Object Resizing: A Subtle Tool in Digital Animation

M. Nuruzzaman, Ph.D.

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

E-mail: nzaman@kfupm.edu.sa
mzamandr@gmail.com

ABSTRACT

Principally this paper addresses digital animation employing image object resizing. The object definition belongs to a still digital image. A character, text, photographic picture, etc., are snap-shotted digitally without the involvement of a camera and a tiny movie segment is generated from various objects. A general flowchart is presented as a means of digital movie making through the application of object resizing. A composite video from these modular movies are animated at the end.

(Keywords: image animation, image scaling, image resizing)

INTRODUCTION

Never have we been so engaged with digital movies, as we are today. In most movies, producers capture real life scenes with a digital camera and manipulate them frame by frame if it is necessary. Alternatively, computer image processing techniques and tools are applied to generate the visual effects without the physical use of a camera which we introduce in this paper employing image resizing.

In recent times, notable works on digital image animation are outlined in [1], [2]. As an open problem, algorithmic approach does not just include translation or polar equation centric animation. Whatever method we apply to generate digital imagery, if it suits to given pixel dimension and frame rate, a movie segment is born that yields the attractiveness of digital animation.

In today's world, line between actuality and virtual reality is becoming closer than ever before. Creative generation or cost effective production of highly realistic movies requires a lot of image processing tools for which image resizing is a fantastic manipulation.

Image resizing is a geometric transform which has an established computing framework [3]-[6]. It goes without saying that the transform is a calculation intensive process. Availability of higher speed processor bypasses this calculation intensity.

A rational query is, what gain would be achieved by image resizing? The answer is that physical camera zooming action is simulated by image resizing so why not to apply the tactic in digital image animation. Our technique demonstrates the combining of image resizing in the spatial domain in frame by frame approach.

IMAGE RESIZING IN SPATIAL DOMAIN

Image resizing is one kind of affine transform which in general is defined in terms of three unknown constants for each directed pixel as follows:

horizontally directed: $m' = a_0 + a_1 m + a_2 n$

and

vertically directed: $n' = b_0 + b_1 m + b_2 n$.

Where (m,n) is image pixel coordinate with the coordinate convention of [6]. If a digital image $f[m,n]$ undergoes to a scaling, 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 image scaling has m and n directed components let us say a and b, respectively, both of which can be more or less than 1. For the proposed image resizing the affine transform constant sets are $\begin{Bmatrix} a_0 \\ a_1 \\ a_2 \end{Bmatrix} = \begin{Bmatrix} 0 \\ a \\ 0 \end{Bmatrix}$ and $\begin{Bmatrix} b_0 \\ b_1 \\ b_2 \end{Bmatrix} = \begin{Bmatrix} 0 \\ b \\ 0 \end{Bmatrix}$.

We discovered that image scaling through the constants a and b alters the image size in accordance with table A description.

Table 1: Image Resizing based on the Scale Factors a and b.

Horizontal stretching: a >1 and b =1	the image area increases in the horizontal direction while vertical dimension of the image remains fixed
Horizontal contraction: a <1 and b =1	the image area decreases in the horizontal direction while vertical dimension of the image remains fixed
Vertical stretching: a =1 and b >1	the image area increases in the vertical direction while horizontal dimension of the image remains fixed
Vertical contraction: a =1 and b <1	the image area decreases in the vertical direction while horizontal dimension of the image remains fixed
Overall image area stretching: a >1 and b >1	the image area increases both in the horizontal and vertical directions
Overall image area contraction: a <1 and b <1	the image area decreases both in the horizontal and vertical directions

A variety of digital images constitute a single frame $f[m,n]$ of the movie. We have chosen a number of objects ranging from simple element to complex image, some of which are the following:

A Single Character: Any ASCII character as defined in the computer keyboard can be the example of a character object. English letters (A, B, C, etc.), numerals (1, 2, 3, etc.), and other characters (#, *, &, etc.) belong to this type. For the single character, increasing or decreasing the character font size resizes the object.

A Text Object: A text object is defined as a set of the ASCII characters just mentioned. Resizing a text object is essentially the resizing of each character in the set. The text orientation and placement brings about different visualization. For instance a text placed left justified and the one placed center justified render different visual perceptions. The rate of resizing in conjunction with the frame rate adds another context of visualization. One may increase or decrease the text in a uniform or non-uniform frame rate.

A 2D Parametric Object: A parametric object is defined with the help of a parameter. The object can be one, two, or three dimensional. For example a circle has the parametric equation $x = r\cos\theta$ and $y = r\sin\theta$. Different r values for different image frames help in resizing the image object. Usually θ is frozen in a particular image frame.

A 3D Parametric Object: A 3D parametric object is similar to the 2D counterpart for example $x = r\cos\theta$, $y = r\sin\theta$, and $z = r^2$. But the 3D object is again transformed to 2D perspective as far as an image frame is concerned.

A Gray Image Object: A gray image by discrete mathematics turns to two dimensional intensity function $f[m,n] \in 2^p$ where p is any positive integer with $f[m,n]$ stretches up to $M \times N$ pixels where 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 [1]. In most digital image formats, the p is typically 8, 9, or higher.

A Color Image Object: 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 size (or $M \times N$) and bears aforementioned $f[m,n]$ meanings. Any digital movie frame is mathematically just the triplet $\begin{Bmatrix} r[m,n] \\ g[m,n] \\ b[m,n] \end{Bmatrix}$.

SIMPLIFICATION OF RESIZING

Earlier quoted affine transform is a general concept. We may further simplify the concept with prototype example which is addressed here. First there is difference between gray level and image scalings. The former and latter refer to $f[m,n]$ value and $[m,n]$ coordinate respectively. Image scaling (more appropriately resizing) basically contracts or stretches an image area which perhaps you exercise in any window related activities on any computer screen. Also image scaling is important when image taken on different area is displayed on a common picture frame.

The scaling relationship introducing a T vector is simplified by $\begin{bmatrix} m' \\ n' \end{bmatrix} = T \begin{bmatrix} m \\ n \end{bmatrix}$ where $T = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$.

Whether gray/color level mapping or area resizing, interpolation (especially two dimensional) is intrinsically connected in the image processing. Available interpolation schemes are nearest neighborhood, bilinear, bi-cubic, and spline [4]-[6]. Bilinear is chosen for widespread application.

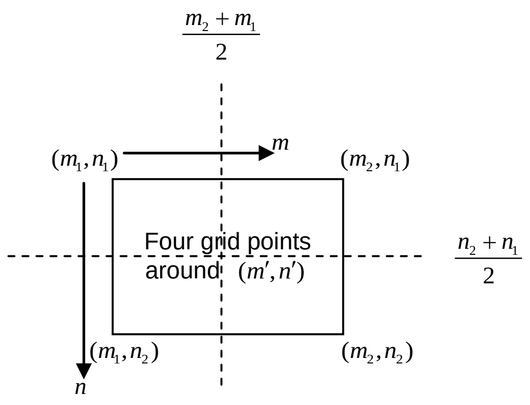

Figure 1: Squarely formed Four Grid Points in 2D FD Domain.

Every interpolation scheme determines $f[m',n']$ at $[m',n']$ from surrounding points. Bilinear interpolation is derived from one dimensional counterpart exercising first along m then along n.

Concerning Figure 1, squarely formed four grid points have the coordinates and functional values (m_1,n_1), (m_2,n_1), (m_2,n_2), and (m_1,n_2) and $f[m_1,n_1]$, $f[m_2,n_1]$, $f[m_2,n_2]$, and $f[m_1,n_2]$ respectively obviously in two dimensional finite difference (2D FD) domain. We determine $f[m',n']$ at fractional (m',n') from these four grid points despite $f[m,n]$ and $[m,n]$ being integer. Relative location of (m',n') compared to the midlines $\frac{m_1+m_2}{2}$ and $\frac{n_1+n_2}{2}$ trends the value of $f[m',n']$ towards the four neighbors.

Following expressions are exercised to calculate the bilinearly interpolated functional value:

linearity along $n = n_1$:

$$f[m',n_1] = \frac{f[m_2,n_1] - f[m_1,n_1]}{m_2 - m_1}(m' - m_1) + f[m_1,n_1]$$

linearity along $n = n_2$:

$$f[m',n_2] = \frac{f[m_2,n_2] - f[m_1,n_2]}{m_2 - m_1}(m' - m_1) + f[m_1,n_2]$$

linearity along $m = m'$:

$$f[m',n'] = \frac{f[m',n_2] - f[m',n_1]}{n_2 - n_1}(n' - n_1) + f[m',n_1]$$

Understandably the last mathematical relationships are valid over $m_1 \le m' \le m_2$ and $n_1 \le n' \le n_2$.

MODULAR EXAMPLE

To exemplify let us consider the prototype digital image,

$$f[m,n] = \begin{bmatrix} 8 & 5 & 5 & 4 \\ 9 & 4 & 0 & 1 \\ 7 & 21 & 14 & 21 \end{bmatrix}$$

which is to be resized on $\begin{Bmatrix} m' = 3.4m \\ n' = 2.3n \end{Bmatrix}$.

From $f[m,n]$ the pixel intervals are $0 \le m \le 3$ and $0 \le n \le 2$ and write the new intervals as $0 \le m' \le 10.2$ and $0 \le n' \le 4.6$ noticing the T vector $\begin{bmatrix} 3.4 & 0 \\ 0 & 2.3 \end{bmatrix}$.

Pixels are integer so rounding towards the nearest integer we get $0 \le m' \le 10$ and $0 \le n' \le 5$, not to mention,

$$[m,n] = \begin{bmatrix} (0,0) & (1,0) & (2,0) & (3,0) \\ (0,1) & (1,1) & (2,1) & (3,1) \\ (0,2) & (1,2) & (2,2) & (3,2) \end{bmatrix}$$

and

$$[m',n'] = \begin{bmatrix} (0,0) & (1,0) & (2,0) & \cdots & (10,0) \\ (0,1) & (1,1) & (2,1) & \ddots & (10,1) \\ & \vdots & & & \\ (0,5) & (1,5) & (2,5) & & (10,5) \end{bmatrix}$$

(matrix size 6×11). Inverse (m',n') is,

$$\begin{bmatrix} (0,0) & (0.29,0) & \cdots & (2.94,0) \\ (0,0.43) & (0.29,0.43) & & (2.94,0.43) \\ \vdots & & \ddots & \\ (0,2.17) & (0.29,2.17) & & (2.94,2.17) \end{bmatrix}$$

(matrix size 6×11) that is obtained by dint of $T^{-1}\begin{bmatrix} m' \\ n' \end{bmatrix}$.

Bilinear interpolation results,

$$f(m',n') = \begin{bmatrix} 8 & 7.12 & \cdots & 4.06 \\ 8.43 & 7.3 & & 2.7 \\ & \vdots & \ddots & \\ 7.52 & 10.18 & & 15.46 \\ ? & ? & & ? \end{bmatrix}$$

(matrix size 6×11) nearest integer of which provides,

$$f[m',n'] = \begin{bmatrix} 8 & 7 & 6 & & 4 \\ 8 & 7 & 6 & & 3 \\ 9 & 7 & 6 & \cdots & 1 \\ 8 & 9 & 9 & & 7 \\ 8 & 10 & 13 & & 15 \\ ? & ? & ? & & ? \end{bmatrix}$$ (matrix size 6×11).

Like most algorithms the scaling implementation also comes up with unwanted outcome which is indicated in the matrix by question mark. The unwanted question mark does not come as a surprise. The given interval of n is $0 \le n \le 2$ unexpectedly the inverse computing results $0 \le n \le 2.17$ which is beyond the image area so machine cannot decide about the last points. Usually the unwanted region is padded by 0s but that may lead to unwanted black region in the image.

The modular example so illustrated concentrates on bidirectional stretching i.e. a>1 and b>1. What if a>1 and b=1 e.g. a=3.4? The $f[m',n']$ would have had the same number of rows as $f[m,n]$ with increased column number. Again a=1 and b>1 would provide $f[m',n']$ with the same number of columns as $f[m,n]$ with increased row number. The last type of table A actually shrinks the $f[m,n]$ in both directions.

It goes without saying that the computing technique exemplified above applies to every frame of a digital movie.

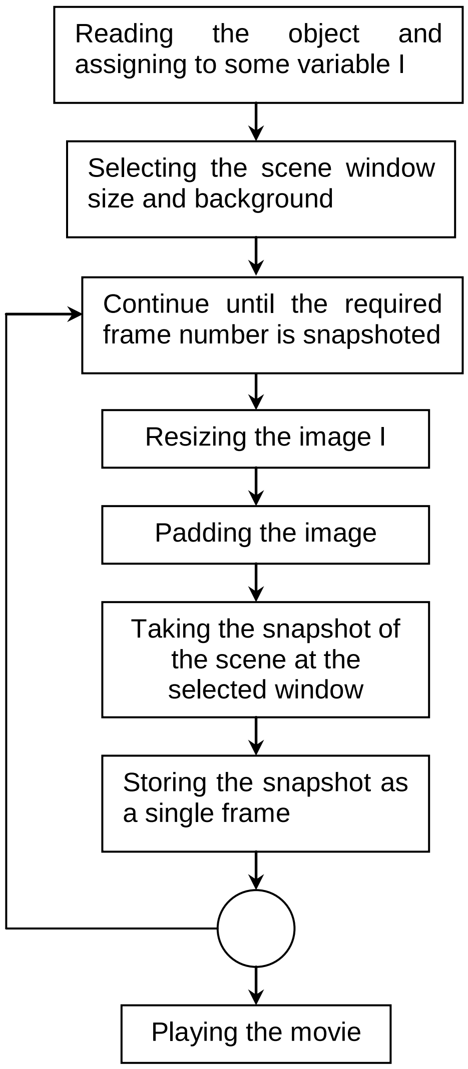
Reading the object and assigning to some variable I
Selecting the scene window size and background
Continue until the required frame number is snapshoted
Resizing the image I
Padding the image
Taking the snapshot of the scene at the selected window
Storing the snapshot as a single frame
Playing the movie

Figure 2: Basic Algorithm for Digital Animation using Image Resizing.

ALGORITHM ON IMAGE RESIZING FOR DIGITAL IMAGE ANIMATION

Digital animation framework illustrated in [1]-[2], [7]-[8] is applied here on image resizing. Figure 2 presents basic flowchart of the whole animation process. In order to implement the animation, a nine step algorithm is applied as follows:

Step 1: reading the image object by an image reader and retaining the image to some buffer variable I, Step 2: selecting the image window size along with the background,

Step 3: a looping operation is conducted for the movie frame storage,

Step 4: resizing the image inside the loop,

Step 5: padding the resized image, this step is mandatory to be strict with display size because discrete computing alters the image size,

Step 6: taking snapshot of the image followed by the padding if it is necessary,

Step 7: storing the snapshot as a single movie frame,

Step 8: continuing step 3 loop operation as many times as required by the user, and

Step 9: finally displaying all accumulated frames as a single movie.

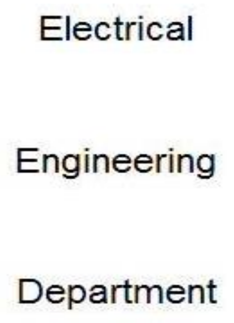

Figure 3: A Sentence Object.

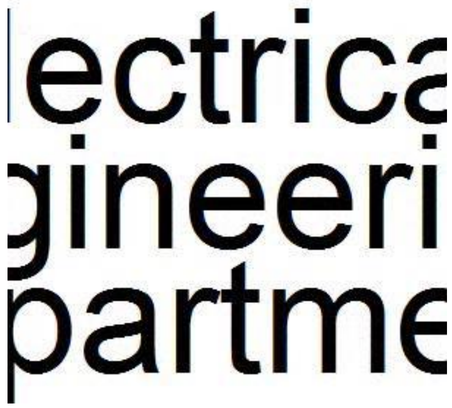

Figure 4: Object of Figure 3 with Font Size 25.

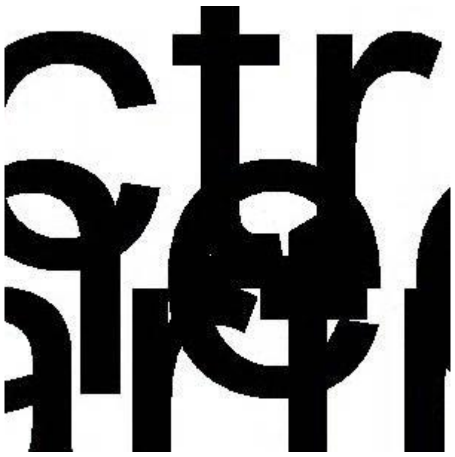

Figure 5: Object of Figure 3 with Font Size 35.

Figure 6: An Image of Two Children.

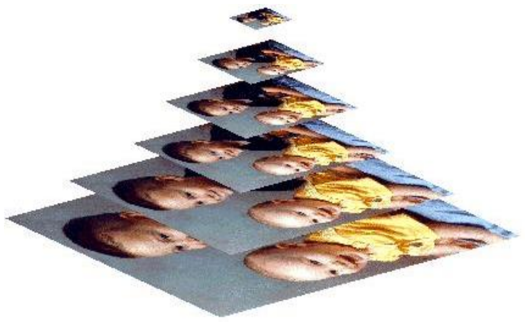

Figure 7: Scale Down Counterparts of an Image in Pyramidal Form.

DIGITAL ANIMATION RESULTS

The algorithm just explained with regards to a single frame will be applicable to all frames of a movie. The problems of presenting a digital movie on a paper are myriad. Necessary technical tools are DVD player, media player, real time player, etc. For this reason only intermediate frames or stacked images are presented to render the sense of digital simulation.

A variety of objects are tested to implement the algorithm, two examples of which are the following: (a) a sentence object which is “Electrical Engineering Department” (Figure 3) and (b) an image of two children (Figure 6). These two base objects along with their animated versions are detailed in the following.

Animation 1

The sentence object is stored as a text image of font size 16 in white background, the pixel size of which is 300×300 and depicted in Figure 3. Although black and white object, the RGB triplet is chosen for simulation reason or general image consideration. Each component of the RGB triplet i.e. $r[m,n]$, $g[m,n]$, or $b[m,n]$ contains in [0,255] or 8-bit. Figures 4 and 5 show two video frames with font sizes 25 and 35, respectively. Increased font size image of the last two frames fits out of the pixel area 300×300 which is cropped for fixed size video frame. With frame rate 15 frames/sec and at sample rate 24 bit, the number of movie frames generated was 20 whence digital video duration is 1.33 sec. One would perceive the digital video as the text “Electrical Engineering Department” becoming inflated gradually and appears to be coming towards the user. The animated digital video was saved in audio visual interleave (AVI) format.

Animation 2

Suitability of the algorithm is seen on the two kids image as portrayed in figure 6 whose pixel size is 600×800. Figure 7 depicts the stacked form of the two kids image. The stacked form itself can be an animation. The reduction factor was 0.8, 0.6, 0.4, 0.3 and 0.25, respectively, that is a=0.8 and b=0.8 for the first reduction factor as regards to table A. New pixel domain encompasses $0 \le m' \le 479$ and $0 \le n' \le 639$ while the original image spans over $0 \le m \le 599$ and $0 \le n \le 799$. Imagine the shrinkage with a=0.8 and b=0.8, the unwanted zone appears for $m' > 479$ and/or $n' > 639$, filling of which on the RGB triplet i.e. $r[m,n]$, $g[m,n]$, or $b[m,n]$ is user-decided. The color level assignment and resolution of animation 1 is exercised here too. Applying scaling is not the only factor which plays role in visual perception. Other factors in conjunction with scaling render different animation perception, some of which are addressed below:

a) justification i.e. placement of the scaled image on left, right, or center,

b) relative position of the scaled

c) image origin with reference to that of the background,

d) other animation criterion for example rotation,

e) three dimensional context of the scaled image placement, and

f) adding a skew to the scaled image.

Each of the above requires algorithm anew. Hopefully the algorithms will be addressed in future. Concerning the Figure 7, the animation needs placing the image on a planar base say x-y then every scale down counterpart has to be placed along z axis for example at z=200, 400, 600, etc. relative to pixel scale. Not to mention, the selection of view angle is kept constant. Certainly the scaled down images shape as a pyramid. Every reduction factor provides one movie frame. Although we quoted five or six, one second movie needs fifteen reduction factors. Regarding distribution, one may choose linear or nonlinear. The distribution also plays a role in the animation.

What about the Figure 8 depicted translational form? The five reduction factors in animation 1 are chosen here too. Since all scaled down images are occupying a plane (say x-y), one of the three coordinates is frozen i.e. z=0 perceptibly for top view. The placement of consecutive frames is user-decided too for example second one at (300,200) in terms of pixel coordinate. Every animation poses a new constraint. Since background image area is increased, the evolved unwanted zone must be padded to have deterministic gray or color level on the whole background. The animation was saved in AVI format too. Each of the two animations renders distinctive visual perception. As far as the title is concerned, we mainly focused on scaling.

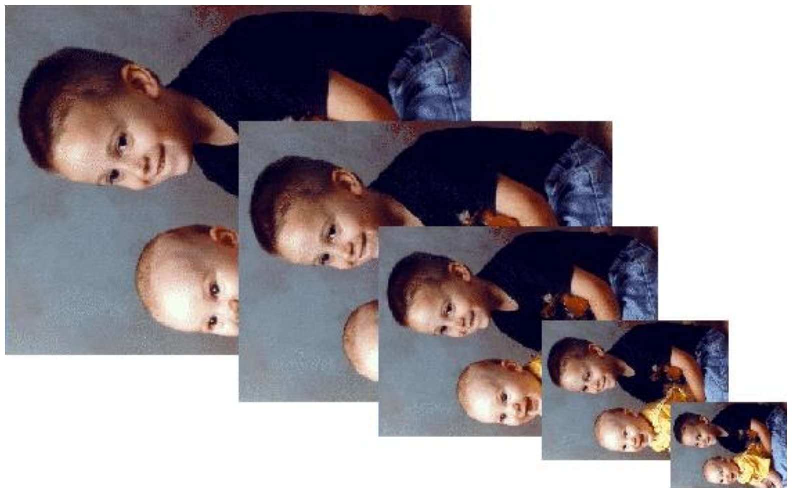

Figure 8: Scale Down Counterparts of an Image in Translational Form.

CONCLUSION

Image and/or digital video algorithm is slightly different from other algorithms. Eventual effectiveness of a digital imagery is utterly user's or viewer's acceptability. Subjective criterion for instance mean square error has little significance. Primary focus is concerned on visual quality. In other words qualitative measures are sought rather than quantitative. The algorithm is proved to be effective for text objects and image frames.

Qualitative attribution to visual context is also checked but not presented. Since the algorithm is partially non-iterative, computational efficiency is superb given the recent personal computer (PC) processor speed. Any PC based animation can easily adopts the algorithm without the need for sequential or parallel counterpart thereby reducing cost or extra acquisition devices in movie making. As platform we have chosen MATLAB®. Figuratively scaling is a fantastic tool to deliver eye-catching digital movie.

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

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

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

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

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

6. Nuruzzaman, M. 2012. *Digital Image: Theories, Algorithms, and Applications*, CreateSpace: Washington, D.C.

7. 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.

8. 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. 2017. “Object Resizing: A Subtle Tool in Digital Animation”. *Pacific Journal of Science and Technology*. 18(2):120-127.

Pacific Journal of Science and Technology