and T. Wiegand, "Adaptive Motion Model Selection Using a Cubic Spline Based Estimation Framework," *IEEE International Conference on Image Processing*, Hong Kong, Sept. 2010.

[11] ITU-T and ISO/IEC JTC1, "Digital Compression and Coding of Continuous-Tone Still Images," *ISO/IEC 10918-1 - ITU-T Recommendation T.81*, Sept. 1992.

[12] Y. Reznik, G. Cordara, and M. Bober, "Evaluation Framework for Compact Descriptors for Visual Search," *ISO/IEC JTC1/SC29/WG11/N12202*, July 2011.

[13] S. Tsai, D. Chen, G. Takacs, V. Chandrasekhar, J. P. Singh, and B. Girod, "Location Coding for Mobile Image Retrieval," *International Mobile Multimedia Communications Conference*, London, U.K., Sept. 2009.

[14] S. Tsai *et al.*, "Improved coding for image feature location information," *SPIE Workshop on Applications of Digital Image Processing*, San Diego, USA, August 2012.

[15] M. Fischler, and R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting," *Communications of ACM*, vol. 24, no. 6, pp. 381-384, June 1981.

[16] http://www.stanford.edu/people/mamakar/ISM2012

[17] G. Francini, S. Lepsoy, and M. Balestri, "Description of Test Model under Consideration for CDVS," *ISO/IEC JTC1/SC29/WG11/N12367*, Dec. 2011.