Real-time Motion Template Gradients using Intel CVLib

James Davis
MIT Media Lab
E15-390, 20 Ames St.
Cambridge, MA 02139 USA
jdavis@media.mit.edu

Gary Bradski
Intel Corporation
2200 Mission College Blvd.
Santa Clara, CA 95052 USA
gary.bradski@intel.com

Abstract

In this paper, we present an extension to the real-time motion template research for computer vision as previously developed in (Davis 1997). The underlying representation is a Motion History Image (MHI) that temporally layers consecutive image silhouettes (or motion properties) of a moving person into a single template form. Originally a global, label-based method was used for recognition. In this work, we construct a more localized motion characterization for the MHI that extracts motion orientations in real-time. Basically, directional motion information can be recovered directly from the intensity gradients within the MHI. In addition, we provide a few simple motion features using these orientations. The approach presented is implemented in real-time on a standard PC platform employing optimized routines, developed in part for this research, from the Intel® Computer Vision Library (CVLib). We conclude with an overview of this library and also a performance evaluation in terms of this research.

1 Introduction

An increasing interest in the recognition of human motion and action using computer vision has appeared, with much emphasis on real-time computability (for example Davis 1997; Bradski 1998; Bradski 1999; Freeman 1998; Ju 1996; Akita 1984; Darrell 1994; Cuttler 1998; Hogg 1983; Little 1995; Polana 1994; Rohr 1994; Haritaoglu 1998; Pinhanez 1999; Yamato 1992; Bregler 1997; Wren 1995; Cham 1998 and surveys in Cedres 1995; Shah 1997; Moeslund 1999a; Moeslund 1999b; Gavrila 1999). In particular, tracking/surveillance systems, human computer interfaces, and entertainment domains have a heightened interest in understanding and recognizing human movements. For example, monitoring applications may wish to signal only when a person is seen moving in a particular area (perhaps within a dangerous or secure area), interface systems may require the understanding of gesture as a means of input or control, and entertainment applications may want to analyze the actions of the person to better aid in the immersion or reactivity of the experience.

In earlier work (Davis 1997), a real-time computer vision representation of human movement known as a Motion History Image (MHI) was presented. The MHI is a compact template representation of movement originally based on the layering of successive image motions. The recognition method presented for these motion templates used a global moment feature vector constructed from image intensities, resulting in a token-based (label-based) matching scheme. Though this recognition method showed promising results using a large database of human movements, no method has yet been proposed to compute the raw motion information directly from the template without the necessity of *labeling* the entire motion pattern. Raw motion information may be favored for situations when a precisely labeled action is not possible or required. For example, a system may be designed to respond to leftward motion, but may not care if it was a person, hand, or car moving.

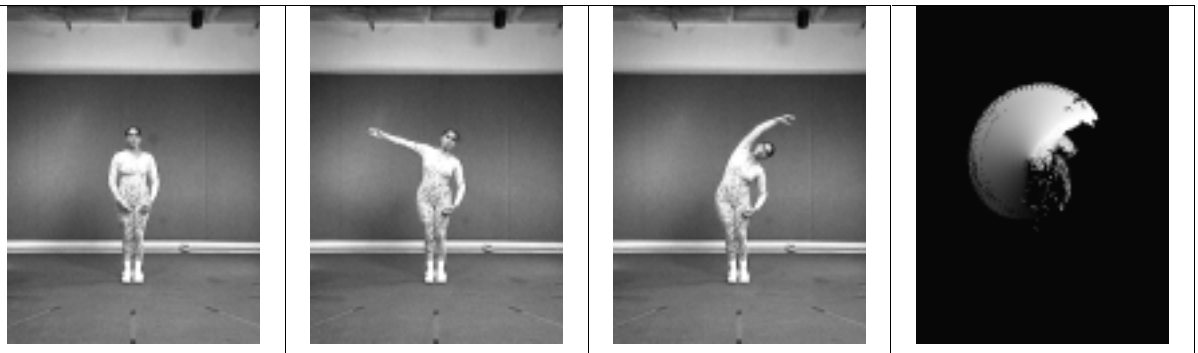
In this paper, we present an extension to the original motion template approach to now enable the computation of local motion information directly from the template. In this extension, silhouettes of the moving person are layered over time within the MHI motion template. The idea is that the motion template itself implicitly encodes directional motion information along the layered silhouette contours (similar to normal flow). Motion orientations can then be extracted by convolving image gradient masks throughout the MHI. With the resulting motion field, various forms of motion analysis and recognition are possible (e.g. matching histograms of orientations). To ensure fast and stable computation of this approach, we exploit the Intel *Computer Vision Library* (CVLib) procedures designed in part for this research. This software library has been optimized to take advantage of StrongArm and Pentium® MMX™ instructions. We offer the research presented in this paper both as a useful computer vision algorithm and as a demonstration of the Intel CVLib.

The remainder of this paper first examines the previous motion template work (Section 2). Next, we present the approach of computing motion orientations from a silhouette-based motion template representation (Section 3), along with describing potential motion features and generalities of the approach. We then present a brief overview of the Intel CVLib, the advantages resulting from the CVLib enhancement in terms of vision processing, and its comparative computational performance for this work (Section 4). We then summarize results and conclusions (Section 5). Appendix A details the calculation of moments and Hu moments used for the pose recognition approach for the templates (Section 3.1.1).

2 Previous Motion Template Research

In previous work, Davis and Bobick (Davis 1997) presented a real-time computer vision approach for representing and recognizing simple human movements. The motivation for the approach was based on how easily people can recognize common human movements (like sitting or push-ups) from low-resolution (blurred) imagery. Here there is an almost total lack of recognizable features, thus showing the role of motion for recognition. Accordingly, the method given relies on “patterns of motion” rather than on structural features as the representation for human motion. In that method, the space-time image volume containing the motion is collapsed into a single 2-D template while still perceptually capturing the *essence* of the movement and its

temporal structure. The template is generated by layering successive image differences of a moving person, and is referred to as a Motion History Image (MHI) (Similar to (Jain 1979)). An example is shown in Figure 1 for the movement of a person stretching her arm over her head.



Frame-0	Frame-35	Frame-70	MHI

Figure 1. Motion History Image for arm-stretching movement generated from image differences.

For recognition of the templates, seven higher-order moments (Hu 1962) are initially extracted from the MHI and also from a binarized version of the MHI. These fourteen moments are used as global shape descriptors and temporal recency localizers for the MHI. The moments are then statistically matched to stored examples of different movements. This method of recognition has shown promising results using a large database of movements.

Already, interactive systems have been successfully constructed using the underlying motion template technology as a primary sensing mechanism. One example includes a virtual aerobics trainer (Davis 1998) that watches and responds to the user as he/she performs the workout (See Figure 2(a)). In this system, motion templates are used to watch and recognize the various exercise movements of the person, which in turn affects the response of the virtual instructor. Another application using the motion template approach is The KidsRoom (Bobick 1997). At one point in this interactive, narrative play space for children, virtual monsters appear on large video projection screens and teach the children how to do a dance. The monsters then dance with the children, complementing the kids whenever they perform the dance movements (See Figure 2(b)). The dancing movements of the children are recognized using motion templates. Thirdly, a simple interactive art demonstration can be constructed from the motion templates. By mapping different colors to the various time-stamps (or graylevels) within the MHI and displaying the result on a large projection screen, a person can have fun "body-painting" over the screen (See Figure 2(c)), reminiscent of Krueger-style interactive installations (Krueger 1991). Other applications in general that must be "aware" of the movements of the person (or people) could also benefit from using the motion template approach. For example, gesture-based computer video games (Freeman 1998) or immersive experiences (e.g. Hawaii flight simulator, as shown in Figure 2(d)) designed to respond to user gesture control could use the previous template approach or employ our new method as an

additional sensing measurement or qualification of the person's movements. Even simpler gesture-based systems which rely on detecting characteristic motion (e.g. *hand swipes* to change a channel or a slide in a presentation) could profit from the fast and simple motion orientation features to be described.

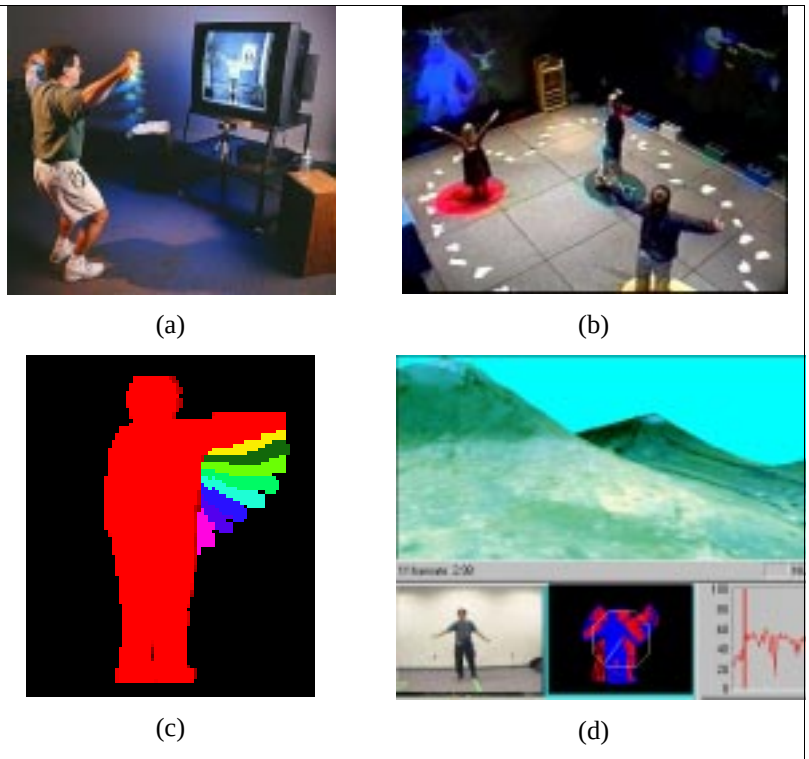


Figure 2. Interactive systems employing motion templates. (a) Virtual aerobics trainer (photo © Sam Ogden). (b) The KidsRoom. (c) Interactive body-painting. (d) Hawaii flight simulator.

The main limitation of single templates is that characterization of the template is token (label) based (e.g. "crouching" or "sitting"), where it cannot yield much information other than recognition matches (i.e. it cannot determine that, say, a lot of "up" motion is currently happening). We therefore wish to develop additional methods for analysis and recognition that are more descriptive of the local motion information. In the next section, we develop a method of analysis which extracts directional motion information from the MHI.

3 Motion Templates and Orientations

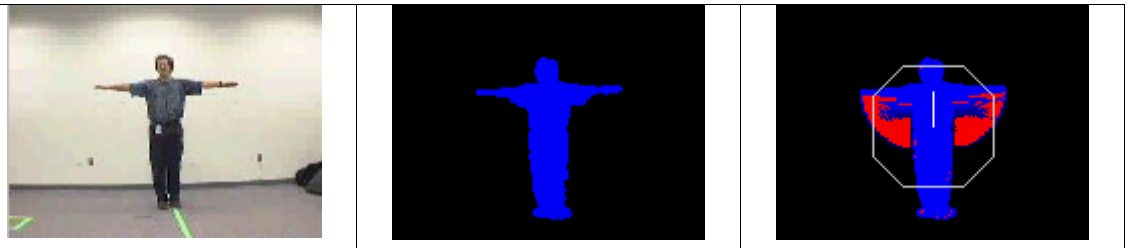
One could alternatively consider generating motion information by computing optical flow over the image or region of interest throughout the sequence, but this is computationally expensive and many times not robust. For example, hierarchical (Bergen 1992) and/or robust estimation (Black 1993) is often needed, and optical flow frequently signals unwanted motion in regions such as loose and textured clothing. Image differencing continues to be a fairly useful and cheap method for locating the *presence* of motion, but cannot directly extract the magnitude or direction of the motion. But as we will show, the *accumulation* of object silhouettes in the

motion template can yield useful motion information along the contours of the silhouette. We use the MHI representation described in the previous section, only now using silhouettes instead of image differences, as the basis for our approach to motion calculation.

3.1 Input

To acquire the full-body silhouettes of the person, we simply use a fast background subtraction routine that labels as foreground those pixels that are a set number of standard deviations from the mean RGB background. Then a pixel dilation and region growing method is applied to remove noise and extract the silhouette. A typical silhouette extracted from a uniform background using the method is shown in Figure 3(b). Other methods are obviously applicable and desirable for more complex backgrounds (Davis 1998; Wren 1995), but for simplicity and speed we opted for this approach. One immediate limitation of using silhouettes is that no motion inside the body region can be seen. For example, a silhouette generated from a camera facing a person would not show the hands moving *in front of* the body. One possibility to help overcome this problem is to simultaneously use multiple camera views. Another approach would be to separately segment flesh-colored regions and overlay them when they cross the foreground silhouette.

Given the silhouettes extracted for constructing the MHI, we can also use the pose of the silhouette to give additional information about the action of the person. For example, we could use the pose to recognize the body configuration, and then use the motion information to qualify how the person moved to that position.



(a) (b) (c)

Figure 3. Motion template generation. (a) Input video frame. (b) Silhouette in extracted from video. (c) MHI motion template (red shows where valid motion was detected, white line shows global direction of motion).

3.1.1 Pose Recognition from Silhouettes

We offer here a simple recognition method for silhouette poses based on the moment descriptors previously used for MHIs (Davis 1997). An alternative approach to pose recognition uses gradient histograms of and in the segmented silhouette region (Bradski, 1999). Seven higher-order moments (Hu 1962, Appendix A) provide excellent shape descriptors that are translation and scale invariant to the silhouettes with only minimal computation (requiring only a single pass of the image data). Since these moments are of different orders, we

cannot use a simple Euclidean metric for matching. Accordingly, we use the Mahalanobis distance metric (Therrien 1989) for matching based on a statistical measure of closeness to training examples:

$$mahal(x) \quad = \quad (x-m)^T K^{-1}(x-m) \tag{1.1}$$

where x is the moment feature vector, m is the mean of the training moment vectors, and K^{-1} is the inverse covariance matrix for the training vectors. This equation basically represents a hyper-ellipse with its center at m and principal axes aligned with the eigenvectors from the covariance matrix K. This distance gives a variance measure of the input to the training class. With enough training vectors, a reliable statistical threshold can be set to announce whether or not the input vector is statistically close to a particular training class.

However, to indicate the discriminatory power of these moment features for the silhouette poses, we need only a few examples of each pose (at least sufficient for matrix inversion) to relate the distances between the classes. For this example, the training set consisted of 5 repetitions of 3 gestural poses ("Y", "T", and "Left Arm") shown in Figure 4 done by each of five people. A sixth person who had not practiced the gestures was brought in to perform the gestures. Table 1 shows the Mahalanobis distances for these new test poses T_i matched against the stored training models P_i. The table correctly shows that the true matches for the test poses (along the diagonal) have distances considerably smaller than to the other model poses in each row even though the first two poses, "Y" and "T", are fairly close to one another. This is a typical result showing that thresholds tend to be easy to set with distances between even fairly similar gestures ("Y" and "T") still about an order of magnitude apart. We next describe the MHI motion template representation (generated from successive silhouettes) used to extract the directional motion information.

P_1

P_2

P_3

Figure 4. Three example poses used for Mahalanobis distance measurements in Table 1. P_1= "Y", P_2= "T", and P_3= "Left Arm".

	P_1	P_2	P_3
T_1	**14**	204	2167
T_2	411	**11**	11085
T_3	2807	257	**28**

Table 1. Discrimination results. Table entries report the Mahalanobis distance between the test poses (T_i) and model poses (P_i). P_1= "Y", P_2= "T", and P_3= "Left Arm". Diagonal entries are the true correspondences. Entries in bold show the minimum distance for each test input to the models.

3.2 MHI Representation

To generate the MHI for the movement sequence, we layer successive silhouette images (as done in (Davis 1998; Bobick 1997) rather than image differences (as in (Davis 1997) and Figure 1). In the MHI, each pixel value is defined as a function of the temporal history of position (or motion) at that point. We currently use a simple replacement and duration operator based on time-stamping (Davis 1999) (the previous method in (Davis 1997) was based on frames rather than time):

$$MHI_{\delta}(x,y) \quad = \quad \begin{cases} \tau & \text{if current silhouette at } (x,y) \\ 0 & \text{else if } MHI_{\delta}(x,y) < (\tau - \delta) \end{cases} \tag{1.2}$$

where τ is the current time-stamp, and δ is the maximum time duration constant (length of memory, typically a few seconds) associated with the template. The use of time-stamps allows for a more consistent port of the system between platforms where speeds may differ. System time is consistent during processing where frame rate is not. Thus time is *explicitly* encoded in the motion template. The above update function is called each time a new image is received and the corresponding silhouette image is formed. By linearly re-normalizing the MHI time-stamp values to graylevel values (0-255), we can generate a viewable image of the MHI. As shown in Figure 5(a), the more recent positions of the person are seen as brighter pixels than the older positions represented with progressively darker values. Almost immediately, the viewer of the motion template is able to discern the current pose of the person and the motion pattern used to reach that pose from some earlier position. This MHI is reminiscent of many famous stroboscopic image (Edgerton 1979; Braun 1992) and comic strip panels showing character motion, where movement is represented only in a single static frame. The premise here is that the motion template itself implicitly encodes motion information from the layered silhouettes.

3.3 Motion Gradient

The MHI layers the silhouette differences in such a way that motion from the silhouette contour can be perceived in the intensity gradient of the MHI pixel values (as originally proposed in (Davis 1999)). Notice in Figure 5(a), that as the arm is raised-up, the intensity gradient (from dark to light) gives the impression of motion in the direction of the upward arm movement. It can be said that the MHI "visually encodes" motion information from the silhouette contours. This boundary motion is very similar to the concept of normal flow along brightness contours (Horn 1986). We clearly see the *direction* of movement, but the magnitude is not as

accessible (unlike normal flow). Therefore we derive a simple method to extract only this directional component of motion.

As shown in Figure 5(a) and (b), the local intensity gradients of the MHI directly yield the orientation of the silhouette contour motion. Hence, we can convolve classic intensity gradient masks with the MHI to compute the motion orientations. For the convolution, we used Sobel gradient masks:

$$F_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \quad F_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}. \tag{1.3}$$

With the gradient images $F_x(x,y)$ and $F_y(x,y)$ calculated from the MHI, it is a simple matter to get the gradient (motion) orientation for a pixel by:

$$\phi(x,y) = \arctan\frac{F_y(x,y)}{F_x(x,y)}. \tag{1.4}$$

To compute the orientation we used a fast arctan approximation as described in (Capelli 1991). We must be careful, though, when calculating the gradient information because it is only valid at particular locations within the MHI. The surrounding boundary of the MHI layering should not be used because non-silhouette (zero valued) pixels would be included in the gradient calculation, thus corrupting the result. Only MHI *interior* silhouette pixels should be examined. Additionally, we must not use gradients of MHI pixels that have a contrast which is too low (inside a silhouette) or too high (large temporal disparity) in their local neighborhood. An example result of the motion gradient calculation is shown in Figure 5(b).

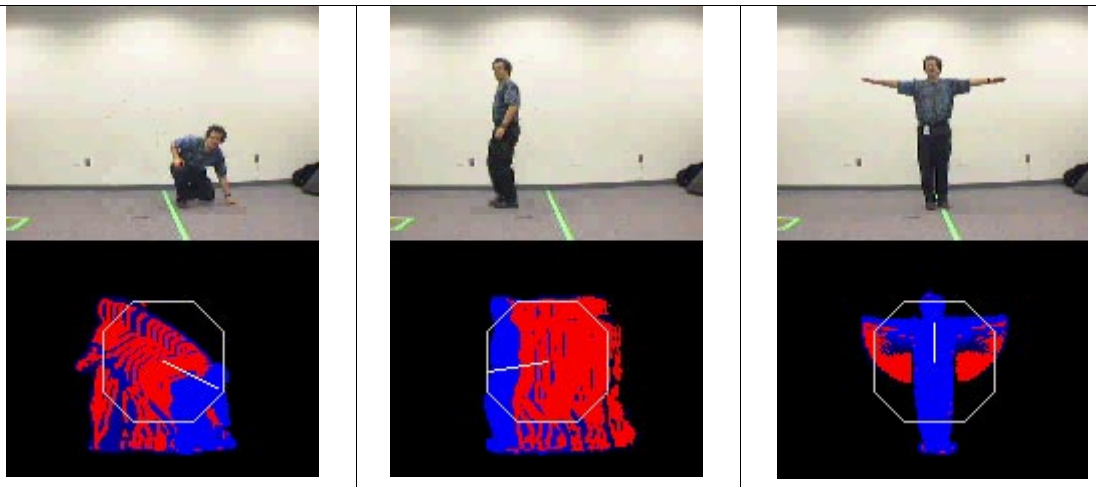
(a) (b) (c)

Figure 5. Motion gradients. (a) MHI for left arm movement up. (b) Sample motion orientations using Sobel convolution masks. (c) Radial histogram of the motion orientations with global direction identified.

With the calculation of this motion information, we can then extract motion features to varying scales. For instance, we can generate a radial histogram of the motion orientations (See Figure 5(c)), which then can be used directly for recognition (as in (Davis 1999) or as similarly done for spatial orientations in (Freeman 1995)). An even simpler measure is the single, *global* motion orientation for the movement. For example, "walking left" or "raising arms" with the camera facing the person would have global motion orientations of *leftward* (i.e. 180°) and *upward* (i.e. 90°) in the image coordinate frame, respectively. To be most useful, the calculation of the global orientation should be weighted by normalized MHI values to give more influence to the most current motion within the template. A simple calculation for the global weighted orientation is as follows:

$$\overline{\phi} = \phi_{ref} + \frac{\sum_{x,y} angDiff(\phi(x,y),\phi_{ref}) \times norm(\tau,\delta,MHI_{\delta}(x,y))}{\sum_{x,y} norm(\tau,\delta,MHI_{\delta}(x,y))} \quad (1.5)$$

where $\overline{\phi}$ is the global motion orientation, ϕ_{ref} is the base reference angle (peaked value in the histogram of orientations), $\phi(x,y)$ is the motion orientation map found from gradient convolutions, $norm(\tau,\delta,MHI_{\delta}(x,y))$ is a normalized MHI value (linearly normalizing the MHI from 0-1 using the current time-stamp τ and duration δ), and $angDiff(\phi(x,y),\phi_{ref})$ is the minimum, signed angular difference of an orientation from the reference angle. A histogram-based reference angle (ϕ_{ref}) is required due to problems associated with circular distance measurements. Figure 6 gives the global motion orientations for the movements of kneeling, walking, and lifting the arms. Complicated (or highly articulated) motions may be too intricate for this simple global measurement, but for some applications this singular direction may suffice. Since the MHI encodes in a single image the temporal nature for the motions over some time interval, we believe that motion segmentation (or clustering) should be easier here than in methods that attempt to segment and propagate (track) motion between frames (e.g. Cuttler 1998). Overall, once the motion orientations for the MHI have been calculated, many routines can be employed to use the raw information for analysis or recognition.



(a) (b) (c)

Figure 6. Global motion orientation measures. (a) Kneeling, viewed from the side: $\overline{\phi} = 335^{\circ}$. **(b) Walking left, viewed from the side:** $\overline{\phi} = 185^{\circ}$. **(c) Lifting arms, viewed from the front:** $\overline{\phi} = 90^{\circ}$.

3.4 Generalities of This Approach

One of the strongest advantage to this approach of motion template generation and analysis is that it is completely general as to the nature of the input data. For example, the input to the template generation could be almost any region-based calculation, such as silhouettes (as done here), binarized image differences or optical flow fields (as in (Davis 1997)), color-based (e.g. flesh-colored) regions, or thresholded stereo depth. Also, the object of interest need not be a person, but can correspond any object in motion, thus extending the method's applicability. Lastly, the use of the template-based motion orientations (whether it be with histograms, global motion, etc.) can be used in place of, or in conjunction with, other token-based (label-based) recognition methods (as shown with the previous pose identification or MHI recognition (Davis 1997)).

4 Intel CVLib and Performance Evaluation

We implemented the above work in CVLib and here we discuss the library itself and its advantages for this and other vision research. The main advantage of the CVLib in terms of hardware is that faster processing is now accessible on standard PC based platforms rather than specialty systems or costly workstations. As of this writing, Pentium II and Pentium III are supported, although StrongArm and Intel 64 bit systems will be supported in the future. The CVLib itself requires no special hardware subsystems or add-ons to perform its processor-optimized computations.

4.1 CVLib Description

The Intel computer vision library (CVLib) is intended to form a broad substrate of computer vision algorithms, highly optimized across Intel microprocessor products from StrongArm to Pentium, and is intended to run on

most popular operating systems: Windows, BeOS, and Linux. CVLib will join other existing, supported high performance libraries in Image Processing, JPEG, MPEG, Signal Processing, Math (matrix algebra), and Pattern Recognition freely available on the web at http://developer.intel.com/vtune/perflibst/. As of this writing, CVLib is in alpha testing and will be released on the web towards the last quarter of 1999.

It is intended that CVLib will eventually include functionality in the following areas:

- Image Statistical Measures: Mean, median, moments, Hu moments.
- Linear Algebra: SVD, eigenvalues , eigenvectors, cross-product, transpose product, matrix math.
- Optimization routines: Newton's, L-M.
- Image Pyramids: Gaussian and Laplacian.
- Clouds of points: Delaunay triangulations, voronoi diagrams, convex hulls.
- Drawing support: Lines, conic sections, flood fills.
- Segmentation: Skeletonization, distance transform, shape descriptors, morphological operators, background subtraction support, histogram backprojection, fast anisotropic diffusion.
- Feature finding: Canny edge detector, Hough transform, face features, corner, ridge, line detectors, general and separable convolution, fast fixed convolution kernels, image derivatives to third order.
- Pattern Recognition: Multi-dimensional histogram collection and comparison, MLP, HMM, DTW, Bayesian, Mahalanobis, K-NN, Eigen objects.
- Motion: Optical flow, normal optical flow, motion segmentation.
- Camera Calibration: Intrinsic, Extrinsic, radiometric, geometric, sub-pixel accuracy corner and line finders.
- 3D, Stereo: 8 point algorithm, matching on epipolar lines, dense stereo, single image model based pose in 6-DOF, finding the Homography matrix from vanishing points.
- Tracking: Color based, Snakes, Kalman filter, condensation filter, correlation.
- Transforms: FFT, DCT, wavelets.
- Utilities: Fast pixel access, cordic algorithm, random number generator, fast square root, inverse square root, image data shuffling.
- Works with all routines in Intel's Image Processing Library.

The low-level motion template routines are included in this library and we used the optimized C code versions of these routines to compare performance against the optimized assembly MMX™ instruction versions in the section below.

4.2 CVLib Performance Evaluation

The motion gradient algorithm described in this paper is a very efficient way to get approximate normal optical flow directions. To get a rough idea what further improvement could be derived from different levels of optimization, the motion code was run on a 400MHz Pentium III machine with quarter resolution frames in 3 different scenarios: a straight forward, pre-CVLib C implementation, an optimized C implementation, and an optimized MMX assembly implementation. Table 2 records the results.

Un-optimized C	Optimized C	Optimized MMX Assembly
22 frames/second	27 frames/second	30+ frames/second

Table 2. Frame rates for different implementations.

To further study the benefits of assembly MMX optimization we took advantage of the following fact. CVLib relies on dynamically linked libraries. At run time, the code checks the CPU type and automatically loads the corresponding optimized dynamic library. If such a library cannot be found, it defaults to using an optimized C library version. Thus when the Pentium III MMX assembly optimized dynamic linked library version (**ippicvlA6.dll**) is present, it will be run. When it is not present, the optimized C version (**ippicvlPX.dll**) will be run. Thus the same application may be run in C optimized or Assembly MMX optimized versions.

A video was recorded of a hand waving back and forth across the scene with a frequency of about one hertz. During the wave, the hand size fluctuated between taking up one to three quarters of the image. The video was two and a half minutes in duration. Intel's VTUNE™ Performance Analyzer code was used to record the performance of the motion template gradient algorithm on this video sequence using both the optimized C (**PX**) and assembly (**A6**) versions of CVLib. VTUNE™ rapidly samples the application and produces a report as to the percent of time spent in each module. Since optimized code runs faster, it might get called more often, so we can't use total percent of time spent in PX vs. A6 as a comparison. Instead we have the situation as shown in Table 3.

	PX	A6
Motion Template Time	A	A/K
Time in rest of code	1-A	1-A

Table 3. "A" is the percentage of time spent in the optimized C code PX. The MMX assembly optimized code A6 has a speed-up factor "K" as compared to PX. The total amount of time spent in the outside code, 1-A is the same for both PX and A6.

We can see from Table 3 that if A is the percent of time spent in the PX motion template gradient code, then if the percent of time spent in the A6 code is B, we have:

$$\frac{\frac{A}{K}}{\frac{A}{K}+1-A} = B. \tag{1.6}$$

The results of VTUNE analysis for motion template gradient code A6 and PX are shown in Figure 7 and Figure 8. We find that A=0.51 and B=0.3589 in equation (1.6). Solving for the speedup factor K yields 1.86 or a 186% speed up by using the A6 version of CVLib, bringing the frame rate to real time (30Hz) or faster. Note that CVLib is still just in alpha release so that these results should improve. The outside application code (MOTIONGRAD.EXE) went from being able to use 25.6% of the available time in PX to 33.75% of the time in A6. This is a 32% improvement in the work the computer can do other than running computer vision despite processing 11% more frames per second.

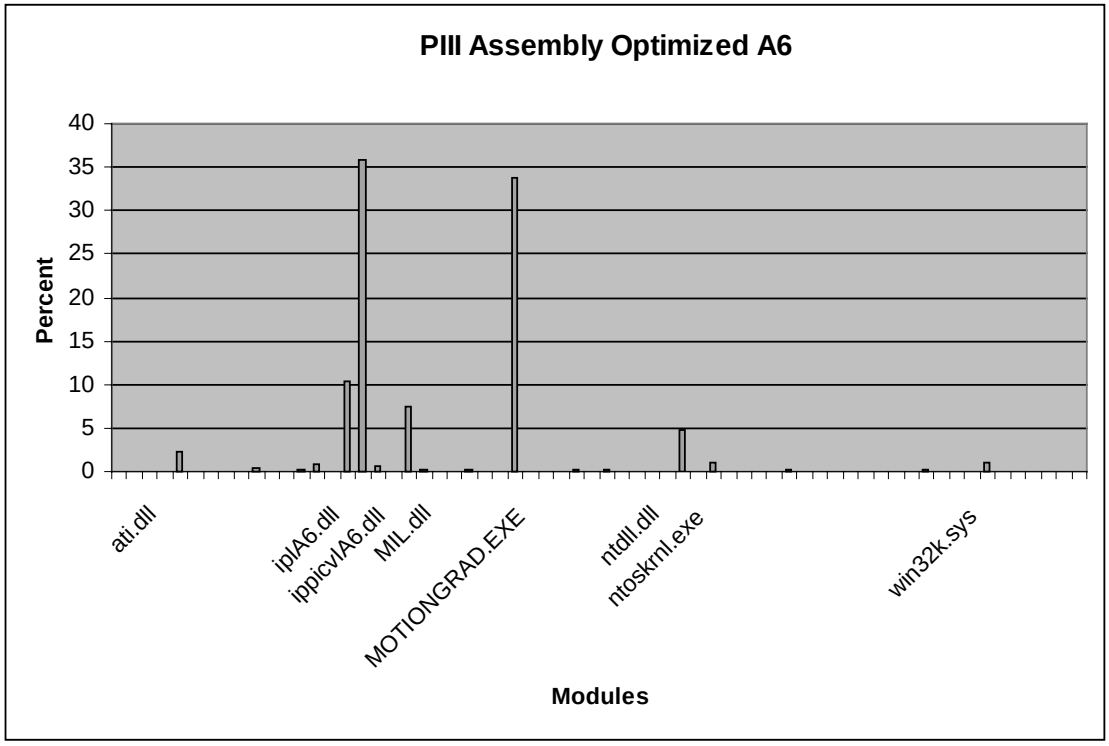


Figure 7. Motion Template Gradient code A6 was active B = 35.89% of the time.

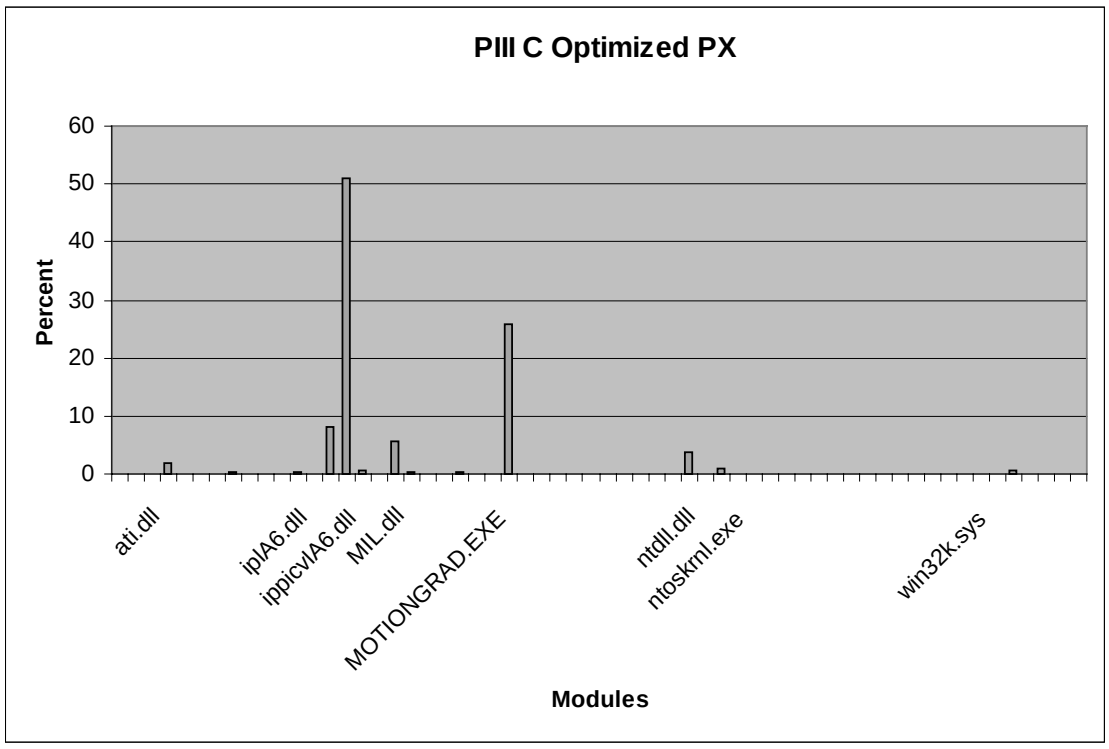


Figure 8. Motion Template Gradient Code PX was active A = 51% of the time.

4.3 CVLib Performance Discussion

In terms of the motion template research described in this paper, there are two main benefits of using the CVLib. First, faster processing rates enabled by using CVLib offer a more stable and consistent MHI representation with silhouettes. Unlike standard optical flow routines which calculate motion usually between successive frames (motion over milliseconds), our silhouette layering approach is based on a larger time scale (motion over seconds). Since the position of the person (silhouette) over time is layered into the template, faster frame rates only reduce the contour disparities, corresponding to a finer and more dense motion resolution. Thus faster frame rates increase the granularity of the motion template. The second advantage of using the CVLib is that it may allow more views to be processed on a single machine. Currently, only one view of the person is processed. As previously stated in Sect. 3.1, silhouettes have the problem of masking "internal" motion within the silhouette region, but using multiple views of the person is one suggestion to overcome this hindrance. Therefore, if the motion template method is fast enough while not occupying most of the CPU, we could simultaneously process multiple views of the moving person on a single machine to account for the possible silhouette occlusions (as similarly performed in (Davis 1997) for robust matching). Having a single machine processing these multiple views of the person (via multi-video inputs or video multiplexer), rather than multiple machines each processing a single view, is obviously advantageous in terms of cost and synchronization.

Given the methods described in this work and the overall functionality of the CVLib, many applications could benefit from the proposed methodology and implementation. Systems incorporating human movements for input must recognize and respond quickly to the user without noticeable lag to give a sense of immersion and control. The quickness of the characterization and its response is paramount. We believe that the motion template research, as optimized in CVLib, offers such a system.

5 Summary

In this paper, we presented an extension to previous motion template research (Davis 1997) by offering a method of calculating local motion orientations directly from the template. The approach is based on a Motion History Image (MHI) that combines successive silhouettes of a person. The main premise of the method is that these layered silhouettes implicitly encode motion information along the silhouette contours of the moving person. The motion appears in the form of directional measurements from the intensity gradients in the MHI. Thus, we provided a method of extracting local motion orientations from the MHI by convolving intensity gradient masks with the MHI. To characterize the motion orientations, we provided a simple method for calculating the *global* motion orientation of the person. The approach presented was implemented on a standard PC platform employing the Intel Computer Vision Library (CVLib). We lastly presented a brief overview of this library and its performance measures for this research.

6 Appendix A: Moments and Hu Moments

This appendix is included to make this paper self-contained for implementation purposes, and may be omitted otherwise. The appendix describes central moments to the 3rd order, and then their combination into rotationally invariant Hu moments (Hu 1962).

6.1 Moments, Their Use up to The 3rd Order

Since we're dealing with images, we need to work with two dimensional moments. Moments give us an indication of the center, spread, skewness etc. of what we are measuring. In our case, our measures are pixel values and locations. In this case, moments can give us a highly compressed indication of the shape of the object being measured in our image. The two-dimensional moment, m_{pq}, of order $p+q$, of a density distribution function, $f(x,y)$, is defined as

$$m_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q f(x,y)dxdy \tag{A1}$$

The two-dimensional moment for a discretized image, $f(x,y)$, is

$$m_{pq} = \sum_{-\infty}^{\infty}\sum_{-\infty}^{\infty} x^p y^q f(x,y) \tag{A2}$$

A complete moment set of order n consists of all moments, m_{pq}, such that $p+q \le n$. The use of moments for image analysis and object representation was introduced in 1962 by Hu in his seminal paper (Hu 1962). Hu's *uniqueness theorem* states that if $f(x,y)$ is piecewise continuous and has nonzero values only in a finite region, then the moments of all orders exist. Moreover, the theorem proves that the moment set $\{m_{pq}\}$ is uniquely determined by $f(x,y)$ and conversely, $f(x,y)$ is uniquely determined by $\{m_{pq}\}$. Since an image segment has a finite area and, in the worst case, is piece-wise continuous, moments of all orders exist, and a moment set can be computed that will uniquely describe the information contained in the image segment. To characterize all of the information contained in an image segment requires a potentially infinite number of moments. The challenge is to select a meaningful subset of moments that contain sufficient information to accurately characterize the image. Since noise effects are large in video images and we are looking for a highly compressed indication of shape, we only consider moments up to the third order.

To gain a better intuition of how to reason about moments and get an insight into how to derive invariant features based on moments, we next consider several low-order moments and describe their physical meaning. The definition of the zero-th order moment, m_{00}, of the image $f(x,y)$ is

$$m_{00} = \sum_{-\infty}^{\infty}\sum_{-\infty}^{\infty} f(x,y) \tag{A3}$$

This moment represents the total mass of the given image. When computed for a silhouette image, the zero-th moment represents the total object area.

The two first order moments, $\{m_{10}, m_{01}\}$, are used to locate the *center of mass* of the object. The center of mass defines a unique location with respect to the object that may be used as a reference point to describe the position of the object within the field of view. The coordinates of the center of mass can be defined through moments as shown below

$$\hat{x} = \frac{m_{10}}{m_{00}} \quad \hat{y} = \frac{m_{01}}{m_{00}} \tag{A4}$$

6.2 Hu Moments

To center a visual object such that its center of mass is at the origin, $\hat{x} = 0$ and $\hat{y} = 0$, of the moments calculation, we define central moments as follows:

$$\mu_{pq} = \sum_{-\infty}^{\infty} \sum_{-\infty}^{\infty} (x - \hat{x})^p (y - \hat{y})^q f(x, y) \tag{A5}$$

We note that central moments are translation invariant. To achieve scale invariance we normalize central moments as follows

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{[(p+q)/2]+1}} \tag{A6}$$

For arbitrary shapes, a potentially infinite number of moments $\{\eta_{pq}\}$ can uniquely describe that shape. These image moments are invariant with respect to translation and scale operations, but an infinite number of moments are obviously impractical, and in any case, we want a compressed representation of shape. Since objects in video images contain a large amount of noise, only moments up to the 3rd order are generally practicable.

There are ten moments up to the 3rd order, but scale normalization and translation invariance fix three of these moments at constant values. Rotation invariance takes away one more degree of freedom leaving us with six independent dimensions. Hu uses seven variables however: six to span the six degrees of freedom, and a final seventh variable who's sign removes reflection invariance. Only the first six of the Hu variables below give reflection invariance.

The seven moment-based features proposed by Hu that are functions of *normalized* moments up to the third order are:

$$\begin{aligned}
\phi_1 &= \eta_{20} + \eta_{02} \\
\phi_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
\phi_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
\phi_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2
\end{aligned}$$

$$\phi_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})^2[(\eta_{03} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] +$$
$$(3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$
$$\phi_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$$
$$\phi_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] -$$
$$(\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \quad \text{(A7)}$$

In the first two equations, ϕ_1 and ϕ_2, provide scale and translation independence, equations ϕ_3-ϕ_6 ensure rotation with reflection invariance, and ϕ_7 provides reflection discrimination in it's sign.

7 Bibliography

Aggarwal, J. and Q. Cai. Human motion analysis: a review. IEEE Wkshp. Nonrigid and Articulated Motion Workshop, Pages 90-102, 1997.

Akita, K. Image sequence analysis of real world human motion. Pattern Recognition, 17, 1984.

Bergen, J., Anandan, P., Hanna, K., and R. Hingorani. Hierarchical model-based motion estimation. In Proc. European Conf. on Comp. Vis., pages 237-252, 1992.

Black, M. and P. Anandan. A frame-work for robust estimation of optical flow. In Proc. Int. Conf. Comp. Vis., pages 231-236, 1993.

Bobick, A., Davis, J., and S. Intille. The KidsRoom: an example application using a deep perceptual interface. In Proc. Perceptual user Interfaces, pages 1-4, October 1997.

Bradski, G., Yeo, B-L. and M. Yeung. Gesture for video content navigation. In SPIE'99, 3656-24 S6, 1999.

Bradski, G. Computer Vision Face Tracking For Use in a Perceptual User Interface. In Intel Technology Journal, http://developer.intel.com/technology/itj/q21998/articles/art_2.htm, Q2 1998.

Braun, M. Picturing time: The work of Etienne-Jules Marey (1830-1904). University of Chicago Press, 1992.

Bregler, C. Learning and recognizing human dynamics in video sequences. In Proc. Comp. Vis. And Pattern Rec., pages 568-574, June 1997.

Capelli, R. Fast approximation to the arctangent. Graphics Gems II, Academic Press, James Arvo Editor, pages 389-391, 1991.

Cedras, C. and M. Shah. Motion-based recognition: a survey. Image and Vision Computing, Vol. 13, Num 2, pages 129-155, March 1995.

Cham, T. and J. Rehg. A multiple hypothesis approach to figure tracking. In Proc. Perceptual User Interfaces, pages 19-24, November 1998.

Cuttler, R. and M. Turk. View-based interpretation of real-time optical flow for gesture recognition. Int. Conf. On Automatic Face and Gesture Recognition, page 416-421, 1998.

Darrell, T., Maes, P., Blumberg, B., and A. Pentland. A novel environment for situated vision and behavior. In IEEE Wkshp. For Visual Behaviors (CVPR), 1994.

Davis, J. Recognizing movement using motion histograms. MIT Media lab Technical Report #487, March 1999.

Davis, J. and A. Bobick. Virtual PAT: a virtual personal aerobics trainer. In Proc. Perceptual User Interfaces, pages 13-18, November 1998.

Davis, J. and A. Bobick. A robust human-silhouette extraction technique for interactive virtual environments. In Proc. Modelling and Motion capture Techniques for Virtual Environments, pages 12-25, 1998.

Davis, J. and A. Bobick. The representation and recognition of human movement using temporal templates. In Proc. Comp. Vis. and Pattern Rec., pages 928-934, June 1997.

Edgerton, H. and J. Killian. Moments of vision: the stroboscopic revolution in photography. MIT Press, 1979.

Freeman, W., Anderson, D., Beardsley, P., et al. Computer vision for interactive computer graphics. IEEE Computer Graphics and Applications, Vol. 18, Num 3, pages 42-53, May-June 1998.

Freeman, W., and M. Roth. Orientation histograms for hand gesture recognition. In Int'l Wkshp. On Automatic Face- and Gesture-Recognition, 1995.

Gavrila, D. The visual analysis of human movement: a survey. Computer Vision and Image Understanding, Vol. 73, Num 1, pages 82-98, January, 1999.

Haritaoglu, I. Harwood, D., and L. Davis. W4S: A real-time system for detecting and tracking people in 2 ½ D. European. Conf. On Comp. Vis., pages 877-892, 1998.

Hogg, D. Model-based vision: a paradigm to see a walking person. Image and Vision Computing, Vol. 1, Num. 1, 1983.

Horn, B. Robot Vision. MIT Press, 1986.

Horprasert, T. Haritaoglu, I., Harwood, D., et al. Real-time 3D motion capture. In Proc. Perceptual User Interfaces, pages 87-90, November 1998.

Hu, M. Visual pattern recognition by moment invariants. IRE Trans. Information Theory, Vol. IT-8, Num. 2, 1962.

Jain, R. and H. Nagel. On the analysis of accumulative difference pictures from image sequences of real world scenes. IEEE Trans. On Pattern Analysis and Machine Intelligence, Vol. 1, Num. 2, April 1979.

Ju, S., Black, M., and Y. Yacoob. Cardboard people: a parameterized model of articulated image motion. Int. Conference on Automatic Face and Gesture Recognition, pages 38-44, 1996.

Krueger, M. Artificial reality II, Addison-Wesley, 1991.

Little, J., and J. Boyd. Describing motion for recognition. In International Symposium on Computer Vision, pages 235-240, November 1995.

Moeslund, T. Summaries of 107 computer vision-based human motion capture papers. University of Aalborg Technical Report LIA 99-01, March 1999.

Moeslund, T. Computer vision-based human motion capture – a survey. University of Aalborg Technical Report LIA 99-02, March 1999.

Pinhanez, C. Representation and recognition of action in interactive spaces. MIT Media Lab Ph.D. Thesis, June 1999.

Pinhanez, C., Mase, K., and A. Bobick. Interval scripts: a design paradigm for story-based interactive systems. Conference on Human Factors in Computing Systems, pages 287-294, 1997.

Polana, R. and R. Nelson. Low level recognition of human motion. In IEEE Wkshp. On Nonrigid and Articulated Motion, 1994.

Rohr, K. Towards model-based recognition of human movements in image sequences. CVGIP, Image Understanding, Vol. 59, Num 1, 1994.

Shah, M. and R. Jain. Motion-Based Recognition. Kluwer Academic , 1997.

Therrien, C. Decision Estimation and Classification. John Wiley and Sons, Inc., 1989.

Wren, C., Azarbayejani, A., Darrell, T., and A. Pentland. Pfinder: real-time tracking of the human body. SPIE Conference on Integration Issues in Large Commercial Media Delivery Systems, 1995.

Yamato, J., J. Ohya, and K. Ishii. Recognizing human action in time sequential images using hidden markov models. In Proc. Comp. Vis. and Pattern Rec., 1992.

Assembly optimized performance libraries in Image Processing, Signal Processing, JPEG, Pattern Recognition and Matrix math are at http://developer.intel.com/vtune/perflibst/.