Digitally Animated Overture using Skewing in Two Dimensions.

M. Nuruzzaman, Ph.D.

Electrical Engineering Department, King Fahd University of Petroleum and Minerals,
Dhahran 31261, Saudi Arabia.

E-mail: nzaman@kfupm.edu.sa

ABSTRACT

Recently, overture effect generation is reported as employing polar equations in spatial domain of digital images. The same effect can be simulated by applying skewing of the affine transform which is the subject matter of this paper. Addressing general affine transform, skewing is introduced with theoretical basics and applied to digital images. A pixel domain based algorithm on finite difference image model simulates the overture effect subject to skewing with varying cases. The PC based algorithm is proven to be effective without extra hardware requirements for digital animations.

(Keywords: digital animation, affine transform, overture effect, special effect, 2D animation)

INTRODUCTION

A digital movie segment is a collection of frames or still pictures typically 24 frames per second. Virtual movie segment simulation requires virtual generation of the digital image frame for that reason the combination of two skills is important – 1) practical events or how it is taking place and 2) computer code writing expertise to turn the event to a realistic movie module. When two dissimilar image frames are displayed consecutively in a digital movie, the visual perception we receive is called an overture effect.

Overture effect simulation is an open problem as introduced in [1]. Concentration had been given in [1] on studying the overture effect generation between two digital images through the processing of the polar equations in spatial domain. That technique is not the sole method to implement the animation. User-defined skewing of the affine transform [2]-[4] generates a similar overture effect which we plan to introduce in this work.

With the availability of higher speed processors, digital animation would be a broad field, as the digital animation industry is growing literally across all corners of the globe. However, there is some exemplary research in the field as follows: fire animation [5]-[6], simulation of augmented reality [7]-[8], real time three dimensional motion capture systems [9], compelling character animation [10], nature based system simulation and evolution [11]-[13], etc.

Digital movie frame and PC-based animation models both are chosen from [1]. Geometric transform is a generic transform and covers a lot of elementary image manipulations such as translation, rotation, skewing, and scaling of digital images, or a combination of all. In this short context it is not feasible to study all elements of the transform to generate the digital animation. We confine our exploration only to digital image skewing.

Most digital animation concerns event-based computer implementation, so possibilities are infinite, and one of those possibilities is the overture generation. Overture generation in two dimensional digital animations is gaining importance for obvious reasons. A brief introduction of digital movie frame and overture generation is provided by applying the notions and concepts of the discrete mathematics and the data structure of a digital movie both to the context of storage and pixel processing in the following section. Addressed in the subsequent sections are the geometric transform basics, skewing based overture algorithm, and the simulation results, respectively.

DIGITAL MOVIE FRAME AND OVERTURE

A digital image frame is discrete, both to the context of image space and intensity. Mathematically discrete two dimensional intensity function $f[m,n]$ represents a monochrome digital image. Another way of explaining the $f[m,n]$ is that it is simply a rectangular matrix [1]. Assuming that the digital image area has $M \times N$ pixels, then the coordinate system in image domain denotes any given pixel as (m,n). For the image, the pixel coordinate variation is as follows:

$$\begin{Bmatrix} m \text{ varies from } 0 \text{ to } M-1 \\ n \text{ varies from } 0 \text{ to } N-1 \end{Bmatrix}$$

clearly m or n is positive integer. Any practical digital image by discrete mathematics turns to two dimensional intensity function $f[m,n] \in 2^p$ where p is any positive integer.

A true color digital image has three color component matrices – red, green, and blue, symbolically $r[m,n]$, $g[m,n]$, and $b[m,n]$, respectively, each of which is identical in matrix size (or $M \times N$) and bear aforementioned meanings pertaining to the discrete space variables m and n therefore any digital movie frame is mathematically just the matrix triplet:

$$\begin{Bmatrix} r[m,n] \\ g[m,n] \\ b[m,n] \end{Bmatrix}.$$

A digital movie segment is born by placing such triplet one after another. If there are P triplets and the hardware system supports f_r triplets per second, the duration of the movie segment becomes $\frac{P}{f_r}$. Digital animation making means generation of the P triplets based on the storyboard for our case the storyboard is the overture formation.

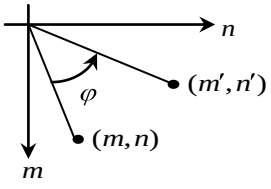

Figure 1: An Image Pixel (m,n) Undergoes to a Geometric Transform to take the New Position (m',n')

Overture details are found in [1]. Two dissimilar image frames render visual sense of discontinuity. We determine the skewed image of the latter by applying affine transform and minimizing the visual discontinuity by different skew angles. Within the bounds of affine transform, horizontal or vertical skewing can be applied to generate different overtures which will be demonstrated and realized in latter sections.

GEOMETRIC TRANSFORM BASICS OF A DIGITAL IMAGE

The geometric transform is part and parcel of digital image processing especially in computer graphics. The principal reason that lies in the geometric transform relates to fitting an image on a 2D monitor screen and in a finite number of pixels. A geometric transform primarily involves two operations namely:

$$\begin{Bmatrix} \text{pixel coordinate transformation} \\ \text{color or gray level interpolation} \end{Bmatrix}.$$

Figure 2: A Picture.

Figure 3: The Image man.jpg is Skewed Vertically by 25^0

For the color or gray level interpolation, usually one of the three methods namely, $\begin{Bmatrix}\text{nearest neighborhood interpolation}\\ \text{linear interpolation}\\ \text{bicubic interpolation}\end{Bmatrix}$ is applied [2]-[4].

By definition, the geometric transform T is a vector function that maps the pixel coordinates (m,n) of a digital image function $f[m,n]$ to the pixel coordinates (m',n') of another digital image function $f[m',n']$ as illustrated in Figure 1. The mapping requirement is completely user-defined (the way user likes to view the image). The (m',n') and $f[m',n']$ are related to the pixel coordinate transformation and gray level interpolation, respectively. The problem statement here is to find (m',n') and $f[m',n']$ from the given (m,n) and $f[m,n]$ respectively.

Skewing or Shearing a Digital Image

The simpler type of geometric transform is the affine transform and is defined in general in terms of three unknown constants for each directed pixel as follows:

vertically directed: $m' = a_0 + a_1 m + a_2 n$ and
horizontally directed: $n' = b_0 + b_1 m + b_2 n$.

If a given digital image $f[m,n]$ undergoes to skewing, we find the affine transform constant sets $\begin{Bmatrix}a_0\\ a_1\\ a_2\end{Bmatrix}$ and $\begin{Bmatrix}b_0\\ b_1\\ b_2\end{Bmatrix}$ from the user definition.

The digital image $f[m,n]$ can be skewed or sheared in the m or n direction by an angle φ (also called the vertical and the horizontal shearings, respectively) and their formulations are as follows:

Vertical Skewing: The pixel coordinate transform relationship: $\begin{Bmatrix}m' = m + n\tan\varphi\\ n' = n\end{Bmatrix}$, T vector: $\begin{bmatrix}1 & \tan\varphi\\ 0 & 1\end{bmatrix}$, and T^{-1} vector: $\begin{bmatrix}1 & -\tan\varphi\\ 0 & 1\end{bmatrix}$ where T^{-1} is the inverse matrix of the vector matrix T.

Horizontal Skewing: The pixel coordinate transform relationship: $\begin{Bmatrix}m' = m\\ n' = m\tan\varphi + n\end{Bmatrix}$, T vector: $\begin{bmatrix}1 & 0\\ \tan\varphi & 1\end{bmatrix}$, and T^{-1} vector: $\begin{bmatrix}1 & 0\\ -\tan\varphi & 1\end{bmatrix}$.

How to Calculate a Skewed Image?

Any pixel coordinate (m,n) of the given image $f[m,n]$ has the integer gray level 2^p where the p is an integer. User-supplied operator T is applied to compute matrix multiplication of $T \times \begin{bmatrix}m\\ n\end{bmatrix}$ which is $\begin{bmatrix}m'\\ n'\end{bmatrix}$ and the multiplication occurs for every (m,n) in $f[m,n]$.

One difficulty arises with $\begin{bmatrix}m'\\ n'\end{bmatrix}$ that is the newly obtained coordinates become fractional due to the computation. We round the $\begin{bmatrix}m'\\ n'\end{bmatrix}$ to its nearest integer since the digital image space also has integer numbered pixels. The $f[m,n]$ gray level value should be the $f[m',n']$ for the rounded $\begin{bmatrix}m'\\ n'\end{bmatrix}$. When performed, this operation shows geometric distortion in the image that is why T^{-1} is multiplied with the rounded $\begin{bmatrix}m'\\ n'\end{bmatrix}$. The last multiplication also results fractional $\begin{bmatrix}m\\ n\end{bmatrix}$ and which also needs rounding due to discrete m - n domain. This $f[m,n]$ gray level value for rounded $\begin{bmatrix}m\\ n\end{bmatrix}$ corresponds to $f[m',n']$ – actually we explained the nearest neighborhood gray level interpolation technique which is chosen for the image skewing. Any $f[m,n]$ outside the given domain of $\begin{bmatrix}m\\ n\end{bmatrix}$ is set to 0 gray level.

Example on Skewing a Digital Image

The image man.jpg of Figure 2 is to be skewed vertically by 25^0.

The image man.jpg is originally an RGB image i.e. it has $\begin{Bmatrix} r[m,n] \\ g[m,n] \\ b[m,n] \end{Bmatrix}$ triplet representation. The man.jpg image dimension is 826×1152 i.e. M =826 and N =1152.

We extract the $r[m,n]$, $g[m,n]$, and $b[m,n]$ components from the triplet. The affine matrix T for the problem becomes $\begin{bmatrix} 1 & \tan 25^0 \\ 0 & 1 \end{bmatrix}$ and we apply the affine matrix T operator to every pixel on each of the three component images.

Earlier mentioned nearest neighborhood interpolation is exercised with $T^{-1} = \begin{bmatrix} 1 & -\tan 25^0 \\ 0 & 1 \end{bmatrix}$ for every pixel to each image component and later we combine the three component images to form the skewed triplet $\begin{Bmatrix} r[m',n'] \\ g[m',n'] \\ b[m',n'] \end{Bmatrix}$. Figure 3 shows the 25^0 vertically skewed image of the image in figure 2.

As seen in Figure 3, unwanted $\begin{Bmatrix} r[m',n'] \\ g[m',n'] \\ b[m',n'] \end{Bmatrix}$

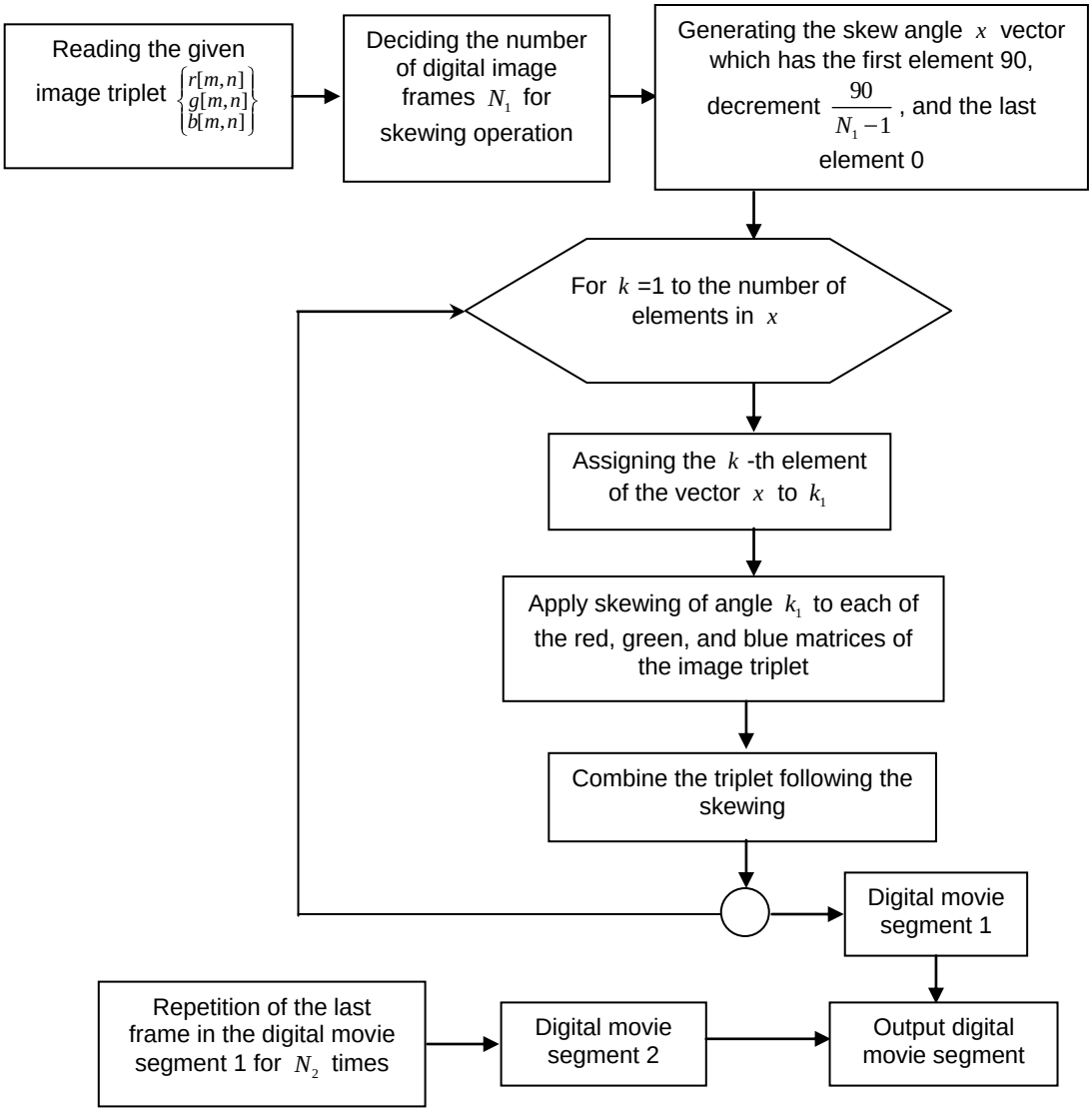

Figure 4: Flow Chart of Digital Animation for Skewing Based Overture over User-Defined Background Color

intensity values are set to 0 which appear as the triangular black region in the figure. During calculation when T^{-1} is applied on (m',n') to find (m,n) for the nearest neighborhood interpolation, a checking is conducted to determine the triangular black region in figure 3 as follows: $f[m',n']$=0 for outside the interval $0 \le m \le M-1$ or $0 \le n \le N-1$ considering the image of the pixel size $M \times N$.

SKEWING BASED OVERTURE ALGORITHM

Skewing based overture algorithm can be defined broadly in two categories: (a) skewing over user-defined background color and (b) skewing on an existing image frame. Each algorithm is explained in the following.

Skewing Over User-Defined Background Color:

Figure 4 shows the flowchart of the overture algorithm when skewing over user-defined background color is conducted and which requires the following steps:

Step 1: The given image is acquired in order to have the RGB triplet $\begin{Bmatrix} r[m,n] \\ g[m,n] \\ b[m,n] \end{Bmatrix}$, from which the triplet component dimension $M \times N$ is obtained.

Step 2: The number of digital image frames N_1 for skewing operation is decided.

Step 3: The skew angle vector x is generated which has the first element 90, decrement $\frac{90}{N_1 - 1}$, and the last element 0 assuming vertically downward skewing.

Step 4: We enter into a programming loop where the loop index k changes from 1 to the number of elements in x. For every index k inside the loop, we carry out the following:

(a) assign the k-th element of the vector x to k_1

(b) apply skewing of angle k_1 to each of the red, green, and blue matrices of the image triplet

(c) combine the triplet following the skewing

(d) upon completing the loop the digital movie segment 1 is born.

Step 5: In a digital movie a picture frame duration is so short (in split second range) that one image frame is not perceived distinctively. Continual perception is obtained by repeating the last image frame for few frames say N_2 times.

Step 6: Finally we place the steps 4 and 5 derived movie segments in order to form the complete movie module.

Even though in step 3 we considered vertically downward skewing as an example yet other skewing is obtainable: considering first element as 0, increment as $\frac{90}{N_1 - 1}$, and the last element as 90 provides vertically upward skewing. Not only vertically, similar directional vector element flipping also renders left or right skewing horizontally.

Regarding still frame repetition of Step 5, consistency is utmost priority for smooth digital movie module production. The last movie frame resulting from the skewing must coincide with the still image frame. If the skewing results squeezed frame to full frame, the still image repetition must be at the end of the skewing otherwise at the beginning of the skewing. In the algorithm description we applied the repetition at the end of the vertical skewing.

In a digital movie the image frame pixel size is fixed. Due to skewing unwanted region appears which is explicit as black triangular region in figure 3. If $f[m,n] \in 2^p$ where p =8 as an example, the intensity value of $f[m,n]$ is between 0 and 255. In most digital imagery 0 intensity is taken as black i.e. $r[m,n]$=0, $g[m,n]$=0, and $b[m,n]$=0 for a true color movie frame. The unwanted region due to skewing can be filled with pre-decided color for instance with white $r[m,n]$=255, $g[m,n]$=255, and $b[m,n]$=255, with red $r[m,n]$=255, $g[m,n]$=0, and $b[m,n]$=0, etc.

Figure 5: Picture of two Children.

Figure 6: Picture of two Children is Skewed at an Angle 60^0 Vertically

Skewing Over User-Provided Image Frame

When it comes to filling the unwanted region by another image, we need an algorithm to do so. In fact an algorithm should be devised for imposing a foreground over a given background.

Let us call the foreground and the background triplets as $\begin{Bmatrix} r_1[m,n] \\ g_1[m,n] \\ b_1[m,n] \end{Bmatrix}$ and $\begin{Bmatrix} r_2[m,n] \\ g_2[m,n] \\ b_2[m,n] \end{Bmatrix}$, respectively.

First we fill the unwanted region of the foreground image by 0 or black color i.e. $r_1[m,n]=0$, $g_1[m,n]=0$, and $b_1[m,n]=0$ for the triangular region due to skewing. Then exclusive or operation (xor) is performed on background and foreground. Next the resulting image is multiplied ($\otimes$) with the background. Incidentally the multiplication means pixel by pixel in the image domain. After that the multiplied image is added with the foreground to obtain the consecutive movie frame. For instance the mathematical operation for the red component image frame we write symbolically as follows:

$$r_2[m,n] \otimes \text{xor}(r_2[m,n] , r_1[m,n]) + r_1[m,n]$$

Similarly the other two component frame operations become:

$$g_2[m,n] \otimes \text{xor}(g_2[m,n] , g_1[m,n]) + g_1[m,n]$$

and

$$b_2[m,n] \otimes \text{xor}(b_2[m,n] , b_1[m,n]) + b_1[m,n]$$

for the green and the blue images, respectively.

SIMULATION RESULTS

This section presents the simulation results which demonstrate the application of the algorithm. In contrast conventional papers our result is a digital video so paper display of the digital video is not possible yet we include one sample frame out of the digital video just to prove the worthiness of the algorithm.

As explained in the algorithm, two types of animations are realized – over user-defined background color and image frame.

For the first type we chose the picture of two children (Figure 5) as foreground which is to be skewed vertically over red background color. The RGB triplet of the two children has the pixel size 600×800 with each color level in [0,255]. We chose 7 frames for the skewing i.e. $N_1=7$ and 3 frames for the repetition i.e. $N_2=3$. Total frame number in the movie is $N_1+N_2=10$. The red color background has the color coding $r[m,n]=255$, $g[m,n]=0$, and $b[m,n]=0$. The 3rd frame in the digital video is shown in Figure 6. The skew angle vector x contains the angles {90^0, 75^0, 60^0, 45^0, 30^0, 15^0, 0^0}. Figure 6 depicted movie frame corresponds to 60^0 vertical skew angle in the red background. The animated movie module has been given the name movie1.avi which has the file size 14.065 MByte without any compression.

The second type animation needs foreground and background for which we chose the picture of two children and the water lily image of the Figure 7, respectively. For the skewing let us choose the vertical one with 7 frames therefore all skew angles are identical with those of the red color background. The RGB triplet of the water lily image also has the pixel size 600×800 with each color level in [0,255]. Particularly in this type we need to repeat the background several times which we did by 3 frames for consistency reason.

Figure 7: Picture of Water Lily.

Figure 8: A Movie Frame with Skewed Water Lily Foreground by 60^0 over two Children Background.

To preserve the consistency for the end movie module we repeated the foreground by 3 times so altogether we have 3+7+3=13 frames in the generated movie. Out of the 13 frames, the 6^{th} frame is shown in Figure 8 which corresponds to the 60^0 skew angle of the water lily image. The movie module we animated has been given the name movie2.avi which has the file size 18.284 MByte without any compression.

Several movie modules are generated to test the algorithm's effectiveness:

- movie3.avi: for vertical skewing over green background color
- movie4.avi: for vertical skewing over blue background color
- movie5.avi: for horizontal skewing of the two children image over red background color
- movie6.avi: for horizontal skewing of the water lily image over two children image

For space reason and inherent similarities, we did not include the description of the last four digital movies. The reader may obtain all above mentioned digital video by making a request at nzaman@ymail.com or nzaman@ kfupm.edu.sa. Microsoft Windows Media Player or any other media player which supports avi format can be used to watch the generated digital video.

CONCLUSION

A part of geometric transform, skewing algorithm, is applied to generate the overture based two dimensional digital animation. Skewing may occur in two background forms – fixed color background and image frame background. An algorithm linking both types is thoroughly explained in the movie frame pixel domain. Both animation types are realized considering two RGB type images with varying skewing cases e.g. horizontal or vertical. The algorithm is proven to be effective not only for paper-cited images rather any digital movie events wherever RGB image type is maneuvered. The algorithm will add some value to animation business due to its usefulness. One distinctive feature of the algorithm is that digital movie is generated without extra hardware. The digital video generation is realized on RGB image format but the algorithm equally works for other types of images for instance indexed, intensity, or bitmapped ones.

ACKNOWLEDGEMENT

The author acknowledges King Fahd University of Petroleum and Minerals (KFUPM) for providing the library and computer facilities necessary for conducting this research.

REFERENCES

1. Nuruzzaman, M. 2009, "Translation Overture Effect through Polar Equations in Two Dimensional Digital Animations". *Pacific Journal of Science and Technology.* 10(1):300-309.

2. Nuruzzaman, M. 2005. *Digital Image Fundamentals in MATLAB*. Author House*:* Bloomington, IN.

3. Gonzalez, R.C. and Wintz, P. 1987. *Digital Image Processing. Second Edition.* Addison-Wesley Publishing: Boston, MA.

4. Russ, J.C. 1999. *The Image Processing Handbook. Third Edition*. CRC Press: Boca Raton, FL.

5. Lee, H., L. Kim, M. Meyer, and M. Desbrun. 2001. "Meshes on Fire*"*. *Proceedings of the Eurographics Workshop on Computer Animation and Simulation*, September.

6. Hargrove, W.W. 2000. "Simulating Fire Patterns in Heterogeneous Landscapes". *Ecological Modeling*.

7. Koller, D., G. Klinker, E. Rose, D. Breen, R. Whitaker, and M. Tuceryan. 1997. "Real-Time Vision-Based Camera Tracking for Augmented Reality Applications". *Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST-97)*. September. 87-94.

8. Klinker, G.J., K.H. Ahlers, D.E. Breen, et al. 1997. "Confluence of Computer Vision and Interactive Graphics for Augmented Reality". *Teleoprerations and Virtual Environments – Special Issue on Augmented Reality.* August. 6(4):433-451.

9. Witkin, A. and Z. Popović. 1995. "Motion Warping". *Computer Graphics Proceedings, SIGGRAPH* 95. Los Angeles, CA. August. 6-11.

10. Ji, Q. 2005. "Weight Computation for Motion Blending". *International Conference on Computer Graphics, Imaging and Vision (CGIV05)*. Beijing, China. July 26-29. 144-150.

11. Haginoya, R., H. Aoyama, K. Honda, and K. Odaka. 2004. "Automatic Animation Generation from Natural Languages". *Proceedings of the International Conference on Computer Graphics, Imaging and Visualization (CGIV04),* School of Mathematical Sciences, USM.103-109.

12. Beer, R.D., R.D. Quinn, H.J. Chiel, and R.E. Ritzmann. 1997. "Biologically Inspired Approaches to Robotics". *Communications of the ACM*. 40(3):31-38.

13. Pina, A. and F.J. Serón. 2000. "Behaviour Modelling, Computer Animation and Ecology". *Proceedings of the IASTED International Conference of Computer Graphics and Imaging.* November 19-23. Las Vegas, NV. 313-318.

SUGGESTED CITATION

Nuruzzaman, M. 2011. "Digitally Animated Overture using Skewing in Two Dimensions". *Pacific Journal of Science and Technology*. 12(1): 252-259.