Feature Tracking for Mobile Augmented Reality Using Video Coder Motion Vectors

Gabriel Takacs[1+] Vijay Chandrasekhar[1+] Bernd Girod[+] Radek Grzeszczuk[*]

[+]Information Systems Laboratory, Stanford University [*]Nokia Research Center, Palo Alto

ABSTRACT

We propose a novel, low-complexity, tracking scheme that uses motion vectors directly from a video coder. We compare our tracking algorithm against ground truth data, and show that we can achieve a high level of accuracy, even though the motion vectors are rate-distortion optimized and do not represent true motion. We develop a framework for tracking in video sequences with various GOP structures. Such a scheme would find applications in the context of Mobile Augmented Reality. The proposed feature tracking algorithm can significantly reduce the required rate of feature extraction and matching.

1 INTRODUCTION

Many researchers think of augmented reality in the context of see-through goggles. Such a system is suitable for Augmented-Reality enhanced brain surgery or combat. However, for consumer use, mobile handheld devices, such as camera phones or PDAs, will be more ubiquitous and socially acceptable. Mobile devices already generate 3D graphics and capture/display live video. New GPS-enabled models are entering the market with an accuracy of at least a few tens of meters. Wireless Internet access, albeit at moderate speeds and substantial latencies, is now universally provided by cellular networks, and broadband WLAN capabilities are starting to show up in mobiles as well. We refer to augmented reality implemented with mobile phones or PDAs as a Mobile Augmented Reality (MAR) system.

MAR promises to leverage the vast amount of information available over the Internet to augment the user's experience of reality. A paradigm we envision for MAR is for a camera-phone to visually sample the world from the user's perspective, extract pertinent information from the images, and display the images to the user with overlaid information. Systems that use a similar approach are [11][14][15].

Our method of extracting information from the images relies on computing unique, identifiable, size-invariant image-features (SURF, SIFT) [6][8]. These features are then matched against a database to discover the contents of the image. The information from the database is then localized in the image for display to the user. These operations should ideally be performed many times per second to give the user a smooth, real-time update of information.

Existing image matching methods deal with still images. In this paper we extend a still-image framework to video sequences. The goal is to provide a smooth user experience while adding minimal processing overhead.

It is inefficient to perform feature extraction and matching for each frame in the video sequence. Correlation in successive frames of the video should be exploited to reduce the frequency of feature extraction and matching. By tracking the content of the frames, we can determine when significant new content has appeared in the video. Feature extraction and database query need only be performed when there is new content.

Various feature tracking algorithms have been proposed, many of which are computationally complex. Since we are targeting the mobile phone platform we desire a fast, low complexity tracking algorithm.

Standardized video encoders, which are widely implemented in camera-phones, produce motion vectors in real-time using dedicated hardware. These motion vectors can be directly obtained from the encoder without any additional computation. We propose a novel feature tracking algorithm that uses these motion vectors and reduces the frequency of feature extraction and matching. Our algorithm has not yet been implemented on a mobile phone.

The structure of this paper is as follows. In Section 2, we present the points we wish to track. In Section 3, we discuss the details of the proposed algorithm. Finally, in Section 4, we present results to evaluate our tracking algorithm.

2 TRACKING

Not all features are good for tracking. For example, the aperture problem states that motion of edges can only be tracked in the same direction as their gradient. Therefore, only the horizontal component of motion can be determined for a vertical edge.

The problem of which features to track has been studied extensively in prior literature [4][7][10]. Traditionally, researchers have proposed tracking corners and textured regions. Our proposed scheme also tracks corners and textures using the algorithms described in the following sections. Texture and corner detection also play an important role in estimating global motion models.

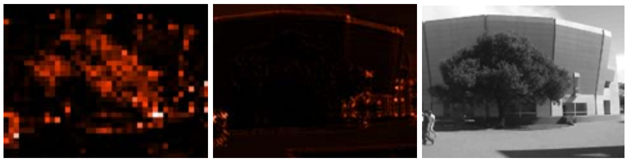
(a) (b) (c)

Figure 1: Output of the (a) texture detector and (b) the corner detector, and the original image (c).

2.1 Corner Detection

Corners are defined as image regions which contain sufficiently large changes in any two orthogonal directions. A common method, used by SURF, for computing corners is by calculating the determinant of the Hessian matrix. The SURF algorithm estimates derivatives with a simple box-filter to speed up detection. The SURF corner detector does respond mildly to edges, so choosing a relatively high threshold is important for

[1] Joint first authors

978-1-4244-1750-6/07/$25.00 ©2007 IEEE

excluding points that are difficult to track due to the aperture problem.

2.2 Texture Detection

Various methods exist to quantify the degree of texture in an image. These methods include using second-order image statistics, Markov Random Fields, and spatial filtering. We use a method based on region counts [12] that is straight-forward, computationally light, and corresponds well with the human visual system's perception of texture.

The algorithm counts the number of connected regions within each block. To count the number of regions, the algorithm quantizes the block into two representative levels. The number of spatially connected regions belonging to each level is computed using either an 8-connected or a 4-connected neighborhood. Counting the number of regions in a block works well for high contrast edges, as seen in Figure 2 (e) and (f).

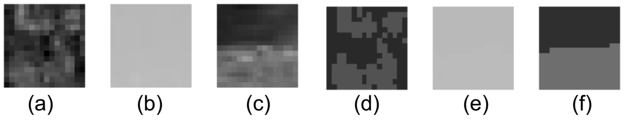

Figure 2: Examples of image blocks and their two-level quantization used for texture detection. (a) A textured block has many disconnected regions with high contrast in (d). (b) A flat block has low contrast regions (e). (c) An edge has two high contrast regions in (f).

However, the region count alone is insufficient for distinguishing flat from textured areas. For example, two-level quantized flat areas may have many disconnected regions due to noise. Considering the contrast difference between the two representative levels can be easily used to distinguish flat from textured areas, as seen in Figure 2 (c) and (d). We multiply the region count with the contrast for robustly detecting textured areas.

3 Algorithm

Our feature tracking algorithm is motivated by the fact that we should be able to track features as long as they are in the frame. Feature extraction and matching need only be done if there is significant new content in the frame. A simple metric for determining the amount of new content is how many features have left the frame.

3.1 Video Encoder

At present, the ITU-T recommendation H.264 [16] is the state-of-the-art video compression standard. Video coders in mobile phones currently use the H.263 [17] standard, even though H.264 can often be decoded. Due to significant compression gains, mobile devices will soon incorporate video encoders using the baseline profile of H.264. Video codecs, including H.264, use motion compensation to exploit the temporal redundancy within frames for the purposes of compression.

In many video encoders a small number of frames are grouped together to form a Group of Pictures (GOP). A GOP can be completely decoded without reference to frames outside the group. A video stream is composed of an ordered sequence of three types of frames: Intra-coded (I) frames, Predictively (P) coded frames, and Bi-directionally (B) coded frames. Intra-frames are coded as single frames without reference to any other frame. These frames are inserted at regular intervals to allow random-access and to mitigate error propagation. P-frames are coded using a motion-compensated prediction from only previous P or I frames, while B frames may use past and/or future frames. Both types of frames have motion vectors pointing to other reference frames. It is possible for P-frames to reference multiple previous frames. Figure 3 illustrates a GOP structure in temporal order, with arrows representing dependencies.

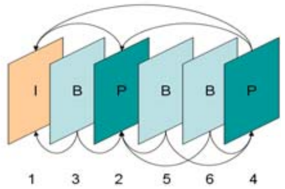

Figure 3: Three different types of frames, Intracoded (I), Predicted (P), and Bi-directionally predicted (B) constituting a Group of Pictures (GOP). Arrows represent dependencies. Numbers are the order in which processing must occur to resolve dependencies.

Each frame is divided into 16x16 macroblocks. Macroblocks in the current frame can be inter-coded (predicted) or intra-coded (not predicted). Inter-coded macroblocks point to macroblocks in previous frames. The residual between the current macroblock and the reference macroblock is quantized and entropy coded by the encoder.

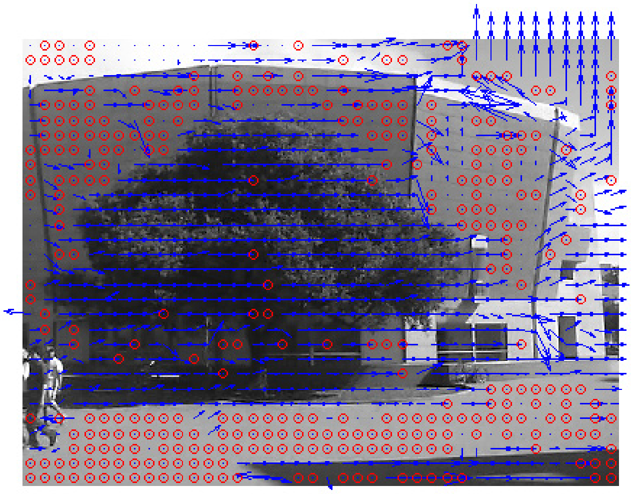

Figure 4: A frame from video sequence *Pan* overlaid with 16x16 macroblock motion vectors. Circles represent intra-coded blocks. This illustrates that motion vectors are not reliable estimates of true motion.

3.2 Challenges of Motion Vectors

There are several challenges in using motion vectors for tracking purposes. First, since motion vectors are used for prediction, they point backwards in time to blocks in previous frames. However, the points to be tracked are known in the previous frame and must be propagated forward in time. Second, motion vectors in video encoders are optimized for compression using a rate-distortion Lagrangian metric. Therefore, the motion vectors do not necessarily represent the true motion of the scene. As a result, many of the motion vectors are erratic, as seen in Figure 4. Third, not all pixels have motion vectors associated with them. For example, the macroblocks that are intra-coded do not reference previous frames and thus do not have motion vectors.

3.3 Model Based Scheme

The problem of Global Motion Estimation (GME) has been studied in computer vision and video processing [1][2][3][9]. GME is typically done in two parts; separating the foreground and the background in the video sequence, estimating the global motion model using the background motion-vectors. Our problem is simplified by the fact that feature matches belong to the background. However, background motion vectors are still not reliable for GME as they come from H.264, a highly rate-distortion optimized encoder, as is evident from Figure 4. An iterative RANSAC algorithm is used to robustly determine the global motion model and remove outliers.

To estimate the global motion, we considered two well-known models, the six-parameter affine model and the eight-parameter perspective model. At least four point-correspondences are required to determine the parameters of the perspective model, and at least three for the affine model.

After SURF feature extraction and matching, we assume that each feature has a texture level and a corner level associated with it. These texture and corner levels can then be used to our advantage when forming a global motion model. Motion vectors of blocks that are high in texture or have corners are more reliable than the motion vectors of blocks in flat or edge regions. Lee *et al*. [6] propose associating a confidence measure with each motion vector based on a normalized cornerness and distinctness metric. A similar approach has been used here.

First, points which do not have sufficient cornerness or texture are ignored. The remaining points are used in a RANSAC algorithm. To ensure the robustness of the resulting motion model we add additional checks to the RANSAC loop. We first ensure that points used to form a model are non-collinear, and well spaced. From these points an affine or perspective model is formed. The model is then checked for excessive shear or reflection, neither of which is possible between two successive frames of a video sequence.

3.4 Extension to Other GOP Structures

The model-based algorithm described in the previous sections is based on a GOP structure where each frame references only the previous frame and has no B frames. The algorithm can be easily extended to different GOP structures such as the one shown in Figure 3. We consider two additional GOP structures. The first is an IBPBP structure where a single B-frame is inserted between each I-frame and P-frame. The second structure allows P-frames to reference two past P-frames. In both cases, frames may reference multiple other frames. This leads to tracking dependencies and ambiguities which must be resolved either by considering the pixel values or by averaging locations.

4 Results

4.1 Test Sequences

We consider three test sequences: *Pan*, *Zoom* and *Occlusion*. All the three video sequences were collected with a Nokia N93 camera-phone outside the David Packard Electrical Engineering building at Stanford University. In the *Pan* sequence, the user stands in a fixed location and pans his camera across the surrounding buildings. In the *Zoom* sequence, the user zooms into a building. Finally, in the *Occlusion* sequence, the user is looking at a building when a person passes in front of the camera and partially occludes the building. We qualitatively and quantitatively evaluate the performance of the point-based and model-based schemes for each one of these sequences**.**

The primary objective of tracking features is to avoid feature extraction and matching on every frame of the video. Therefore, the ground-truth data for the location of the features is obtained by performing feature extraction on every frame. Feature matching is then carried out between the first frame and every subsequent frame using the SURF pair-wise matching algorithm [5][8]. The ground-truth data obtained from the feature-matching algorithm is used to evaluate the schemes.

4.2 Quantitative Results

After computing the ground-truth feature locations we compared our tracking schemes with two metrics – average and maximum error for each frame. Both metrics are measured with the Euclidean distance between the ground-truth locations and tracked locations. All experiments were performed in MATLAB.

In Figure 5 we first compare the tracking results for an affine and a perspective motion model using both metrics. The small number of tracked features and the extra degrees of freedom in the perspective model contribute to less accurate tracking results. As such, we use the affine model for all subsequent experiments.

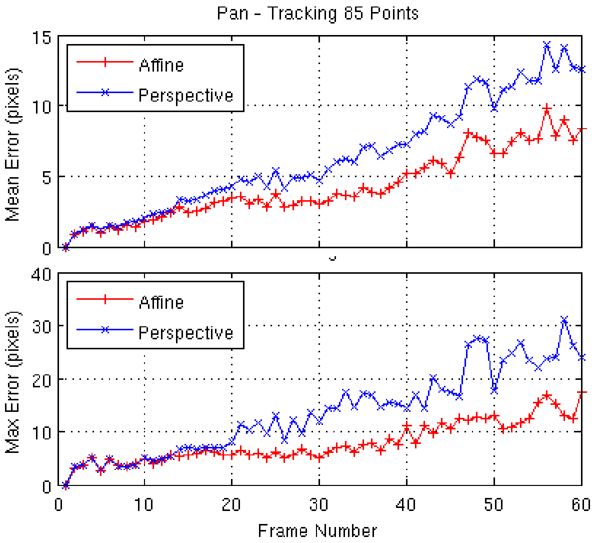

Figure 5: Comparison of tracking results for affine and perspective motion models with the model-based scheme.

Figures 6-8 show the average and the maximum error for the *Zoom*, *Occlusion*, and *Pan* sequences respectively. Each plot shows four different cases – a point-based scheme [13] for a single P-reference GOP, the model-based scheme for a single P-reference GOP, the model-based scheme with B-frames, and the model-based scheme with two P-reference frames.

We observe that the model-based scheme outperforms the point-based scheme in all three test sequences. The tracking for the model-based scheme is accurate to less than 10 pixels even after 100 frames. The point-based scheme, on the other hand, performs poorly as is evident from the large maximum error in each case.

For each of the sequences, the average error grows with time. The error in each of the affine models accumulates with time, and the points slowly drift from their ground-truth locations. However, the drift for the model-based scheme is small compared to the size of the frame (640x480 pixels).

The average error is lower for the B-frame sequences than the single and multiple P-reference sequences. For B-frame sequences tracking errors only accumulates between P-frames. Thus, less error is introduced because a lower number of tracking steps are used. The tracking accuracy is comparable for single and multiple P-reference frame sequences.

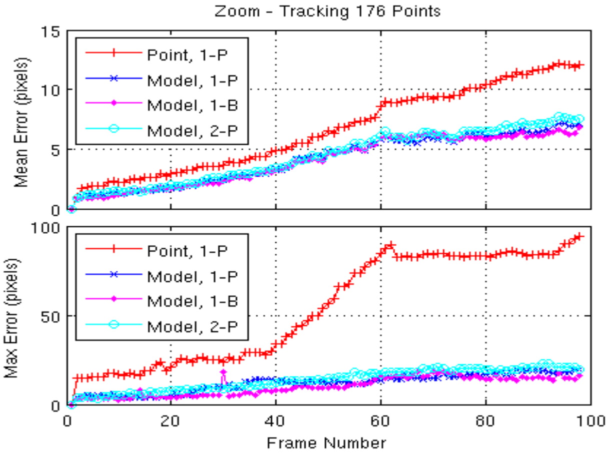

Figure 6: Tracking results for the *Zoom* sequence with point-based scheme, proposed scheme with single reference P frames, and proposed scheme with B frames. Top plot shows the mean error for all points. Bottom plot shows the maximum error for each scheme.

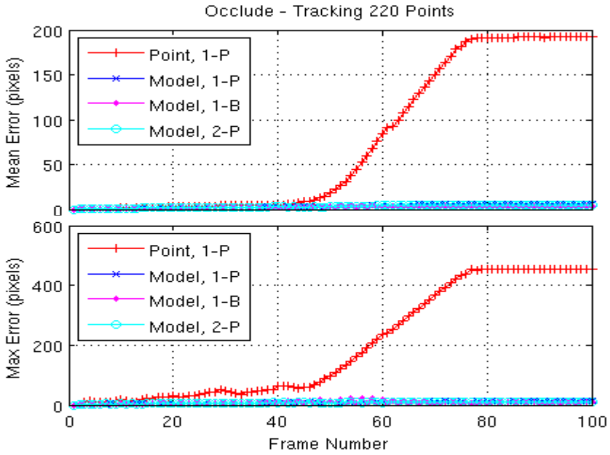

Figure 7: Tracking results for the O*cclusion* sequence with point-based scheme, proposed scheme with single reference P frames, and proposed scheme with B frames. Top plot shows the mean error for all points. Bottom plot shows the maximum error for each scheme.

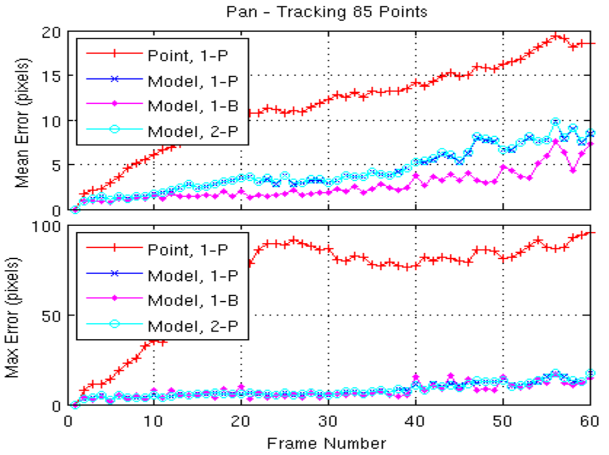

Figure 8: Tracking results for the *Pan* sequence with point-based scheme, proposed scheme with single reference P frames, and proposed scheme with B frames. Top plot shows the mean error for all points. Bottom plot shows the maximum error for each scheme.

5 CONCLUSION

We have developed a novel, low-complexity, tracking scheme that uses motion vectors directly from a video encoder. We have shown that we can track with a high level of accuracy, even though the motion vectors are highly rate-distortion optimized and do not represent true motion. We use a model-based approach that fits a global motion model to all the points we wish to track. The resulting tracking error is less than 10 pixels. We have extended our algorithm for tracking in video sequences with various GOP structures. Such a scheme would find applications in the context of MAR. Our algorithm can significantly reduce the required rate of feature extraction and matching.

REFERENCES

[1] Z. Zhang, F. Wang, G. Zhua, L. Xie, J. Gao, “A new global motion estimation algorithm”, *Proc. SPIE Multispectral Image Processing and Pattern Recognition (MIPPR)*, vol. 6044, pp. 599-606, 2005.

[2] Y.Huang, C.M.Kuo, C.L.Kuo, “Efficient global motion equation algorithm using recursive least squares”, *Proc. SPIE,* vol. 45(5), 2006.

[3] D. Xu, J. An, “Robust Global motion estimation method for aerial imagery”, *Proc. SPIE,* vol 44(9), 2005.

[4] C. Tomasi, T. Kanade, “Detection and tracking of point features”, *Technical Report CMU-CS-91-132,* April 1991.

[5] M.Brown, R. Szeliski, S. Winder, “Multi-image matching using multi-scale oriented patches”, *Proc. of the International Conference on Computer Vision and Pattern Recognition*, June 2005.

[6] D.Lowe, “Distinctive image features from scale-invariant keypoints”, *International Journal of Computer Vision,* vol 60(2), pp. 91-110, 2004.

[7] J. Shi, C. Tomasi, “Good features to track”, *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* June 1994.

[8] H. Bay, T. Tuytelaars, L. Van Gool “SURF: Speeded up Robust Features”, *Proc. of the ninth European Conference on Computer Vision*, May 2006.

[9] M. Pilu, “Using raw MPEG motion vectors to determine global camera motion”, *Proc. SPIE Conference on Visual Communications and Image Processing,* vol 3309, pp. 448-459, 1998.

[10] C. Kamath, A. Gezahegne, S. Newsam, G. Roberts, “Salient points for tracking moving objects in video”, *Proc. of Image and Video Communications and Processing Conference,* Jan 2005.

[11] G. Reitmayr, T. Drummond, “Going Out: Robust Model-based Tracking for Outdoor Augmented Reality”, *Proc. IEEE International Symposium on Mixed and Augmented Reality,* 2006.

[12] R. Bergman, H. Nachieli, G. Ruckenstein, “Detection of textured areas in images using a disorganization indicator based on component counts”, *HPL-2005-175(R.1)*, 2007.

[13] A. Mavlankar, D. Varodayan, B. Girod, “Region-of-Interest Prediction for Interactively Streaming Regions of High Resolution Video”, *Submitted to IEEE International Packet Video Conference Workshop*, 2007.

[14] R. Azuma, B. Hoff, H. N. Iii, and R. Sarfaty. “A Motion-Stabilized Outdoor Augmented Reality System.” In *VR ’99: Proceedings of the IEEE Virtual Reality,* page 252, Washington, DC, USA, 1999. IEEE Computer Society.

[15] T. Höllerer, S. Feiner, T. Terauchi, G. Rashid, and D. Hallaway. “Exploring MARS: Developing Indoor and Outdoor User Interfaces to a Mobile Augmented Reality System.” *Computers and Graphics,* 23(6):779-7785, 1999.

[16] “Advanced Video Coding for Generic Audiovisual Services”, ISO/IEC 14496-10, ITU-T Rec. H.264

[17] “Video Coding for Low Bit Rate Communication”, ISO/IEC 14496-2, ITU-T Rec. H.263