from Proceedings of Spring Conference on Computer Graphics 2006
Bratislava: Comenius University, 2006, p. 177-183. ISBN 80-223-2175-3

Coding of vectorized cartoon video data

Martin Koloros*
Czech Technical University

Jiří Žára†
Czech Technical University

Abstract

We present an efficient video compression technique suitable for outline based cartoon animations. In contrast to commonly used video compression approaches we adopt a new technique where the dynamic homogenous foreground layer is separated from the static textural background and each is coded using appropriate compression technique. The aim of this paper is to reduce temporal and spatial redundancy caused by similar shape patterns in the foreground layer. Our shape matching methods are designed to be able to find identical and similar shapes even if they are transformed towards the original by translation, rotation or scaling. These methods will replace identical polygons by reference to the parent polygon and therefore save vertices. We have three steps applied on vectorized layer of video data. The first stage consists of search for unchanged polygons within set of frames is being performed. At second step the similar polygons within one frame are being replaced by reference and finally similar polygons within certain frame range are replaced by reference. Experiments proved that proposed methods are efficient, useful and they contribute to ability of playing flawless quality vectorized video data at minimum bitrate.

CR Categories: E.4 [DATA]: CODING AND INFORMATION THEORY—;

Keywords: video, compression, shape match, cartoon

1 Introduction

Classical storage techniques like celluloid films or analogue video tapes undergo various mechanical and physical degradations that significantly reduce the visual quality in time. Sampling of analog signal and expressing the information in digital form is used today to guarantee the information quality and make it media independent. Sampling frequency is one of factors determining target quality and storage space requirements. To achieve good fidelity it is often necessary to produce large amount of data. Several video compression methods were invented to be able to effectively store this information on common digital media.

Today popular video compression techniques based on MPEG-2 standard [Group 1995] use discrete cosine transform (DCT) where separated luminance and chrominance frame layers are divided into small blocks (typically 8x8 pixels) and converted into frequency domain. Lower encoding bit-rate is achieved by cutting out less important frequencies. In this process quantization matrix is used to trade off final visual quality and encoding bit-rate. Resulting stream is compressed using Huffman algorithm [Salomon 1998].

The advantage of DCT-based methods lies in general purpose applicability and quality control. By employing more advanced algorithms for motion compensation and coding only inter-frame changes, it is possible to lower bit-rate to very useable level. However, the problem is rather complex as video streams need high bitrates to preserve compelling visual quality. When not enough high frequencies are included then the artifacts will arise. These artifacts are observable especially at high contrast areas. To address this issue several methods have been developed to preserve several important DCT coefficients [Horng and Ahumada 1995], to post-process the stream for playback [Yen and Tai 2005] or to change macro-block size [Ebrahimi and Horne 2000]. However these modifications still do not allow to take full advantage of specific patterns occurring at classic outline based cartoon animations.

In this paper we adopt the concept from MPEG-4 proposal [Group 1998], where scene is divided into a set of separate layers depending of its content and expectations. Each layer is then encoded into separate stream using appropriate coding technique. Final streams are multiplexed and adjusted for network transmission or storage. In the case of classical cartoon animation it is possible to create at two separate layers – one for static background and one for outlined animated parts in the foreground layer.

Contents in the foreground layer are in certain rate similar between frames as in most cases only part of the frame is being engaged into animation. Removing identical entities and replacing them with reference to their parent objects is core of this work. Several techniques from fast and simple to more advanced but time demanding are presented. The paper is organised like follows. Section (2) informs about previous research achievements in this area. In section (3) the data source is described and removing redundant frames follows in section (4). Part (5) is core of this paper and describes techniques used in inter-frame shape matching.

2 Previous work

Even that compression and coding of video stream is long term intensively studied area, not many methods to code as specific data as outline based cartoon were presented.

[Kwatra and Rossignac 2002] exploit visual aspects of cartoon animations are considered. In this approach each region is first represented as a 3D volume by sweeping its 2D shape through the time. Then edge-breaker compression scheme is used to encode volume geometry. Decoding is performed by intersecting compressed volume with image plane using GPU-based solid clipping technique. Unfortunately this compression is suitable only for regions that change their shape and/or position slightly. Authors also did not address the problem of region shape extraction for more complicated color and gray-scale image sequences.

Our aim is to extend approach presented in [Sýkora et al. 2005b]. In this method the original animation frame is first separated into a set of regions using unsupervised image segmentation technique designed for classical cartoons [Sýkora et al. 2005a]. Motion estimation is then used to register parts of the background to stitch and

*e-mail: kolorom@fel.cvut.cz
†e-mail: zara@fel.cvut.cz

from Proceedings of Spring Conference on Computer Graphics 2006
Bratislava: Comenius University, 2006, p. 177-183. ISBN 80-223-2175-3

store background layer as one big image. Shapes of homogenous color regions in the foreground layer are converted from raster to vector representation and encoded separately using 1D DCT-based technique. To search for frame duplicities and to store new frames only a pool of already stored frames is maintained. During the playback standard graphics hardware is used to render the background layer as a textured rectangle and in front of it foreground layer as a set of flat colored polygons.

In this paper we are dealing with polygon similarity. The essential step is to find proper polygon description prior to any polygon matching. There are already many techniques for polygon description. Nice survey from [Loncaric 1998] gives a brief information about this theme.

Shape matching itself is rather complex issue where used algorithms are often adapted to measured content. Taxonomy of shape matching is described by [Tangelder and Veltkamp 2004].

3 Acquiring vector based data

3.1 Data source

In our work we focus on classic handmade content where primary storage medium is celluloid slice. Filmed sequence of these slices is source material for our work. To be used in digital form several steps must be taken.

The Telecine process is used to transform information from film into digital video format. For production PAL or NTSC standard an appropriate resolution of 720x576 or 720x480 pixels is used and final data are commonly stored at Betacam tape. One tape can keep up to 124 minutes of 2:1 lossless compressed video data. Use of MPEG-2 stream will produce about 3.5 Mbps bitrate video steam and will result in lower quality. According to upcoming use of MPEG-4 codec in digital video broadcasting (DVB) or high definition (HDTV) stream we can expect better quality at reasonable data-rate. For our intention to vectorize video data we need them to be the best possible quality. Any previous DCT coding with artifacts will make the vectorization difficult or even impossible.

4 Processing of vectorized data

4.1 Omitting redundant frames

It cartoon video it is common technique to repeat frames. Either to present still scene and entertain the observer by sound channel only or to decrease the frame rate. Just by duplicating each frame the framerate drops by half. Third case with repeating frames is based on repeating whole set of consecutive frames. It occurs in animation like bicycle riding and pushing the pedals, etc. Here frame redundancy is not just between consecutive frames but between frames placed into stream with given period. We can expect this and have it on mind when coding vectorized video sequence. Detecting frame change is possible in two ways. Either by comparing frames in raster form in original sequence by measuring PSNR[1] of their difference or by comparing vectorized layers only. First method is sensitive to any changes between frames therefore it is essential to use almost noise free source data. The second method uses vector layer only where it is resolution independent and not very sensitive to noise. To speed up the search an hierarchical approach is used. First thumbnails of raster form consecutive frames are produced and aligned using phase correlation. Bitwise XOR and distance transform [Borgefors 1984] then produces weighted difference map which emphasizes large changes against small global shifts. Weights from this map are summed up and compared to certain threshold. If threshold is not exceeded the compared frames are considered as identical and only one frame is being coded as parent where the second one is stored as its reference. By adjusting the threshold it is possible to omit insignificant inter-frame changes and therefore lowering bitrate or tuning fine quality details. In our experiments this method has been used not only between two consecutive frames but between current frame and all previous frames therefore the best possible match could be found.

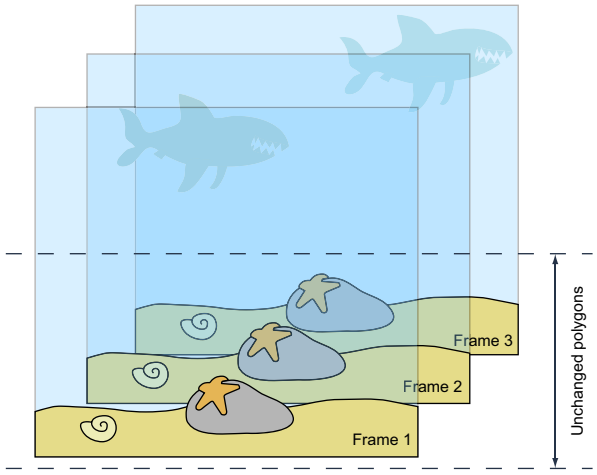

Figure 1: Referencing shapes unchanged on several frames

5 Frame correlation

5.1 Basic identical polygon matching

When an animator creates a cartoon clip he usually keeps focusing just on parts which are changing between frames. The rest remains unchanged in a way to prevent inter-frame shape mismatch or just to ease the work. In two or more consecutive frames certain portions are being used in unchanged form several times. Detecting these identical polygons will allow to code them as parent polygon and its instances. Naive approach comes from native polygon representation (set of connected 2D vertices creating polygon boundary) and seeks for unchanged polygons sharing the same position within set of neighbouring frames (see fig. 1). First the boundary rectangles are created over all polygons and for selected polygon in current frame we try to find its counterpart in neighbouring frame. When boundary rectangle center and dimensions match with another one within acceptable range then candidates polygons are compared by vertices. For each vertex $A(i), i \in \langle 1,N \rangle$ in source polygon A a vertex $B(j), j \in \langle 1,M \rangle$ with minimal distance in target polygon B is searched. When sum of all vertex to vertex distances (1) is below threshold we may consider target polygon as unchanged and replace it by reference to its ancestor.

$$err(A,B) = \frac{\sum_{i=1}^{N} \min_{j \in \langle 1,M \rangle} \{dist(A(i),B(j))\}}{N} \quad (1)$$

[1] PSNR - Peak Signal to Noise Ratio

from Proceedings of Spring Conference on Computer Graphics 2006
Bratislava: Comenius University, 2006, p. 177-183. ISBN 80-223-2175-3

This method is quite fast but it has some shortcomings like it is insensitive to local folds at polygon B (Fig. 2). Since the method is asymmetric this problem is easily eliminated by checking distances from polygon B to polygon A and selecting $max(err(A,B), err(B,A))$.

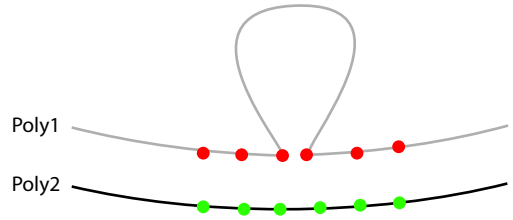

Figure 2: Omitted local fold. Nodes from polygon 2 are being compared to nodes of polygon 1 excluding the fold.

On convex polygons a Hausdorff distance (2) may be used to reveal greatest local deviation. First for each vertex $a \in A$ we found vertex $b \in B$ with minimum distance $dist(a,b)$. The result of $h(A,B)$ is then maximum of previous distances. By taking maximum from $h(A,B)$ and $h(B,A)$ we got Hausdorff distance.

$$H(A,B) = max\{h(A,B), h(B,A)\} \tag{2}$$

where

$$h(A,B) = max_{a \in A}\{\min_{b \in B}\{dist(a,b)\}\}$$

5.2 Advanced shape matching

Contribution of this paper lies in extending basic techniques by adding certain methods, which are capable of detecting even dislocated, rotated and scaled polygons. These methods were used on results from previous techniques.

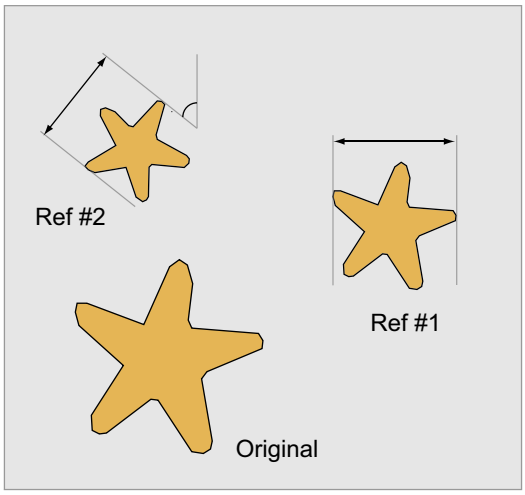

Figure 3: Referencing similar shapes within one frame

As coincident regions from neighbouring frames are matched, we can proceed to more advanced polygon search. Polygon entity may be changed in different ways as it comes along the movie. Since its first occurrence just by appearing in scene from nothing or by coming from the border until it disappears in similar way. The most common inter-frame change is content translation. Another operations include rotation, scaling and shape deformations (see figure 3).

First thing prior to detecting translated, rotated or scaled polygon is to find its proper representation. Reducing classic two dimensional vertex based boundary to 1D function will move our work to well known area of 1D function analysis and synthesis. One method uses turning function $\theta(s)$. Starting from some reference vertex 0 on shape boundary this function goes along the boundary and for each distance s it measures certain angle. Either it is angle from some reference vector (i.e. horizontal line, vector from $\theta(0)$ to centroid, etc.) or it is angle between previous and next segment at each vertex along the boundary. It may be used to shape comparison as proved by [Arkin et al. 1991].

Another approach is to create a function where distance between centroid of the polygon and its vertices is measured. This function $\gamma(\phi)$ will return distance for an angle ϕ between vector from centroid to boundary vertex and base vector. Base vector goes from centroid to first boundary vertex (see Fig. 5)

Similar results with certain quantization may be expected from shape matrix representation [Goshtasby 1985]. Here shape is positioned inside circle which is divided to concentric circles of smaller diameters and to sectors. Matrix is built by sampling shape at the intersections of the circles and radial lines.

This method is suitable for fast shape similarity estimation. We adopted this idea as a modification of log-polar bin histogram [Mori et al. 2005] method. Here we use disc divided to 12 sectors by 5 bins each (Fig. 4 (b)). After a shape is put over this disc the bins will receive count of boundary vertices from the shape. These values are represented by dark-bright scale at histogram (Fig. 4 (c).

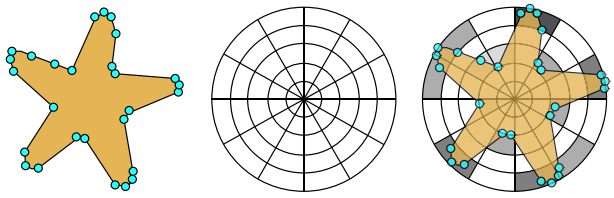

Figure 4: (a) Source shape, (b) Disc with bins, (c) Shape with histogram.

The original concept from [Mori et al. 2005] uses logarithmic scale for diameters of inner circles. Shape is positioned above the disc so as randomly selected boundary vertex is in the center of the disc. Like this it is possible to obtain several histograms for each shape by sampling them at random boundary vertices. Each shape is then equipped by set of its histograms. At shape comparison the correlation between items from histogram sets is examined.

In our case we are creating just one histogram for each shape where the reference point of each shape is being positioned in the center of disc. Here logarithmic scale is of no use as vertices are not expected to occur close to the centroid. Homogenous bin distribution gives better coverage.

Important question is selection of reference point. This point should be bound to the polygon in a way where it is independent of polygon rotation. Center of bounding box does not fulfill this condition but bounding circle and centroid do. Comparing bounding circle and centroid it is obvious that bounding circle is insensitive to amount of vertices used on polygon representation but it is more sensitive

from Proceedings of Spring Conference on Computer Graphics 2006
Bratislava: Comenius University, 2006, p. 177-183. ISBN 80-223-2175-3

to local shape deviation at most distant vertices. When vertices are smartly distributed along the shape the centroid option does better job.

Another question is invariance to scale when using polar bins histogram. Either we may normalize distances by biggest value between vertex and reference point or we may use mean value to estimate appropriate scaling. It is necessary to say that both options are not ideal solution and this problem is object of further research.

5.3 Fine shape comparison

Having candidate shapes from previous step we proceed to fine shape comparison. Here no more quantization of angle and distance is performed. Instead we implemented full transformation to polar coordinates. Coming from the reference point of each shape (centroid) a vector to each boundary vertex is constructed. Length of these vectors and their angle to the base vector are values used to create graph in Euclidean space. Angles up to 360 degrees are covered. After normalization this graph is invariant on shape location and size.

As original polygon is represented by drive points we transfer just these points which are later connected. Projecting result in Euclidean space will produce graph like (Fig. 5).

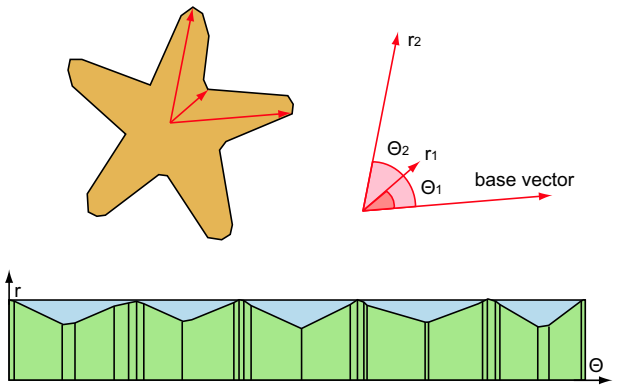

Figure 5: Starfish polygon and its graph represented in polar coordinates projected into Euclidean space.

Unfortunately it is not sufficient in all cases. On rectangle represented by 4 vertices only the result will get flat as distance to each corner from reference point is the same. Mentioned rectangle and circle intersecting its corners will produce the same result. Subdividing parts of the polygon in way that vertices are distributed more uniformly by angle related to reference point solves mentioned shortcoming (see Fig. 6).

As every polygon has its polar representation the next step is to compare these graphs. Correlation measurement is performed using discrete form (4) of Pearson's product-moment coefficient. Result is given by dividing covariance by multiplication of standard deviations σ_x, σ_y of random variables X and Y (3). Final value may range from -1.0 to 1.0, where 1.0 means perfect match, 0 means no match and -1 means perfect inverse match.

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y} \tag{3}$$

$$r_{X,Y} = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n\sum x_i^2 - (\sum x_i)^2}\sqrt{n\sum y_i^2 - (\sum y_i)^2}} \tag{4}$$

If polygons are identical but they differ in point selected as first vertex there will be certain angle between their base vectors. The same situation will occur when shapes are rotated to each other. Polar coordinate graphs are then shifted (see Fig. 6). Use of convolution at polar graph may reveal the angle shift which when used for compensation will get the best result for further correlation test.

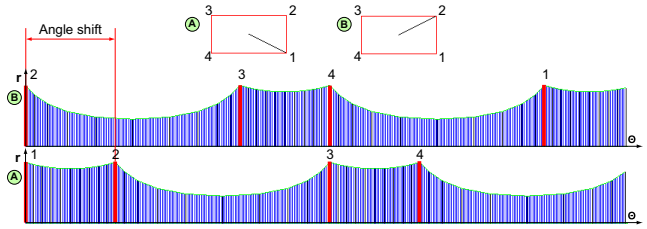

Figure 6: Rectangles with different position of first vertex will result in shifted polar graphs.

Another issue is non convex polygon with any kind of folds where more vectors from reference point to various vertices share the same angle (Fig. 7). This case results in graph where relation between angle and distance value is no longer bijection. Folds on shape are projected to polar coordinate graph. Any kind of correlation cast upon this graph will have to handle the possibility of more results per one angle. Storing values for each angle to 1D array and using these arrays as result will do the work. Items in arrays are entering correlation by their order and if one array has lower cardinality, last value from this array is used with the rest of values from the second array.

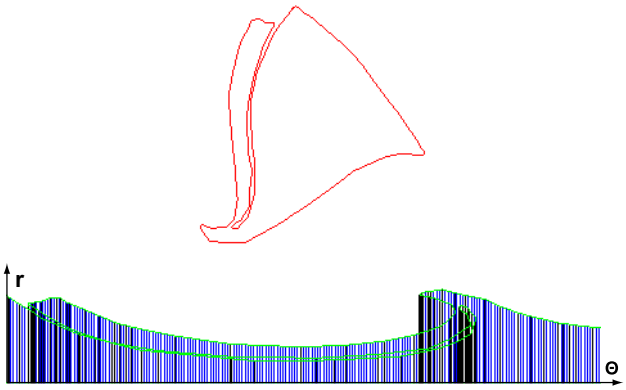

Figure 7: Nonconvex shape produces graph which is not bijection.

Due to its nature this method will find identical shapes and it is invariant of rotation, translation and partially on scale. Invariance on scale needs distance normalization step which may be based on maximum distance or mean distance as mentioned at log-polar bins. Any of this method is relevant only as long as there is not any significant local deviation. If one vertex at compared polygon is relocated to very distant place comparing to its parent shape then scale ratio of entire polar graph will be compromised even if the rest of polygon is just identical to its parent.

Shortcoming of this idea with polar graph lies right in this high dependence of reference point position. Apparently wrong position of this point will produce different graph and no further comparison is possible. Positioning based on centroid is good as long as we are looking for identical shapes represented by very similar vertex distribution along the shape. Refining nodes by method presented in

from Proceedings of Spring Conference on Computer Graphics 2006
Bratislava: Comenius University, 2006, p. 177-183. ISBN 80-223-2175-3

[Sýkora et al. 2005b]) will provide us useable data for very similar shapes, but nothing more. Centroid symmetric shape deviation is one possible way how to have two different shapes with the same centroid position but this example is statistically insignificant.

5.4 Shape vocabulary

Shape matching is not an easy task especially at large data set. Any improvement contributing to lowering computational load is helpful. Classic MPEG-2 based coder works in two phases. First it compares macroblocks of parent frame to neighbouring frames and any pattern match is being coded with motion vector between corresponding areas. As there is not always perfect match then even partial matches are coded this way. A new frame with motion predicted patterns is compared to real frame subsequent to parent. Coding the residual frame of their difference is the second phase. We have the same intention on vectorized video layer. Advantage of vectorized scene is that it does not contain as much data as real video scene. Comparing frame patterns can be performed with huge amount of other frames as shapes usually do not have big memory demands. The only issue is computational load.

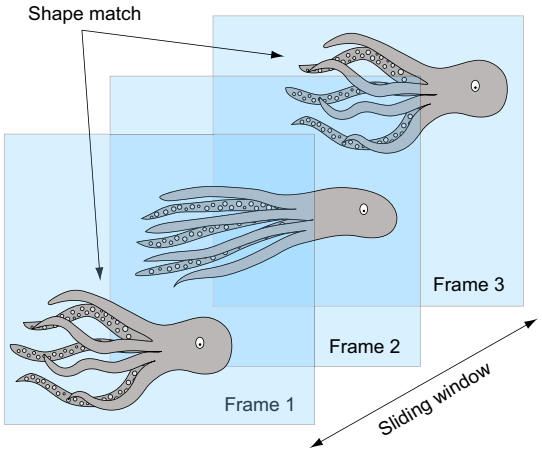

Figure 8: Referencing similar shapes within several frames

Based on this conclusion only certain frame range may be included into polygon comparison. In this level we are considering just non-redundant frames with already referenced identical polygons sharing the same position at consecutive frames. These references are excluded from further processing. Coming out from time available for coding, processing power and scene complexity the processing window of certain frame count is created (see fig. 8). This window covers current frame and certain number of its descendants. Each polygon from current frame is being compared to polygons in frames covered by this processing window. As current frame is finished, window slides forward by one frame and the whole process is being repeated.

Many shapes are replaced by instance of their parent soon after they are reached by processing window. Those shapes are no longer included into comparison therefore if they occur in frame sequence longer than the processing window they are not replaced until their frame becomes current (first) frame of the window. Before this happens they are involved in comparison process vainly. This may be avoided by shape vocabulary which means that any shape successfully replaced by more than once is being added into vocabulary. Items within vocabulary are tested against polygons within processing window along with contents of current window. If any polygon appearance is long-running then it is being replaced as soon as it is reached by processing window. LRU[2]or any other method known from caching algorithms may be used for replacing of shapes within this vocabulary.

5.5 Measurement

All measurements in this paper were performed in order specified by following list.

1. Removing redundant frames - basic technique of selecting frames holding changes.
2. Referencing unchanged polygons - polygons unchanged in any way from their occurrence at previous frames.
3. Referencing polygons within frame - finding and referencing similar shapes within one frame.
4. Referencing polygons between frames - using sliding window and referencing similar polygons in frames covered by this window

First two items are representing basic techniques and the remaining items are advanced matching methods invariant to common transformations. As every method is applied on results coming from previous step we will study the results by order of their acquiring. First lets take a closer look to basic techniques.

Our tests included 6 sample clips containing from 157 to 301 frames. After detecting and removing frame redundancy we got *CoreF* frames with total of *CoreV* vertices as referred by (table 1). Detecting polygons unchanged in any way between consecutive frames and referencing them will save *CopiedV* vertices in these clips.

Clip	Frms	CoreF	[%]	CoreV	CopiedV	[%]
1	157	41	26.11	330679	16521	5.00
2	250	33	13.20	215053	17137	7.97
3	301	38	12.62	299418	34524	11.53
4	250	69	27.60	505435	69519	13.75
5	157	37	23.57	295945	26233	8.86
6	300	82	27.33	572203	24615	4.30

Table 1: Measurement of basic techniques. Frms - frames in original clips, CoreF - frames holding changes, CoreV - number of vertices in core frames, CopiedV - number of vertices in referenced polygons unchanged to previous frames

We are focusing on non-redundant frames marked as *Core* which are representing only about 10-30% of total frame count. All further tests are performed just on these frames. At this moment we were handling about 200-600 thousands of Core vertices stored in polygons.

Comparing consecutive frames for unchanged polygons (sharing the same location within frame) we found that up to 14% of core vertices were in polygons replaced by reference to their parent.

Second measurement (see table 2) is based on advanced shape matching technique performed on data remaining from previous steps. Here not exactly shape identical polygons were searched but a tolerance of 5% has been given to shape matching function. It should be enough to eliminate inaccuracy of rotated or scaled polygons and still ensure sufficient shape match.

[2]Least Recently Used algorithm - discards the least recently used items first.

from Proceedings of Spring Conference on Computer Graphics 2006
Bratislava: Comenius University, 2006, p. 177-183. ISBN 80-223-2175-3

[%]					
Clip	IVerts	Wind2	Wind5	Wind10	Wind20
1	0,17	3,61	4,55	5,00	5,53
2	0,19	3,80	4,48	4,85	5,10
3	0,42	5,71	8,08	11,33	11,97
4	1,00	6,62	12,09	12,46	12,74
5	0,11	4,82	5,79	6,19	6,60
6	0,04	5,56	16,01	19,81	24,99

Table 2: Measurement with search for similar polygons. IVerts - percentage of core vertices saved by referencing polygons within one frame, Wind2 - percentage of core vertices saved by sliding window covering 2 frames, Wind5 - sliding window covering 5 frames, Wind10 - 10 frames and Wind20 - 20 frames

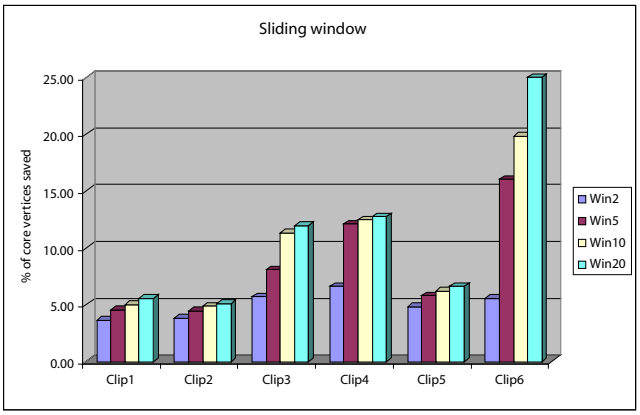

Figure 9: Percentage amount of core vertices in polygons successfully replaced in relation to sliding window size.

Finding similar polygons within one frame (intra-frame method) proved itself as not very efficient in common video clip. By *IVerts* column in table 2 maximum of 1% of vertices from core frames were replaced. In some cases of i.e. flying balloons or bubbles it might be more useful.

Inter-frame method with sliding window shows much better results. About 4-7% of core vertices were replaced when sliding window covered two consecutive frames (*Win2* column in table 2). It means that there are still many almost identical polygons in two neighbouring frames but they are at different positions and they may be also rotated or scaled. Interesting values were produced when size of sliding window has changed. We tested it for size of 2,5,10 and 20 frames. Graph 9 shows the relation between window size and number vertices of successfully replaced polygons. No significant change due to window size is revealed at clips 1,2 and 5. Clips 3,4 and above all clip 6 demonstrated strong influence of window size.

At clip 4 the tested method gives generally better results at window size of 5 frames but bigger size has no further effect. At clip 3 the gain of saved vertices stopped at window size of 10 frames but at clip 6 it seems that we got better values with increasing the window size. An answer for this behavior lies in the content. Clip 6 contains quite long frame sequence where the main animated character is laughing. This is being animated by shaking the main object with many remaining objects bound to it. Although there are minor shape changes at smaller objects, many polygons are being repeated several times. Bigger window can found relations between these polygons within bigger frame range therefore it is recommended to have this window as big as it can contain all similar frames.

Proper window size can be managed by method where clip is being processed several times with window size increasing in each iteration. These iterations are then being performed as long as there is significant gain of vertex saving. Other side of coin is CPU demands. We measured that average time for frame to frame comparison took about 1 second at AMD AthlonXP 2000+ with 1GB RAM.

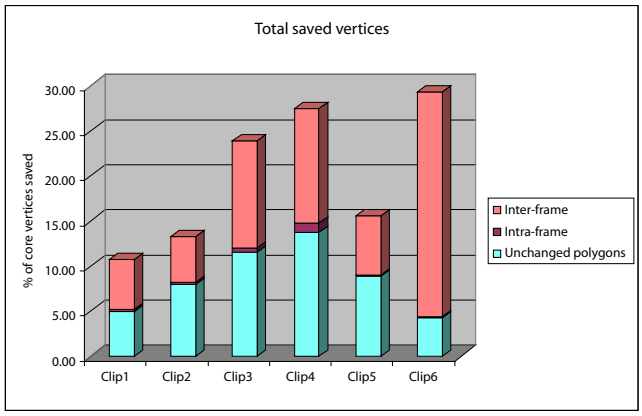

Figure 10: Percentage amount of core vertices in polygons successfully replaced by all used techniques together.

All proposed techniques for replacing similar shapes are used in one chain. Their contribution to save core vertices is aggregated into final result (Fig. 10). Searching for similar polygons within one frame (intra-frame method) did not saved much vertices but extending the search at consecutive frames did good job. Sliding window technique with proper size adjustment may saved up to 25% of total vertices in core frames. When results coming from basic and advanced shape matching techniques are added together then about 10-30% of core vertices are saved which is good result.

5.6 Data coding

Each frame has hierarchical data organization where polygon vertices are consuming majority of used space. Hierarchy of frame-layer-regions structures is being stored into final data stream along with special keywords at the beginning of each element. I.e. frame keyword indicates if following frame is just time placeholder as there is no change to its ancestor or if there is any global motion compensation vector for whole scene or if postprocessing special effect like fading out or fading in will occur, etc. Very similar keywords are available for layers and regions. By region the typical example is when there is only reference to its parent region represented by frame-layer-region code. Angle, scale and motion vector is used to compensate difference to its parent when necessary. Each keyword indicates which type of data chunk will follow.

As vertices are the biggest space consumer then just by transcoding floating number representation into integer values a lot of space is saved. Polygon vertices resolution is not very sensitive therefore unsigned short integer values are sufficient. Polygons are being stored with essential vertices only. Their fine shape can be reconstructed by player by subdividing polygon parts or by interpolation in the case of spline representation.

For more complex scene structure, where scene decomposition to objects can be achieved at satisfying rate it is suitable to use MPEG-4 based approach [Concolato et al. 2003]. Here BIFS[3]encoding scheme is used to describe both objects in the scene and their inter-

from Proceedings of Spring Conference on Computer Graphics 2006
Bratislava: Comenius University, 2006, p. 177-183. ISBN 80-223-2175-3

frame operations. Using linear quantization and predictive coding there is an efficient way to code data into low bitrate stream.

6 Conclusion and future work

With the possibility of playing streaming video on mobile devices and with rapid coming of MPEG-4 implementations it is obvious that vectorized video has its future safe. Turning old classic cartoons into digital vectorized form will help archiving this content in perfect quality and with low space demands. Due to its vector nature this representation is very scalable and therefore useable in various environments.

In this paper we focused on vectorized data at non-redundant frames which have already been processed by simple polygon matching techniques. When using more sophisticated methods for shape similarity it is important to mention that some intershape deviations should be tolerated because of certain error possibility coming from common shape transformations. But it must be considered carefully as loosing even small detail at exposed area may harm entire clip. Due to technique of vectorization we do not have true overlapping layers but adjacent shapes only. Inaccurate shape replacement may cause an unhandled hole between animated objects. In our tests we allowed up to 5% error at result of correlation. It is also necessary to avoid cross-referencing so that replaced shape cannot be compared to any other shape anymore. Violation of this rule may lead to error accumulation and inaccuracy of shape matching.

Results coming from essential contribution of this paper showed that it is useful to involve advanced matching methods. They were designed to be applied on polygons intact by basic matching techniques. Beyond not so well demonstration of intra-frame technique we got quite good vertex savings at inter-frame method with sliding window. From 5% to 25% of core vertices were saved by polygon references. Together with basic detection of unchanged polygons we got about 10-30% vertices saved which is significant success.

Although it cannot be said that there is general rule which may guarantee efficiency this step may contribute to lower computational and bitrate requirements in the further processing.

6.1 Future work

Proposed methods still need a lot of improvements. Starting with speed-ups and ending with extending the reliability and possibilities. Used techniques are just one of steps which need to be taken to get efficient coding of vector based cartoon video data. Object of further research may be shape matching based on features. Simply by comparing few features and their relation at shape pair it may be possible to exclude compared object from further comparison and increase processing speed. Shape features may also help to find the way how exactly shapes are different. In some cases there are shapes with stronger local deviation but almost the same at rest of their body. Detection of these parts is important for coding shape differences. Similar parts may be referenced here but coding of the differences should be more investigated. How big the deviation should be to still worth being coded like shape difference instead of considering it as completely new object? It is one of possible questions. Searching for local shape changes may be performed using shape contexts proposed in [Mori et al. 2005]. Coding shape differences will be next step how to build efficient coding scheme similar to MPEG2 coding used on various content.

[3] BInary Format for Scenes - part of MPEG 4 standard

References

ARKIN, E. M., CHEW, L. P., HUTTENLOCHER, D. P., KEDEM, K., AND MITCHELL, J. S. B. 1991. An efficiently computable metric for comparing polygonal shapes. *IEEE Trans. Pattern Anal. Mach. Intell. 13*, 3, 209–216.

BORGEFORS, G. 1984. Distance transformations in arbitrary dimensions. *Computer Vision, Graphics, and Image Processing 27*, 321–345.

CONCOLATO, C., DUFOURD, J.-C., AND MOISSINAC, J.-C. 2003. Creating and encoding of cartoons using MPEG-4 BIFS: Methods and results. *IEEE Transactions on Circuits and Systems for Video Technology 13*, 11, 1129–1135.

EBRAHIMI, T., AND HORNE, C. 2000. MPEG-4 natural video coding – An overview. *Signal Processing: Image Communication 15*, 4–5, 365–385.

GOSHTASBY, A. 1985. Description and descrimination of planar shapes using shape matrices. *IEEE Transactions on PAMI*, 7, 738–743.

GROUP, M.-. V., 1995. Information technology – generic coding of moving pictures and associated audio: Part 2 – Video. International Standard: ISO/IEC 13818-2.

GROUP, M.-. V., 1998. Generic coding of audio-visual objects: Part 2 – visual. International Standard: ISO/IEC 14496-2.

HORNG, R., AND AHUMADA, A., 1995. A fast dct block smoothing algorithm.

KWATRA, V., AND ROSSIGNAC, J. 2002. Space-time surface simplification and edgebreaker compression for 2d cel animations. *International Journal on Shape Modeling 8*, 119–137.

LONCARIC, S. 1998. A survey of shape analysis techniques. *Pattern Recognition 31*, 8, 983–1001.

MORI, G., BELONGIE, S., AND MALIK, J. 2005. Efficient shape matching using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence 27*, 11, 1832–1837.

SALOMON, D. 1998. *Data compression: The complete reference*. Springer Verlag.

SÝKORA, D., BURIÁNEK, J., AND ŽÁRA, J. 2005. Colorization of black-and-white cartoons. *Image and Vision Computing 23*, 9, 767–782.

SÝKORA, D., BURIÁNEK, J., AND ŽÁRA, J. 2005. Video codec for classical cartoon animations with hardware accelerated playback. In *Advances in Visual Computing*, vol. 3804 of *Lecture Notes in Computer Science*, Springer, 43–50.

TANGELDER, J. W., AND VELTKAMP, R. C. 2004. A survey of content based 3d shape retrieval methods. In *SMI '04: Proceedings of the Shape Modeling International 2004 (SMI'04)*, IEEE Computer Society, Washington, DC, USA, 145–156.

YEN, W.-C., AND TAI, S.-C. 2005. Dct-based image compression using wavelet-based algorithm with efficient deblocking filter. In *ICIS '05: Proceedings of the Fourth Annual ACIS International Conference on Computer and Information Science (ICIS'05)*, IEEE Computer Society, Washington, DC, USA, 489–494.