ST. XAVIER’S COLLEGE

**Maitighar, Kathmandu**

**(Affiliated to Tribhuvan University)**



**Computer Graphics**

**Lab Assignment #7**

To implement transformations (translate, scale and rotate)

**Submitted By**

Alok Shrestha

B.Sc. CSIT

Year II/IV Semester

013BSCIT005

**Submitted To**

Er. Anil Sah

Lecturer

Department of Computer Science

St. Xavier’s College

Maitighar, Kathmandu

**Submitted On**

September, 2015

**STATEMENT**

**TO IMPLEMENT TRANSFORMATIONS (TRANSLATE, SCALE AND ROTATE)**

**ALGORITHM**

**For Translation**

1. Get the width and height of the source image
2. Get translation vector (Tx, Ty)
3. For each point i in width

For each point j in height

The translated point (x’, y’) is given by

x' = i + Tx

y’ = j + Ty

Plot the points (x’, y’) with the same color as source in destination

1. Stop

**For Scaling**

1. Get the width and height of the source image
2. Get scaling parameter Sx, Sy
3. For each point i in width

For each point j in height

The translated point (x’, y’) is given by

x' = i \* Sx

y’ = j \* Sy

Plot the points (x’, y’) with the same color as source in destination

1. Stop

**For Rotation**

1. Get the width and height of the source image
2. Get rotation degree
3. Convert degree to radian

radian = degree / 180 \* 3.141592

1. For each point i in width

For each point j in height

The translated point (x’, y’) is given by

x' = (i \* cos(radian) – j \* sin(rad)) + 250;

y’ = (j \* cos(radian) + i \* sin(rad)) + 200;

where (200, 250) have been added to align original position of source image to around 2/3rd of the output image area

Plot the points (x’, y’) with the same color as source in destination

1. Stop

**SOURCE CODE**

//---------------------------------------------------------------------------

#include <vcl\vcl.h>

#include <math.h>

#pragma hdrstop

#include "Unit1.h"

//---------------------------------------------------------------------------

#pragma resource "\*.dfm"

TForm1 \*Form1;

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{

int x=Image1->Width;

int y=Image1->Height;

int Tx=StrToInt(Edit1->Text);

int Ty=StrToInt(Edit2->Text);

for(int i=0; i<=x; i++)

{

for(int j=0; j<=y; j++)

{

int a=i+Tx;

int b=j+Ty;

Image2->Canvas->Pixels[a][b]=Image1->Canvas->Pixels[i][j];

}

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button2Click(TObject \*Sender)

{

int x=Image1->Width;

int y=Image1->Height;

int Sx=StrToInt(Edit3->Text);

int Sy=StrToInt(Edit4->Text);

for(int i=0; i<=x; i++)

{

for(int j=0; j<=y; j++)

{

int a=i\*Sx;

int b=j\*Sy;

Image2->Canvas->Pixels[a][b]=Image1->Canvas->Pixels[i][j];

}

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button3Click(TObject \*Sender)

{

int x=Image1->Width;

int y=Image1->Height;

int deg=StrToInt(Edit5->Text);

float rad=deg/180.0\*3.141592;

for(int i=0; i<=x; i++)

{

for(int j=0; j<=y; j++)

{

int a=(i\*cos(rad)-j\*sin(rad))+250;

int b=(j\*cos(rad)+i\*sin(rad))+200;

Image2->Canvas->Pixels[a][b]=Image1->Canvas->Pixels[i][j];

}

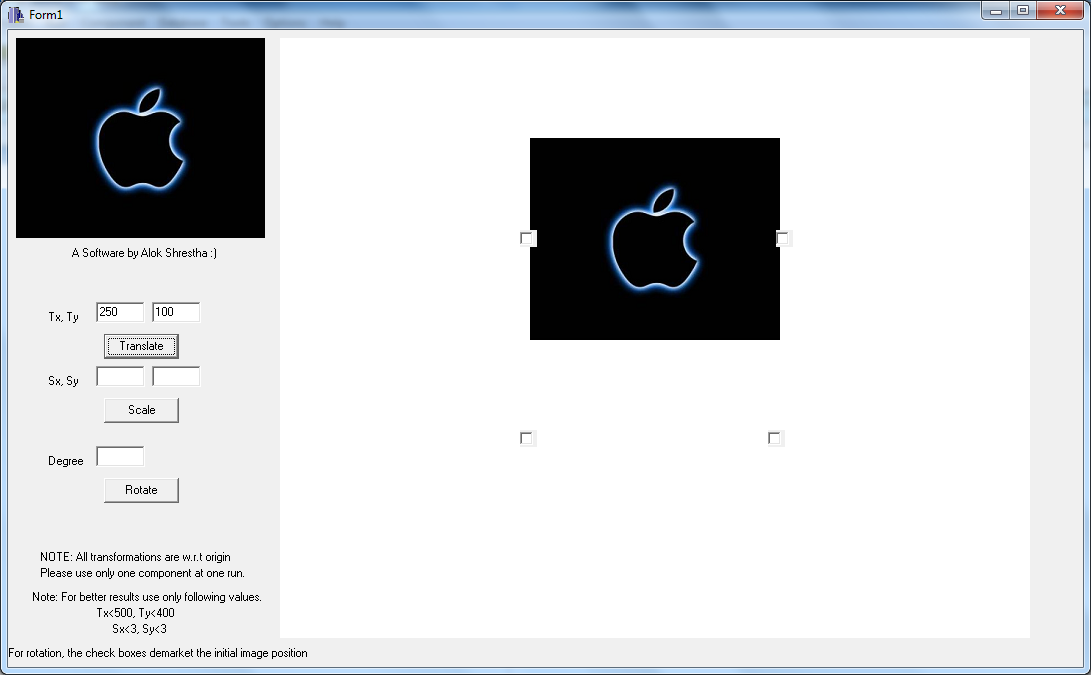
}

}

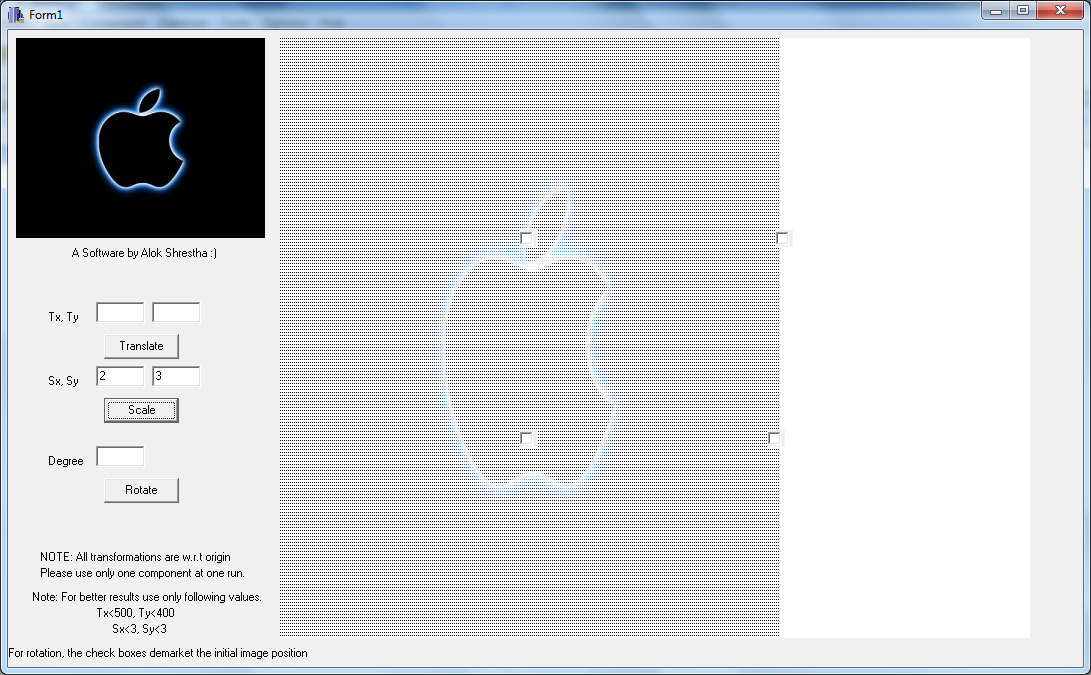
//---------------------------------------------------------------------------

**OUTPUT/s**

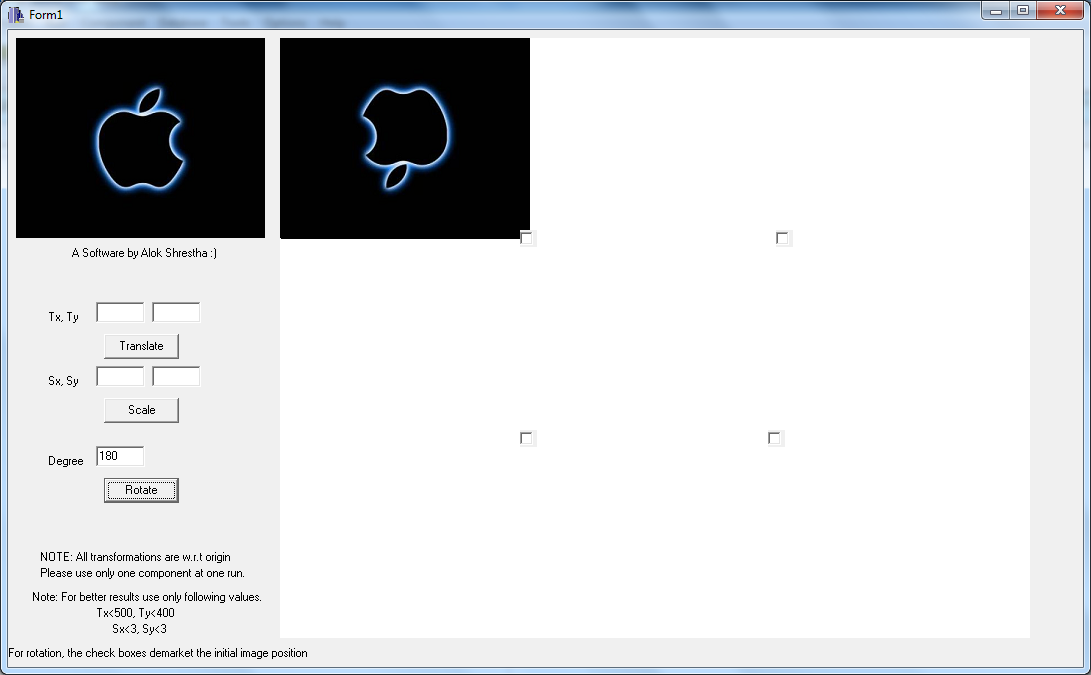
**First Run: Translation by (250,100)**

****

**Second Run: Scaling by Sx=2, Sy=3**



**Third Run: Rotate by 180 degrees**



**CONCLUSION**

Hence, the given transformations were performed in C++builder.