Multimedia Technologies - Open GL

1- Setting Up (Initializing The Screen)

The header "windows.h" is needed for the Windows platform only

- GLUT header, which is guaranteed to include "glu.h" (for GL Utility) and "gl.h" (for Core OpenGL).

Core OpenGL (GL): consists of hundreds of commands, which begin with a prefix "gl" (e.g., glColor, glVertex, glTranslate, glRotate). The Core OpenGL models an object via a set of geometric primitives such as point, line and polygon.

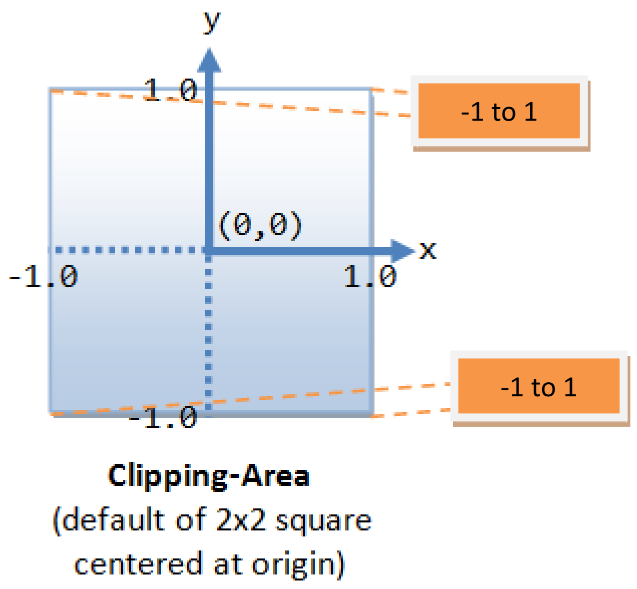
OpenGL Utility Library (GLU): built on-top of the core OpenGL to provide important utilities (such as setting camera view and projection) and more building models (such as qradric surfaces and polygon tessellation). GLU commands start with a prefix "glu" (e.g., gluLookAt, gluPerspective).

```
#include <windows.h>
#include <GL/glut.h>

void display()
 {
   glClearColor(0.0f, 0.0f, 0.3f, 0.0f);
   glClear(GL_COLOR_BUFFER_BIT);

///----- Objects code will be written here

    glFlush();
}

void main(int argc, char** argv)
{
   glutInit(&argc, argv);
   glutInitWindowSize(800,600);
   glutCreateWindow("My OpenGL Window");
   glutDisplayFunc(display);
   glutMainLoop();
}
```

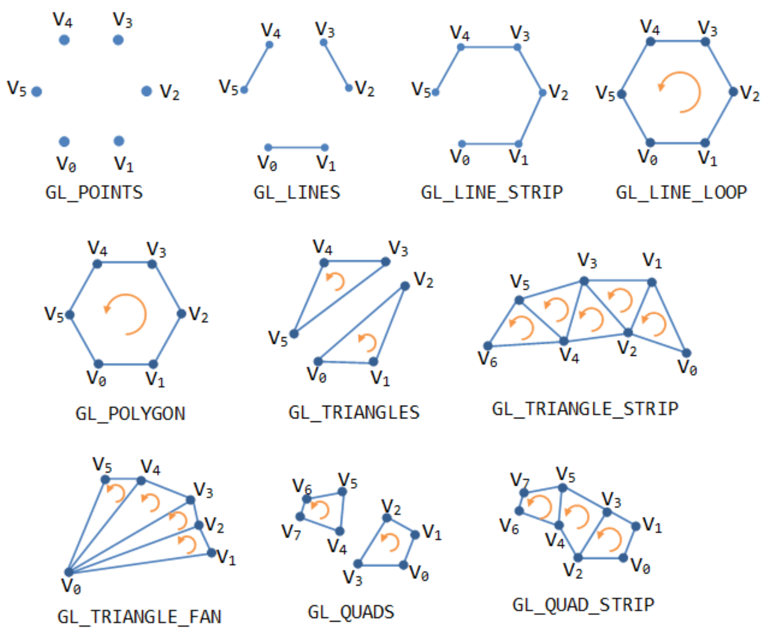
- glClearColor(0.0f, 0.0f, 0.3f, 1.0f) : To set the background color (Red, Green , Blue , and Opacity) in the example, the background color is set to blue and opaque. The range of the color is between 0-1.
- glClear(GL_COLOR_BUFFER_BIT : Clear the color buffer and set the one used in previous command
- Objects will be created in this part. ////....
- glFlush(); Render the drawn objects.
- Void main : To call the functions that are responsible to draw the objects.
- glutInit: initializes GLUT, must be called before other GL/GLUT functions. It takes the same arguments as the main().
- glutInitWindowSize(800,600) : To set the screen dimensions (size)
- glutCreateWindow("My OpenGL Window") To create window with a title
- glutDisplayFunc(display): Display drawn objects
- glutMainLoop: enters the infinite event-processing loop, i.e., put the OpenGL graphics system to wait for events (such as re-paint), and trigger respective event handlers (such as display()).

Screen Coordinates :

The default OpenGL 2D clipping-area (i.e., what is captured by the camera) is an orthographic view with x and y in the range of -1.0 and 1.0, i.e., a 2x2 square with centered at the origin. As shown in figure:

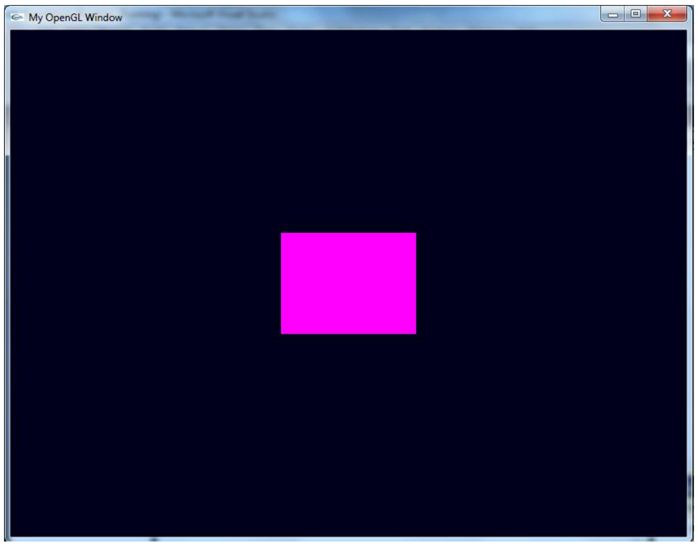


2-Different Geometric Primitives in Open GL:
As shown in figure, each primitive is consist of set of vertices, to draw a certain primitive, the name of that primitive should be written after GLBegin command , followed by the location of each pixel constructing that object.



OpenGL Primitives

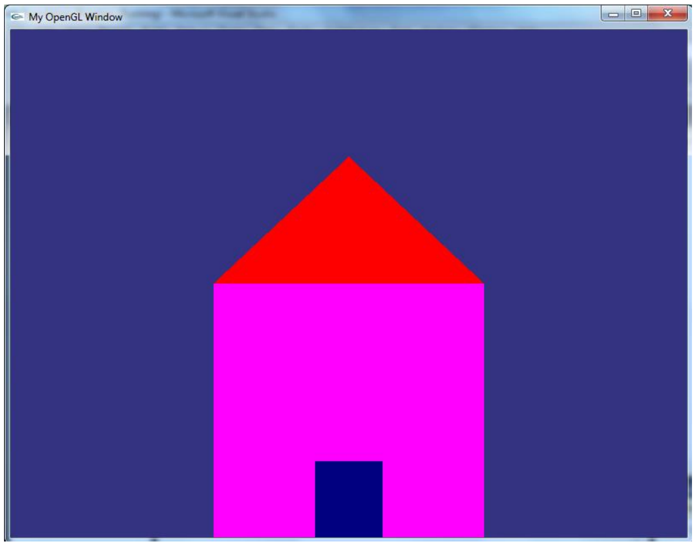
Example : An Open GL code to draw a square on the center of the screen with side length of 0.4.



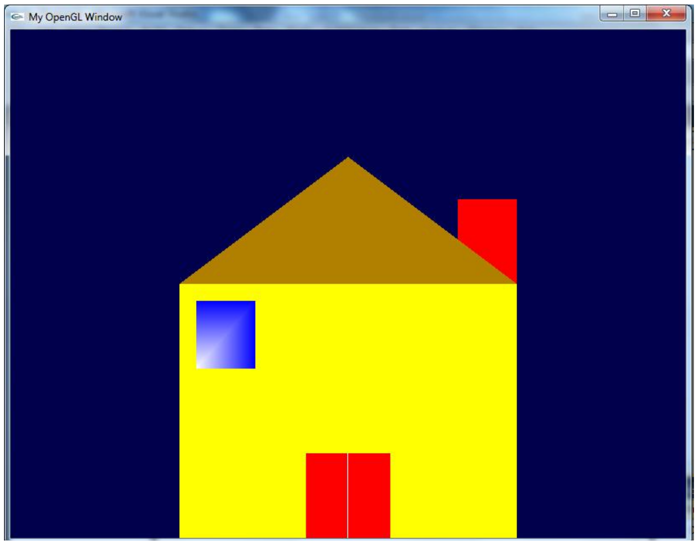
Vertices are drawn in anti clockwise.

```
// Square
#include <windows.h>
#include <GL/glut.h>


void display()
 {
   glClearColor(0.0f, 0.0f, 0.1f, 0.0f);
   glClear(GL_COLOR_BUFFER_BIT);

glBegin(GL_QUADS);
   glColor3f(1.0f, 0.0f, 1.0f);

   glVertex2f(-0.2f,-0.2f);
   glVertex2f(0.2f,-0.2f);
   glVertex2f(0.2f,0.2f);
   glVertex2f(-0.2f,0.2f);


glEnd();

    glFlush();
}

void main(int argc, char** argv)
{
   glutInit(&argc, argv);
   glutInitWindowSize(800,600);
   glutCreateWindow("My OpenGL Window");
   glutDisplayFunc(display);
   glutMainLoop();
}
```

Ex : Write Open GL code to draw the following Object.



Home work:



```
//House
glBegin(GL_QUADS);
   glColor3f(1.0f, 0.0f, 1.0f);
   glVertex2f(-0.4f,-1.0f);
   glVertex2f(0.4f,-1.0f);
   glVertex2f(0.4f,0.0f);
   glVertex2f(-0.4f,0.0f);
glEnd();


glBegin(GL_QUADS);
   glColor3f(0.0f, 0.0f, 0.5f);
   glVertex2f(-0.1f,-1.0f);
   glVertex2f(0.1f,-1.0f);
   glVertex2f(0.1f,-0.7f);
   glVertex2f(-0.1f,-0.7f);
glEnd();


glBegin(GL_TRIANGLES);
 glColor3f(1.0f, 0.0f,0.0f);
  glVertex2f (0.4f,0.0f);
  glVertex2f (0,0.5f);
  glVertex2f (-0.4f,0);
glEnd();
   glFlush();
}
```

3-Translation in Open GL:

1-glTranslatef(X,Y,Z);
To translate(move) the object according to a certain value (0-1) towards X axis, Y axis and Z axis (which is towards the viewer)

2- glRotatef(Degree, X, Y, Z);
To rotate an object by angle specified in degrees about the axis which its value is 1.

Example beside shows the translation of a triangle towards Y axis and it is rotated by 180^o about X axis.

```
//Translate and Rotate
void display()
 {

   glClearColor(1.0f, 1.0f, 0.5f, 1.0f);
   glClear(GL_COLOR_BUFFER_BIT);

glTranslatef(0.0, 0.5f, 0.0f);
glRotatef(180,1.0, 0.0f, 0.0f);

glBegin(GL_TRIANGLES);
glColor3f(1.0f,0.0f,0.0f);
glVertex2f(-0.5f,0);
glVertex2f(0.5f,0);
glVertex2f(0.0f,0.5);
glEnd();
glFlush();

}
```

4-Animation:

Animation is the process of changing the location or the shape of a certain object during a period of time. To perform animation in open GL, idle function is needed In the idle() function, you could issue glutPostRedisplay command to post a window re-paint request, which in turn will activate display() function. We also use glPushMatrix to save the current state, perform transformations, and restore the saved state via glPopMatrix.

In following example the variable "mov" will be accumulated and used to specify the new position of the object.

H.W : Use animation to continously rotate the triangle drawn previous example around Z.

```
//Position Animation
#include <windows.h>
#include <GL/glut.h>
GLfloat mov = -1.0f;

void idle() {
   glutPostRedisplay();
}
void display()
 {
   glPushMatrix();
   glClearColor(1.0f, 1.0f, 0.5f, 1.0f);
   glClear(GL_COLOR_BUFFER_BIT);
   glTranslatef(mov, 0.0f, 0.0f);

    glBegin(GL_TRIANGLES);
       glColor3f(1.0f,0.0f,0.0f);
       glVertex2f(0,-0.5f);
       glVertex2f(0.5f,0);
       glVertex2f(0.0f,0.5);
    glEnd();
glPopMatrix();
glFlush();

mov = mov + 0.001;
if (mov>1)
       mov=-1;

}

void main(int argc, char** argv)
{
   glutInit(&argc, argv);
   glutInitWindowSize(800,600);
   glutCreateWindow("My OpenGL Window");
   glutDisplayFunc(display);
   glutIdleFunc(idle);
   glutMainLoop();
}
```

5-Keyboard Interaction

Two functions are used to interact with keyboard input one for special keys (like arrows) and other for traditional keys. Both are listed in the example. Here the object will be rotated clock wise while pressing the key 'a' from keyboard, while it will be rotated anti clock wise when pressing the left key.

H.W : Write OpenGL code to move a certain object to four directions according to following keyboard keys:
W : To move forward.
S : To move backward.
A : Moving to right.
D : Moving to left.

Also add the clockwise rotation by the key R.

References:

https://www3.ntu.edu.sg/home/ehchua/programming/opengl/CG_Introduction.html#zz-5.

Shreiner, D., Sellers, G., Kessenich, J., & Licea-Kane, B. (2013). OpenGL programming guide: The Official guide to learning OpenGL, version 4.3. Addison-Wesley.

```
//keyboard interaction
#include <windows.h>
#include <GL/glut.h>

GLfloat rot = -1.0f;

void idle() {
   glutPostRedisplay();
}
void display()
 {
   glPushMatrix();
   glClearColor(1.0f, 1.0f, 0.5f, 1.0f);
   glClear(GL_COLOR_BUFFER_BIT);
   glRotatef(rot, 0.0f, 0.0f,1.0f);

    glBegin(GL_TRIANGLES);
       glColor3f(1.0f,0.0f,0.0f);
       glVertex2f(0,-0.5f);
       glVertex2f(0.5f,0);
       glVertex2f(0.0f,0.5);
    glEnd();

glPopMatrix();
glFlush();
}

void specialKeys(int key, int x, int y) {
   switch (key) {
     case GLUT_KEY_LEFT:
rot = rot + 0.9;
if (rot>360)
       rot=-1;
         break;
   }
}
void keyboard(unsigned char key, int x, int
y) {
   switch (key) {
      case 'a':
         rot = rot - 0.9;
if (rot>360)
       rot=-1;
         break;
   }
}



void main(int argc, char** argv)
{
   glutInit(&argc, argv);
   glutInitWindowSize(800,600);
   glutCreateWindow("My OpenGL Window");
   glutDisplayFunc(display);
   glutIdleFunc(idle);
   glutSpecialFunc(specialKeys);
   glutKeyboardFunc(keyboard);
   glutMainLoop();
}
```

Multimedia Technologies

Adobe Photoshop

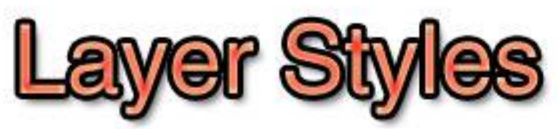
Adobe Photoshop is a software application for image editing and photo retouching for use on Windows or MacOS computers. Photoshop relies on layers. Photoshop layers are like sheets of stacked on over the other. transparent areas of a layer can be seen to the layers below.

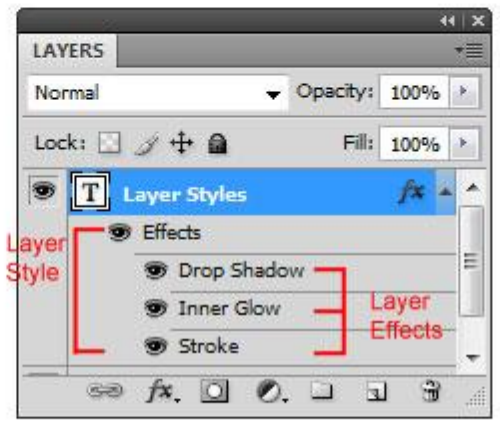
Layer Styles

What is a Layer Style?

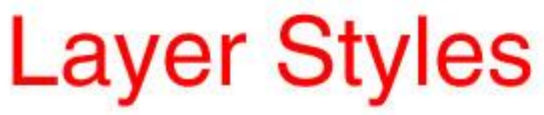
A layer style is simply one or more layer effects and blending options applied to a layer. Layer effects are things like drop shadows, stroke, and color overlays.

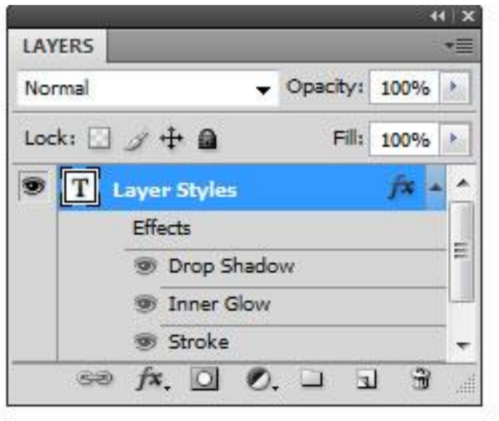
Here is an example of a layer with three layer effects (Drop Shadow, Inner Glow, and Stroke).



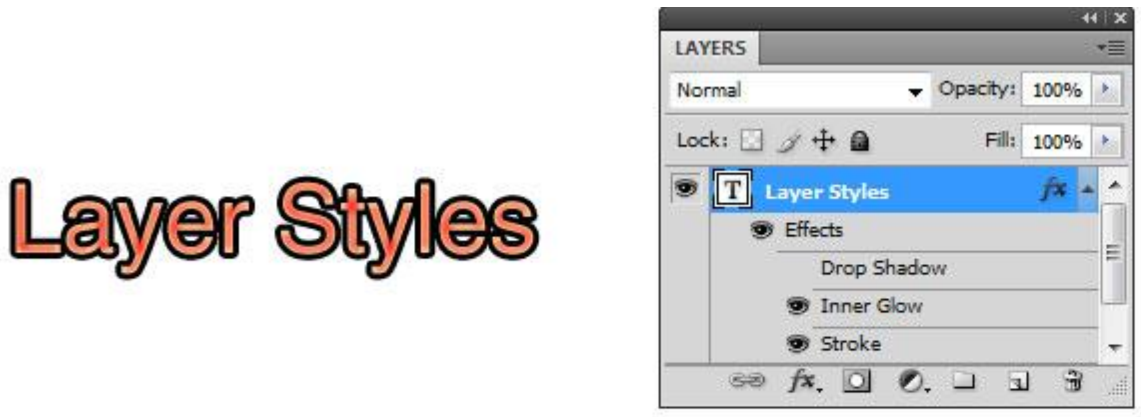


You can turn off a layer style by clicking on the eye icon beside Effects so that you can see how the original layer looks without its layer style.

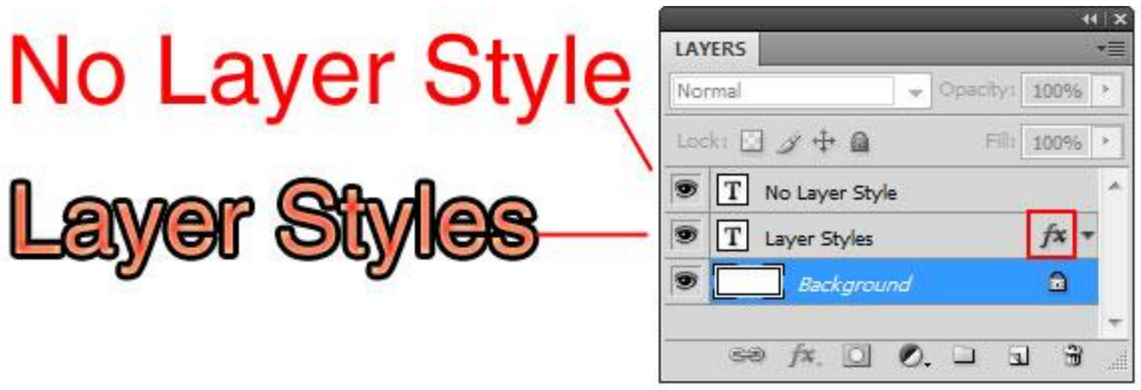




Similarly, you can turn off the visibility of each layer effect. In the following example, the Drop Shadow layer effect was disabled.



You can tell that a layer has a layer style because of the fx icon on its right side.

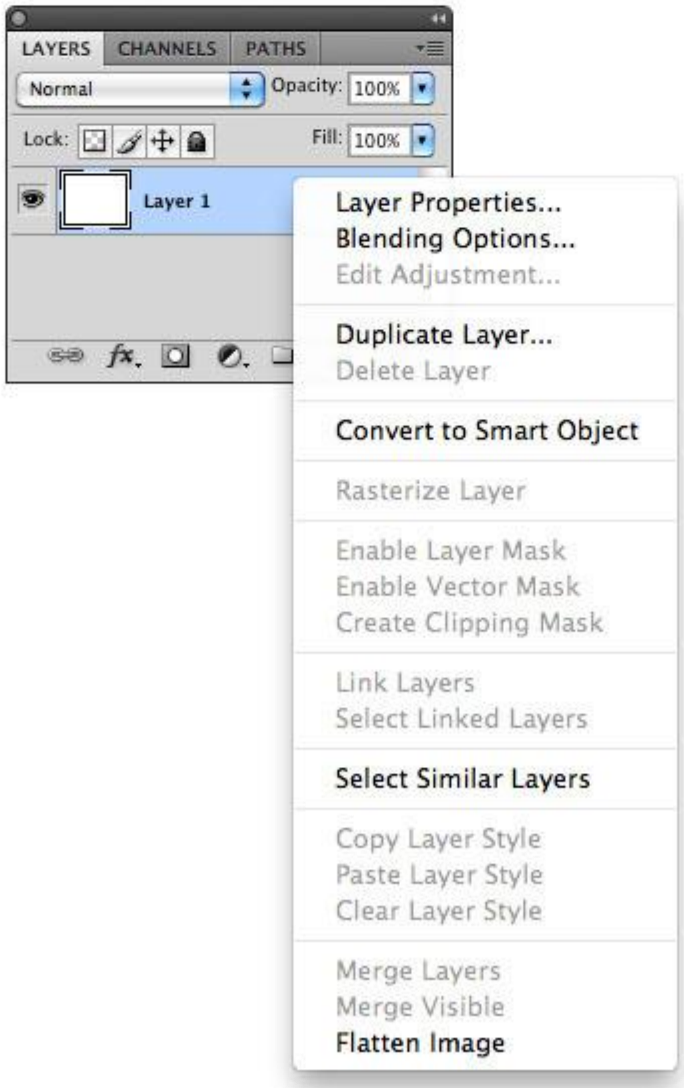


Applying Layer Styles

To set up a layer style, you will use the Layer Style dialog window. There are various ways to access the Layer Style dialog window.

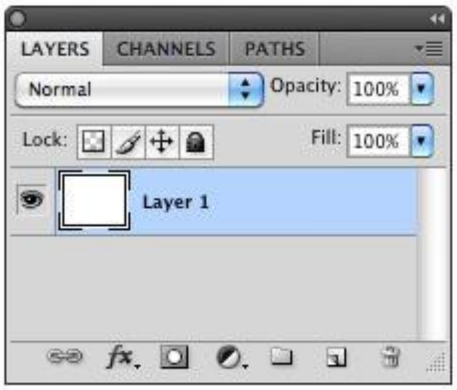
Right-Clicking on a Layer

By right-clicking on your layer, you can select Blending Options to open up the Layer Style dialog window.



Double-Clicking on a Layer

By double-clicking on your layer's thumbnail preview in the Layers Panel, you can open up the Layer Style window.



Application Bar Menu

Masks

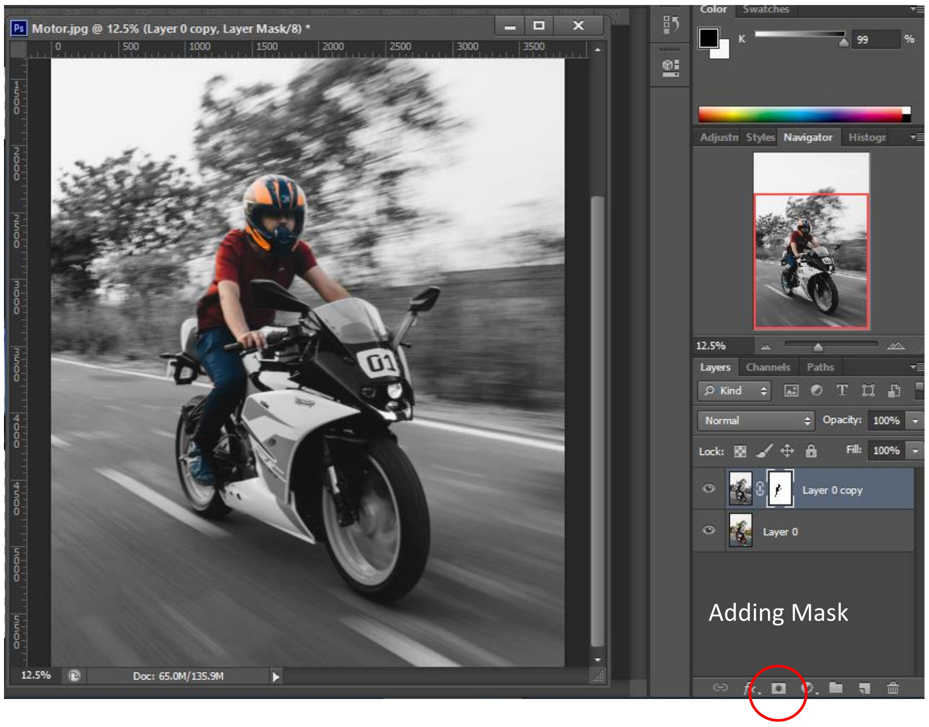
The term “mask” isn’t immediately understandable to someone outside the realm of graphic design. At its simplest definition a mask is a way to apply something to a very specific portion of an image.

Example

- Open the image Motor.jpg and unlock the layer.
- Duplicate the layer.
- By choosing the upper layer go to Image→ Adjustment → Hue / Saturation.
- Set the saturation to -100.

Now we create a mask over the upper layer (the grey one)

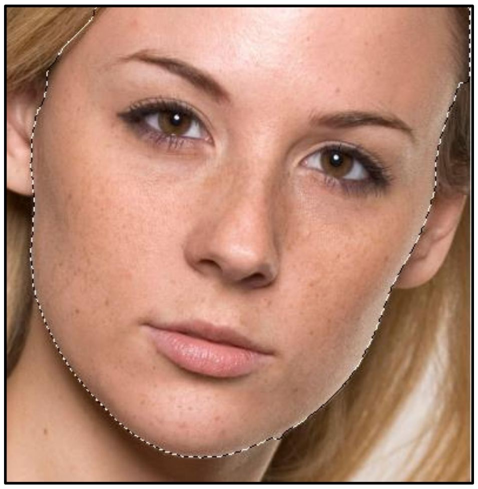
- Click on add layer mask at the bottom of the layer panel.
- Use the brush with black color and draw over the places where the driver exists, the black color will hide the parts and show the background, while the white color will keep the original mask.
- Save your work.



Mask by layer

Choose the upper layer, press ALT and the line between to layers, the top layer will be appeared according to the bottom one.

Image Retouching:



Open the Image

Choose the face areas with any suitable selection tool.

Go to Select → Refine edges.

Output it as New Layer with Layer Mask.

Click to view the original image with the new one.

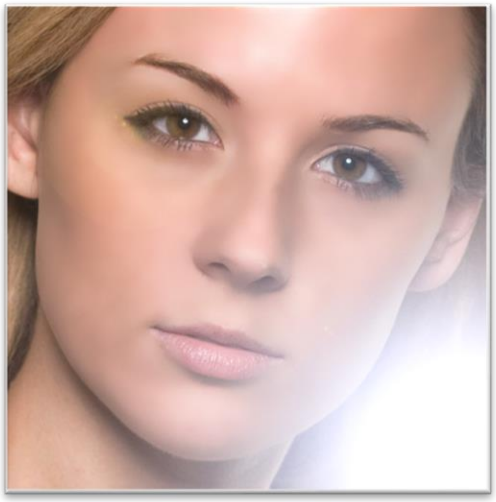
On the upper masked layred, choose filter→Blur→Surface Blur.

Choose the mask and cancel places where the blur is not needed, as eyes.

Add suitable noise by choosing Filter→Noise→add Noise.

Use the spot healing brush tool to remove any other blemishes.

Enhance the lighting/Coloring of the image or by adding filters as Filter→ Render→Lens Flare.

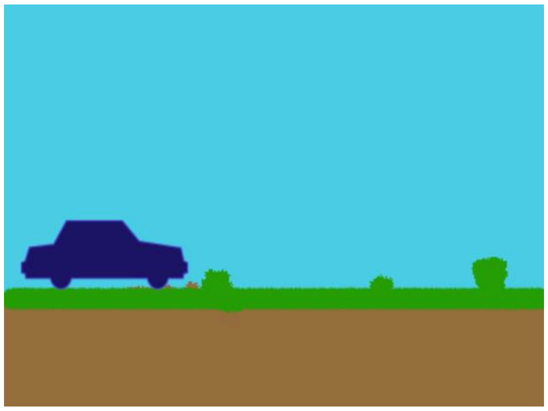


Time Line (Animated Images)

Create a 400 × 300 new image. Fill the background with the color blue.

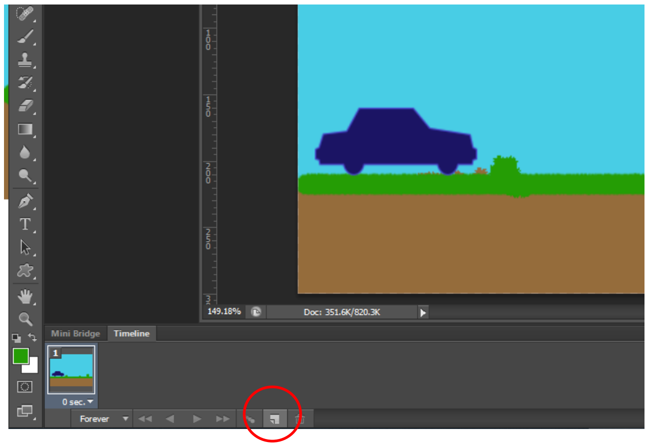
Fill the bottom quarter with brown color and draw a green stright line by pressing on shift key while dragging the brush from left to right.

Add a custom figure (a car) as new object (See figure)



Choose animation from the menu tab.

Press on duplicate the selected frame, and change the time from 0 sec to 0.1 sec.



While the new frame chosen, move the car to the right.

Repeat creating new frames and moving the car.

Play the frames by pressing the ▸ button.

Save the animated image by File → Save for web

Q\ Apply the animation using masks.

Adobe Premiere Pro

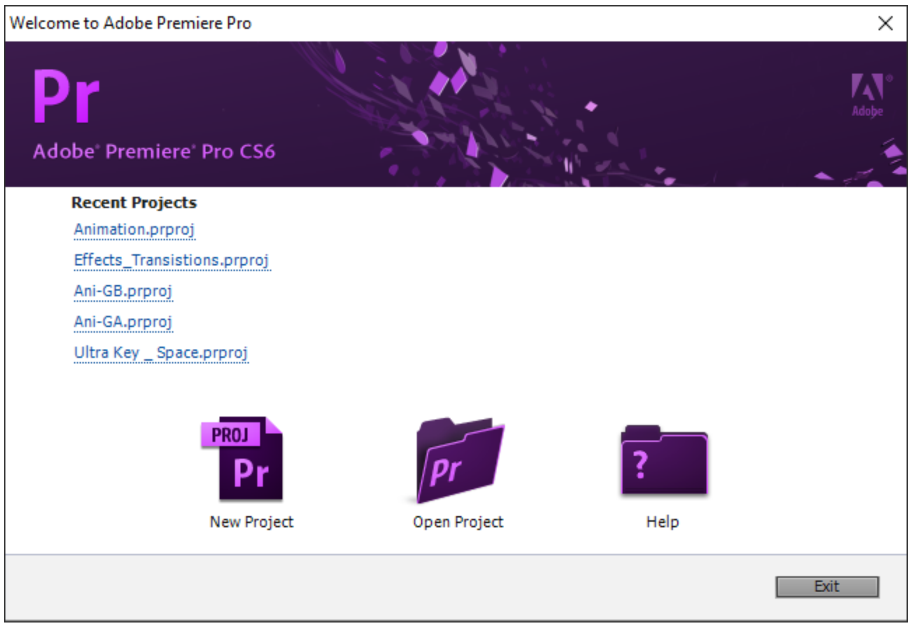
Adobe Premiere Pro: Adobe® Premiere® Pro CS6, the essential editing tool for video enthusiasts and professionals, enhances your creative power and freedom. Adobe Premiere Pro is the most scalable, efficient, and precise video-editing tool available. It supports a broad range of video formats, and it is compatible with other adobe applications.

Welcome Screen:

Project: For Creating a new project

Open Project: To open a previously saved scene.

Help: Showing application's help



Main Premiere Window:

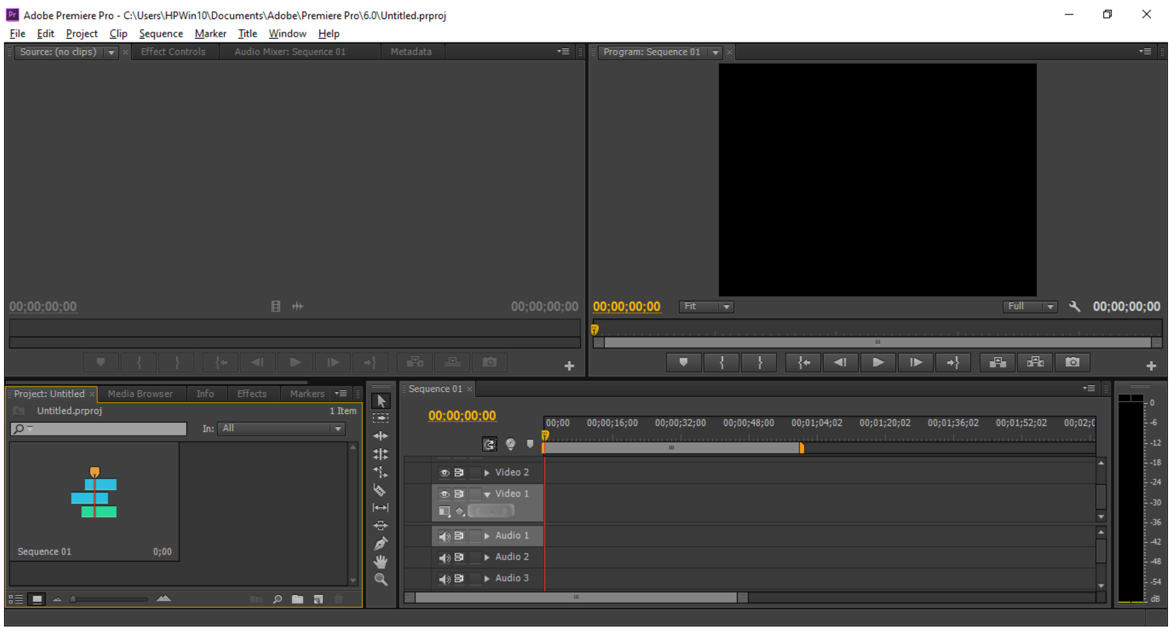
Project Panel: The place where all the media materials are gathered before adding it to the source monitor. To import media to the project panel press CTRL+I from keyboard and choose the required material.

Source Monitor: In source monitor the media can be viewed or cut before adding it to the time line. To add a file to the source monitor, choose it from the project panel and click and drag it to the source monitor.

Time line: It is the place where all the video, audio, and text files are gathered and arranged. It is also the place where the transitions and effects are applied. The sequence of files in the time line is the one will be shown in the end video that is displayed on program monitor.

Program monitor: The final result after the arrangement of scenes in the time line will be shown in the program monitor.

Tool box: It has the required tools for modifying the media, as clipping, changing time or selecting.



Adding Scenes to a project

After importing a certain multimedia file to the project panel and preview it in the source monitor, it can be added to the timeline by click and drag it. Note that by default there are three tracks for video and three for audio. When adding a video with a sound, one track will be occupied for each of video and audio tracks. Using multiple tracks will allow to preview more than one video or mixing two or more audio files.

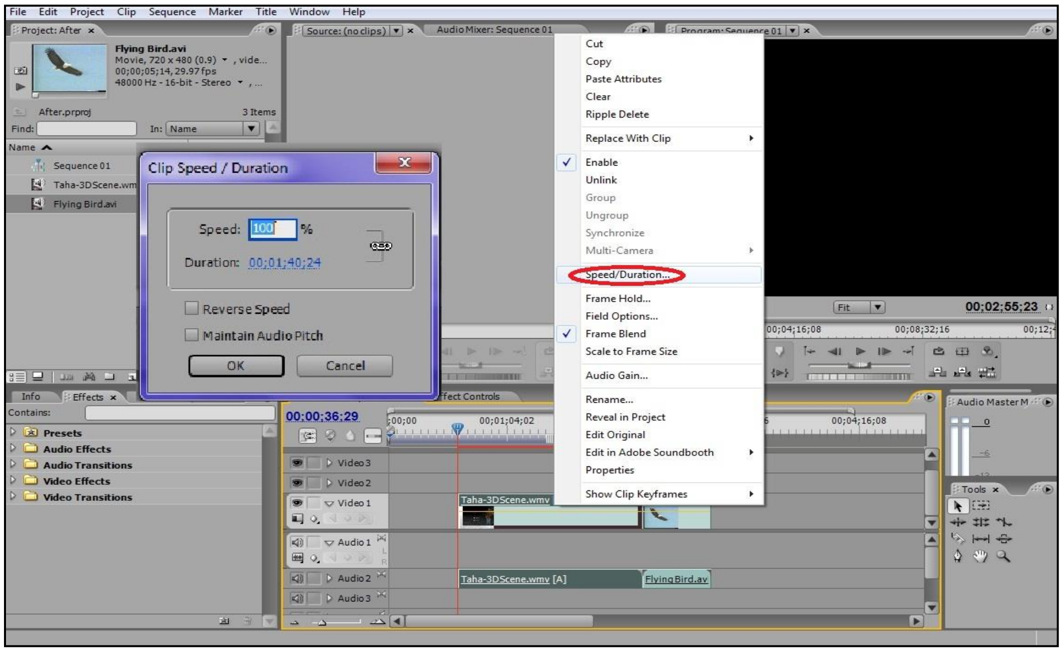
To show the final video after adding multiple tracks, press on enter and after rendering the video will be shown in the program monitor.

Modifying scenes:

When a certain scene is added to the timeline, different modification, effects and translations can be applied. For instance, the video speed can be changed by pressing right click and choosing Speed/Duration. Where speed is the original video percentage speed which is by default 100%. Increasing the ration will increase the speed and verse versa. While in duration the duration of a certain scene can be decided.

Note that when an audio is attached to a video the sound will be changed according to the speed/duration. To choose the video or the audio individually press on Alt key when selecting the file.

To apply the reverse of a certain video, check the Reverse Speed option. See the image below.



Clip Speed / Duration

Speed: 100 %

Duration: 00;01;40;24

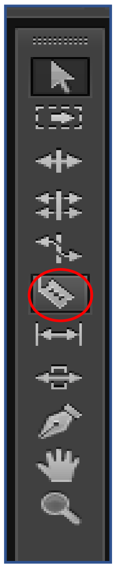
☐ Reverse Speed

☐ Maintain Audio Pitch

OK Cancel

Video clipping

To clip or extend a media file in timeline, move the mouse cursor to the end of the video as the shape changes to [or]. To split a certain scene to more than one clip, choose the razor tool and press on the video in the timeline to split.



Save and Open

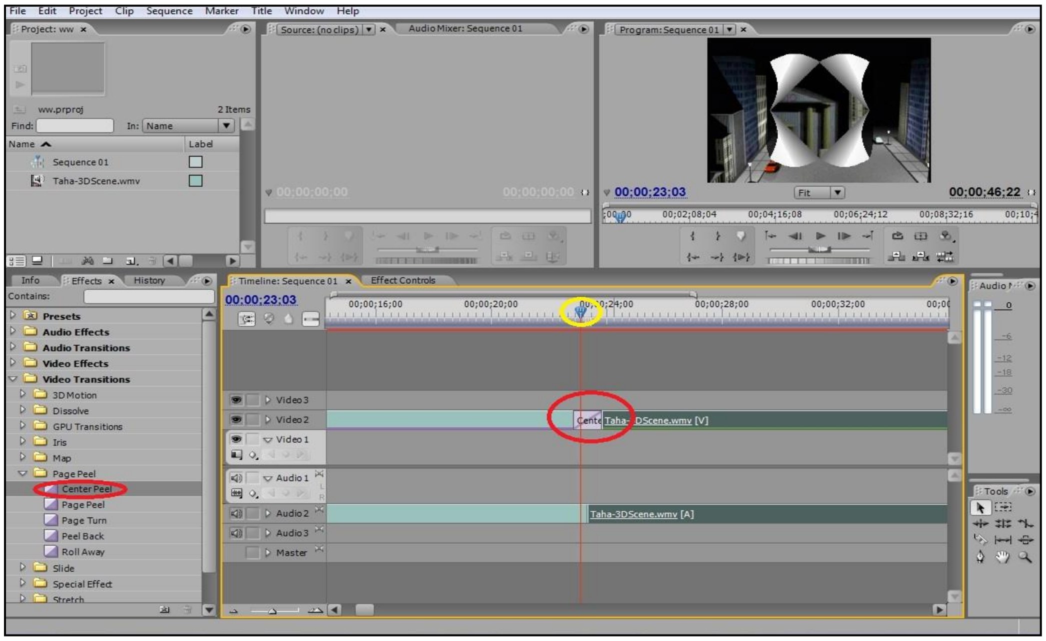
To save the project choose file→ Save, the file will be saved as a (.prproj) exetension.

Translations and Effects

Transitions means adding a transform between on scene and another or one sound and another.

To add a transition in video, go to the effect panel and choose video transition. The transition can be added to the beginning of the scene, between two scenes or at the end of the scene.

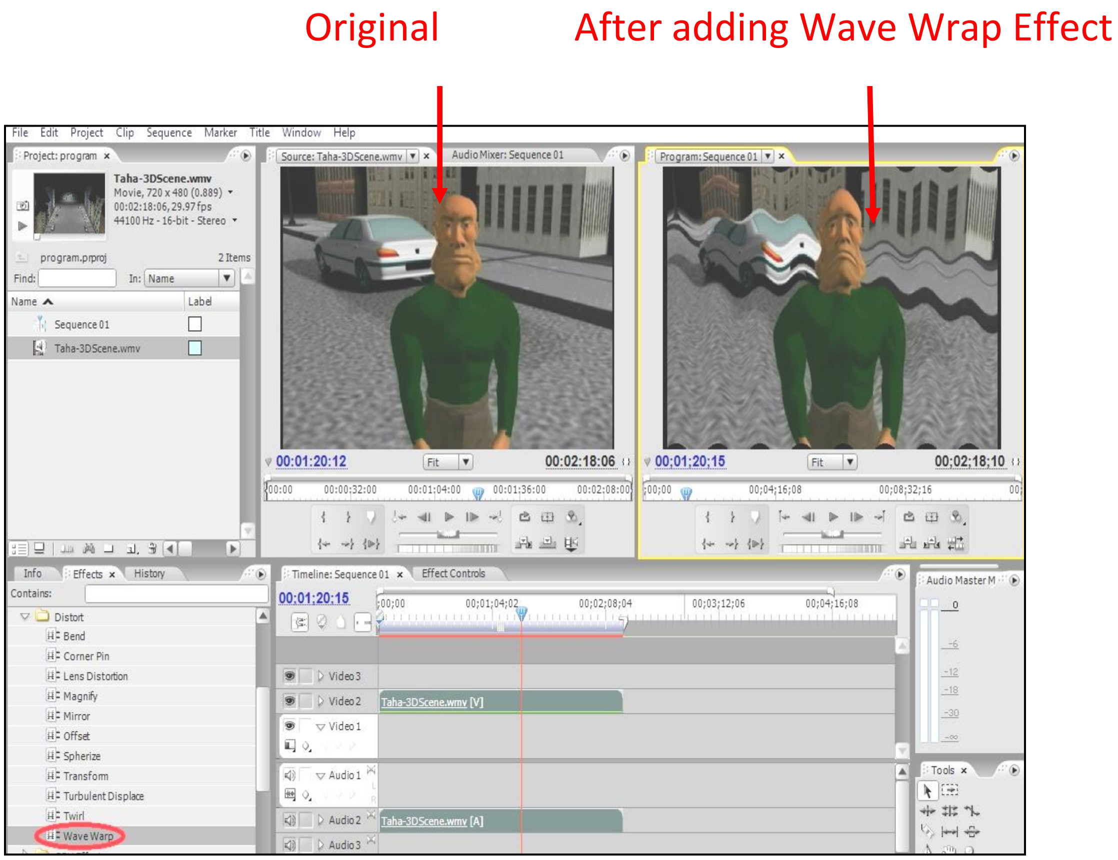
Choose the transition and add it between two scenes. A transition between the two will be shown. When selecting that transition, it can be modified in effect control panel, as extending the duration of the transition.



Video effect will change the appearance of the scene, while audio effect will change the sound attributes.

Adding an effect is similar to adding a transition, go to the effect panel and choose video effects. The transition can be added to scene it self.

Choose the effect and add to the scene. A change will be appeared instantly or after adding the scene. When selecting that effect, it can be modified in effect control panel.



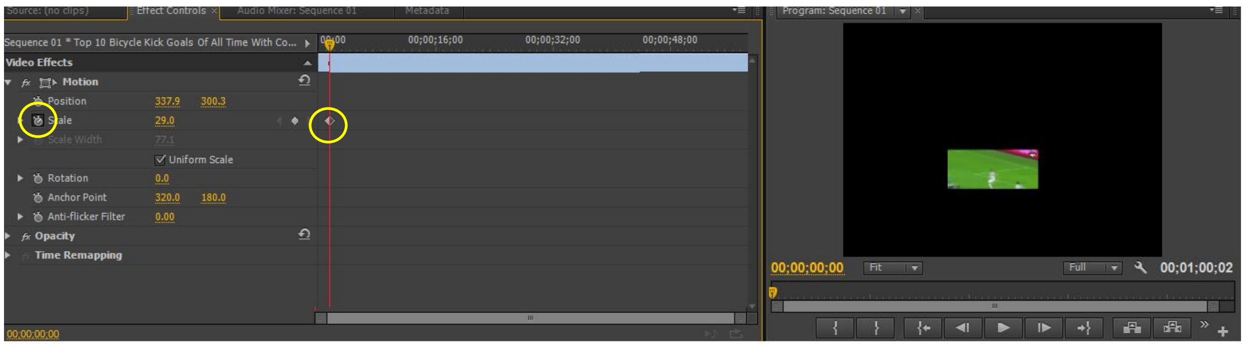
Animation

Animation is related to any change that occurred within time, as changing the position, size, opacity or any other effect within the time. An example of animation the size of a video is explained below.

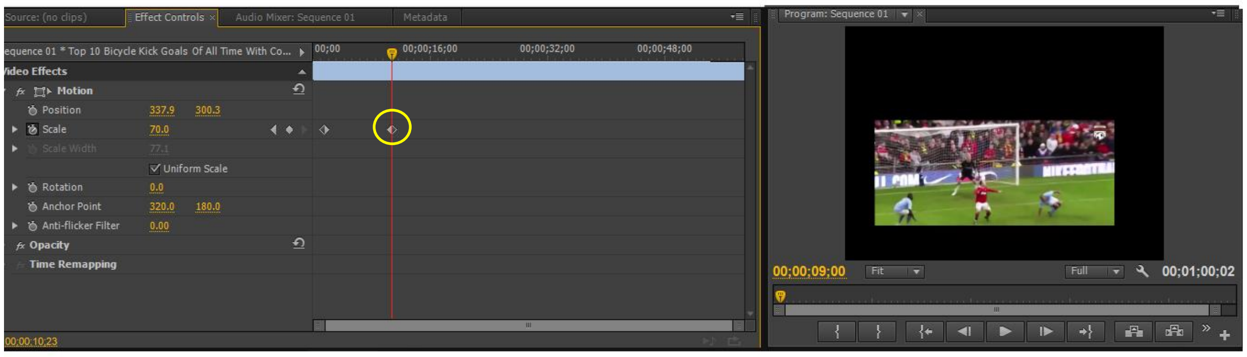
Import a scene and drag it to the time line. Click on the scene and choose Effect control (Window →Effect control).

A panel of the effects that can be applied on videos will be appeared, from the Motion option press on the white triangle to open the motion contents, choose scale and reduce it (if the video filling the screen it is changed to 29 instead of 100). In the same way the position, opacity and rotation can be changed.

To apply the change of the scale (for example) by time, click on the stop watch near the Scale to save the scale at the current frame and record any other change in next frames, a diamond sign will appear to show that there is a time frame at the current time. Move the time slider to another frame by dragging the slider in the time line or the effect control panel, then make a change in the scale, for example 70. Now between the two frames, the size will increase from 30% to 70%. This can be applied to any effect assigned to a stop watch.



Position 337.9 300.3
Scale 29.0
Scale Width 77.1
✓ Uniform Scale
Rotation 0.0
Anchor Point 320.0 180.0
Anti-flicker Filter 0.00
00;00;00;00
00;00;00;00
00;01;00;02



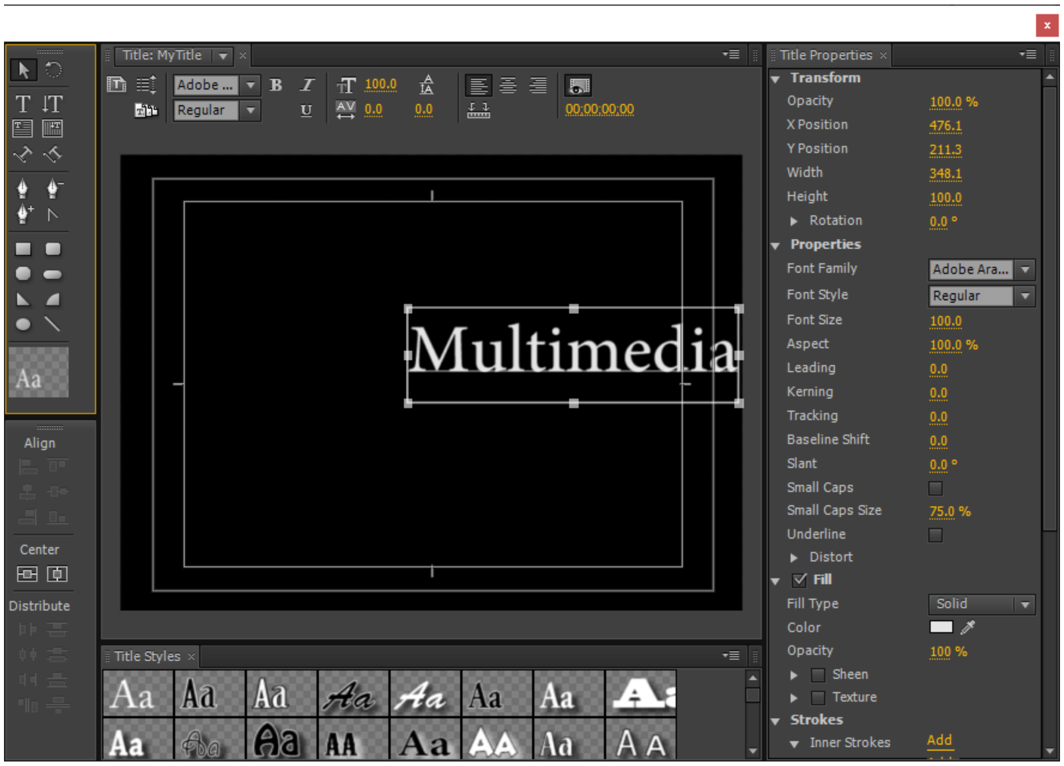
Position 337.9 300.3
Scale 70.0
Scale Width 77.1
✓ Uniform Scale
Rotation 0.0
Anchor Point 320.0 180.0
Anti-flicker Filter 0.00
00;00;10;23
00;00;09;00
00;01;00;02

Adding Text

To add a text to the scene, select title from menu bar and choose new title, choose default still and give the title a name to recognize it from other titles that might be added.

The title window will be displayed, write the title and change its size, font, orientation or any other feature as it is applied in text editors.

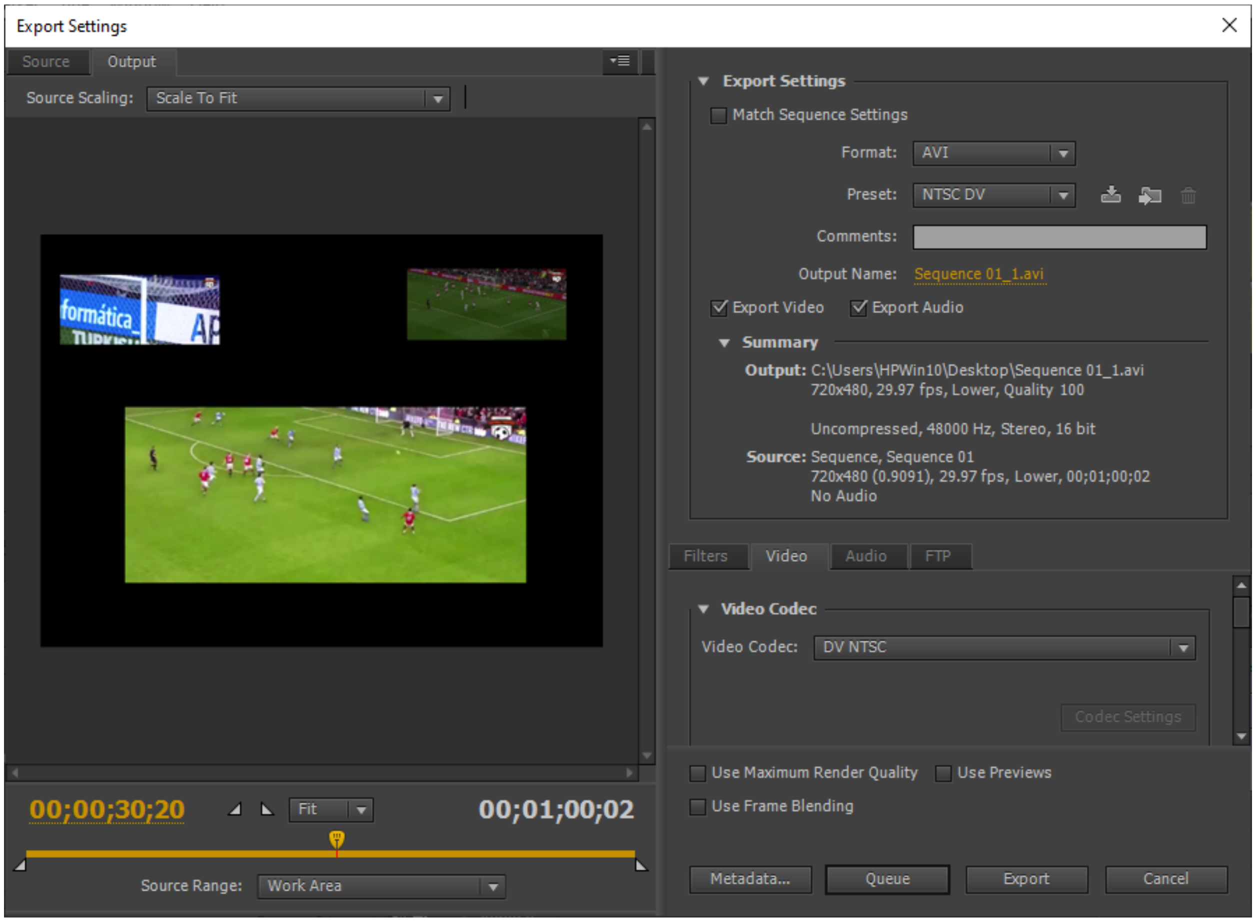
After writing the text, close the title window and it will appear in the project panel. Click and drag it to the time line (Video Tracks) as any other scene.



Transform	
Opacity	100.0 %
X Position	476.1
Y Position	211.3
Width	348.1
Height	100.0
Rotation	0.0 °
Properties	
Font Family	Adobe Ara...
Font Style	Regular
Font Size	100.0
Aspect	100.0 %
Leading	0.0
Kerning	0.0
Tracking	0.0
Baseline Shift	0.0
Slant	0.0 °
Small Caps	☐
Small Caps Size	75.0 %
Underline	☐
Distort	
✓ **Fill**	
Fill Type	Solid
Color	
Opacity	100 %
☐ Sheen	
☐ Texture	
Strokes	
Inner Strokes	Add

Exporting Scenes

To export the file after rendering it (By pressing Enter key), choose Export from File in the menu bar. The Export setting window will be shown, choose the video format from Format and select the name and location of the saved output by clicking on the name beside Output Name, it is set as (Sequence 01) by default, and press on Export.



Source Scaling: Scale To Fit
00;00;30;20
Fit
00;01;00;02
Source Range: Work Area

Export Settings
Match Sequence Settings
Format: AVI
Preset: NTSC DV
Comments:
Output Name: Sequence 01_1.avi
✓ Export Video ✓ Export Audio
Summary
Output: C:\Users\HPWin10\Desktop\Sequence 01_1.avi
720x480, 29.97 fps, Lower, Quality 100

Uncompressed, 48000 Hz, Stereo, 16 bit
Source: Sequence, Sequence 01
720x480 (0.9091), 29.97 fps, Lower, 00;01;00;02
No Audio

Video Codec
Video Codec: DV NTSC

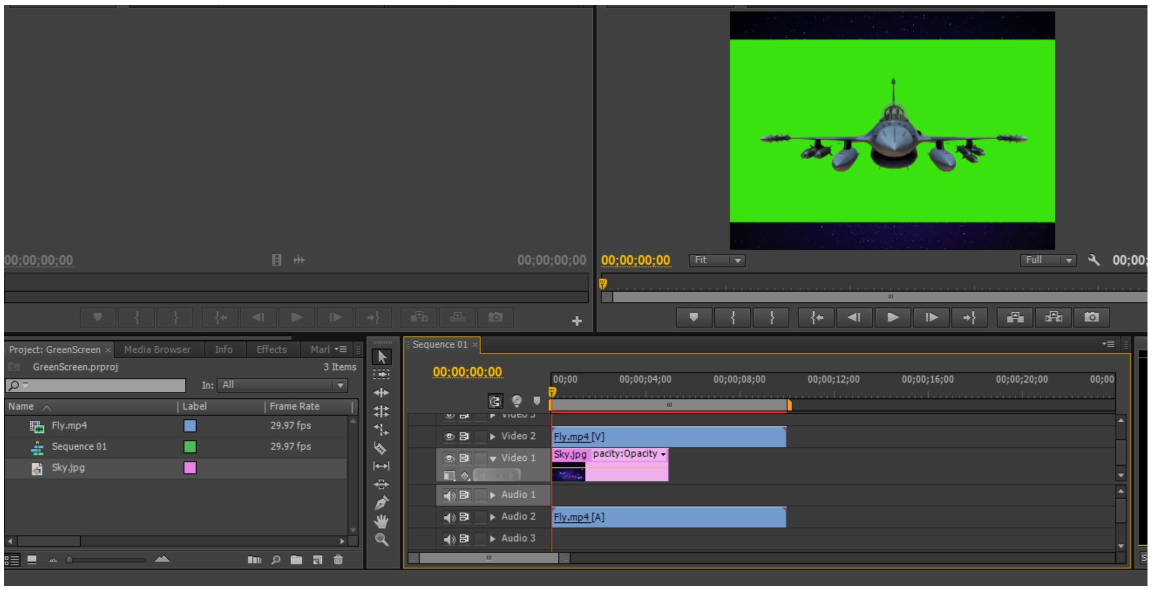
Use Maximum Render Quality
Use Previews
Use Frame Blending

Video Tricks

A. Green Screen (Keying)

1- Import two scenes to the project file, the first on with the green background, for example the file fly.mp4, and the other is the background movie or image, in this example Sky.jpg image.

2- Add both scenes to the time line, the video with the green background should be on the top track.

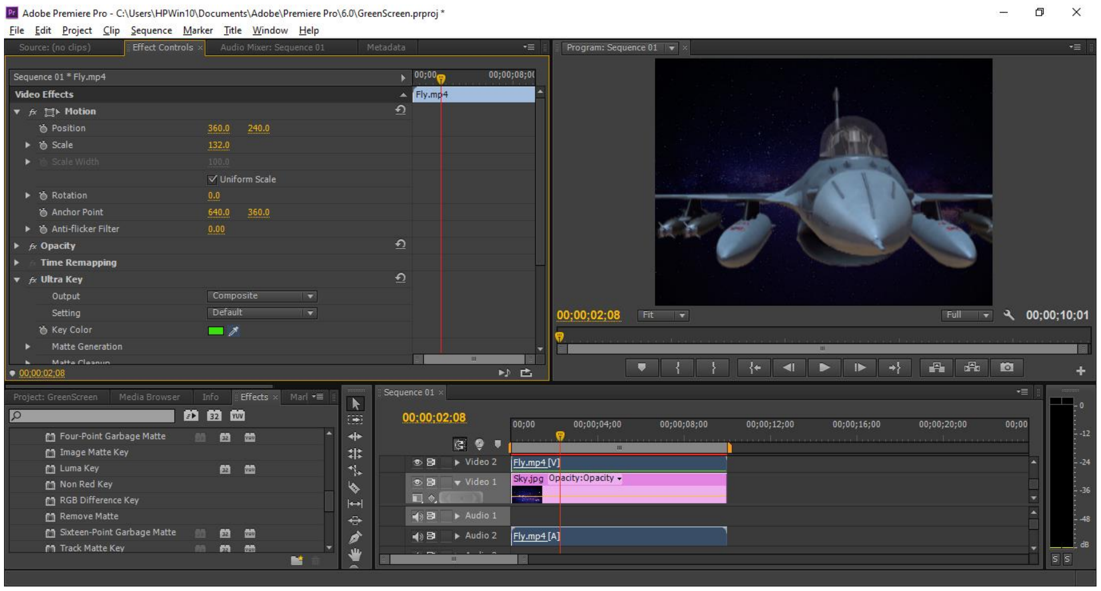


Name	Label	Frame Rate
Fly.mp4	🟦	29.97 fps
Sequence 01	🟩	29.97 fps
Sky.jpg	🟪	

3- Resize the video if required and modify the length of the background (Sky image in the example) to be the same length of the top video.

4- Select the top video. Go to effect and choose video effects→ Keying → Ultra Key.

5- In the effect control window, choose the color picker from key color and move it to the green background. The green color will be disappeared to show the background.


Adobe Premiere Pro - C:\Users\HPWin10\Documents\Adobe\Premiere Pro\6.0\GreenScreen.prproj *
File Edit Project Clip Sequence Marker Title Window Help
Source: (no clips)
Effect Controls
Audio Mixer: Sequence 01
Metadata
Program: Sequence 01
Project: GreenScreen
Media Browser
Info
Effects
Sequence 01

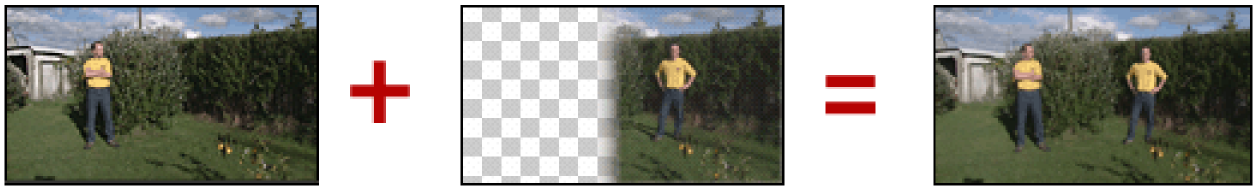
```
Sequence 01 * Fly.mp4
Video Effects
  Motion
    Position            360.0   240.0
    Scale               132.0
    Scale Width         100.0
                        ✓ Uniform Scale
    Rotation            0.0
    Anchor Point        640.0   360.0
    Anti-flicker Filter 0.00
  Opacity
  Time Remapping
  Ultra Key
    Output              Composite
    Setting             Default
    Key Color           🟩
    Matte Generation
00;00;02;08
```

```
Four-Point Garbage Matte
Image Matte Key
Luma Key
Non Red Key
RGB Difference Key
Remove Matte
Sixteen-Point Garbage Matte
Track Matte Key
```

```
00;00;02;08
Video 2   Fly.mp4 [V]
Video 1   Sky.jpg  Opacity:Opacity
Audio 1
Audio 2   Fly.mp4 [A]
```

B. Character Duplication

1. Use two videos captured by same camera position and lighting.
2. Capture the video as each time the character is on a different location that is not interfere with the other.
3. Add each video in one track.
4. Use the crop effect and crop the top video from the place where the character is not shown.
5. The result is a duplicated character as shown in the figure. You may create a conversation between the character and the duplicate.



5-Keyboard Interaction

Two functions are used to interact with keyboard input one for special keys (like arrows) and other for traditional keys. Both are listed in the example. Here the object will be rotated clock wise while pressing the key 'a' from keyboard, while it will be rotated anti clock wise when pressing the left key.

H.W : Write OpenGL code to move a certain object to four directions according to following keyboard keys:
W : To move forward.
S : To move backward.
A : Moving to right.
D : Moving to left.

Also add the clockwise rotation by the key R.

```
//keyboard interaction
#include <windows.h>
#include <GL/glut.h>

GLfloat rot = -1.0f;

void idle() {
   glutPostRedisplay();
}
void display()
 {
   glPushMatrix();
   glClearColor(1.0f, 1.0f, 0.5f, 1.0f);
   glClear(GL_COLOR_BUFFER_BIT);
   glRotatef(rot, 0.0f, 0.0f,1.0f);

    glBegin(GL_TRIANGLES);
       glColor3f(1.0f,0.0f,0.0f);
       glVertex2f(0,-0.5f);
       glVertex2f(0.5f,0);
       glVertex2f(0.0f,0.5);
    glEnd();

glPopMatrix();
glFlush();
}

void specialKeys(int key, int x, int y) {
   switch (key) {
     case GLUT_KEY_LEFT:
rot = rot + 0.9;
if (rot>360)
       rot=-1;
         break;
   }
}
void keyboard(unsigned char key, int x, int
y) {
   switch (key) {
      case 'a':
         rot = rot - 0.9;
if (rot>360)
       rot=-1;
         break;
   }
}



void main(int argc, char** argv)
{
   glutInit(&argc, argv);
   glutInitWindowSize(800,600);
   glutCreateWindow("My OpenGL Window");
   glutDisplayFunc(display);
   glutIdleFunc(idle);
   glutSpecialFunc(specialKeys);
   glutKeyboardFunc(keyboard);
   glutMainLoop();
}
```

References:

https://www3.ntu.edu.sg/home/ehchua/programming/opengl/CG_Introduction.html#zz-5.

Shreiner, D., Sellers, G., Kessenich, J., & Licea-Kane, B. (2013). OpenGL programming guide: The Official guide to learning OpenGL, version 4.3. Addison-Wesley.