Chapter 2
A Taste of Multimedia

2.1 Multimedia Tasks and Concerns

- More basic concerns will impact multimedia as it now appears in products e.g., we expect impact from Computer Vision: a camera owner might be encouraged to think like a computer scientist and ask:
  - "What is going on in an image?"
    - A less high-level question is "Where has this image been taken?" ***(scene recognition)***
  - "Does the image contain a particular object?" ***(object classification)***
  - "Where is an object of interest?" ***(object detection)***
  - "Which object does each pixel belong to?" ***(image segmentation)***

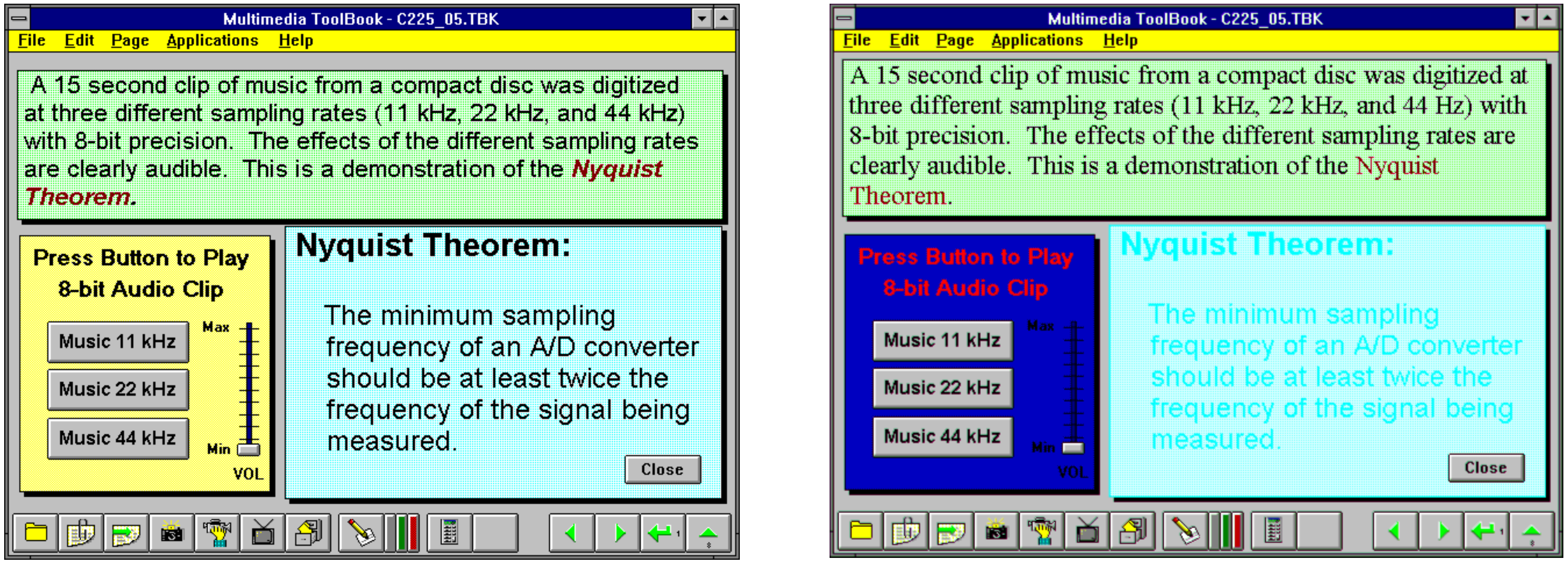
== a classic Computer Vision hierarchy of high-level to detailed description of an image, with scene recognition at the top and image segmentation at the bottom.

2.2 Multimedia Presentation

- **Graphics Styles**: Human visual dynamics impact how presentations must be constructed.

  a) **Color principles and guidelines**: Some color schemes and art styles are best combined with a certain theme or style. A general hint is to not use *too many* colors, as this can be distracting.

  b) **Fonts**: For effective visual communication in a presentation, it is best to use large fonts (i.e., 18 to 36 points), and no more than 6 to 8 lines per screen (*fewer than on this screen*!). Fig. 2.1 shows a comparison of two screen projections:


Multimedia ToolBook - C225_05.TBK
File Edit Page Applications Help
Max
Min
VOL
Music 11 kHz
Music 22 kHz
Music 44 kHz
Close

A 15 second clip of music from a compact disc was digitized at three different sampling rates (11 kHz, 22 kHz, and 44 kHz) with 8-bit precision. The effects of the different sampling rates are clearly audible. This is a demonstration of the ***Nyquist Theorem.***

Press Button to Play 8-bit Audio Clip

Nyquist Theorem:

The minimum sampling frequency of an A/D converter should be at least twice the frequency of the signal being measured.

A 15 second clip of music from a compact disc was digitized at three different sampling rates (11 kHz, 22 kHz, and 44 Hz) with 8-bit precision. The effects of the different sampling rates are clearly audible. This is a demonstration of the Nyquist Theorem.

Press Button to Play 8-bit Audio Clip

Nyquist Theorem:

The minimum sampling frequency of an A/D converter should be at least twice the frequency of the signal being measured.

Fig. 2.1: Colours and fonts [from Ron Vetter].

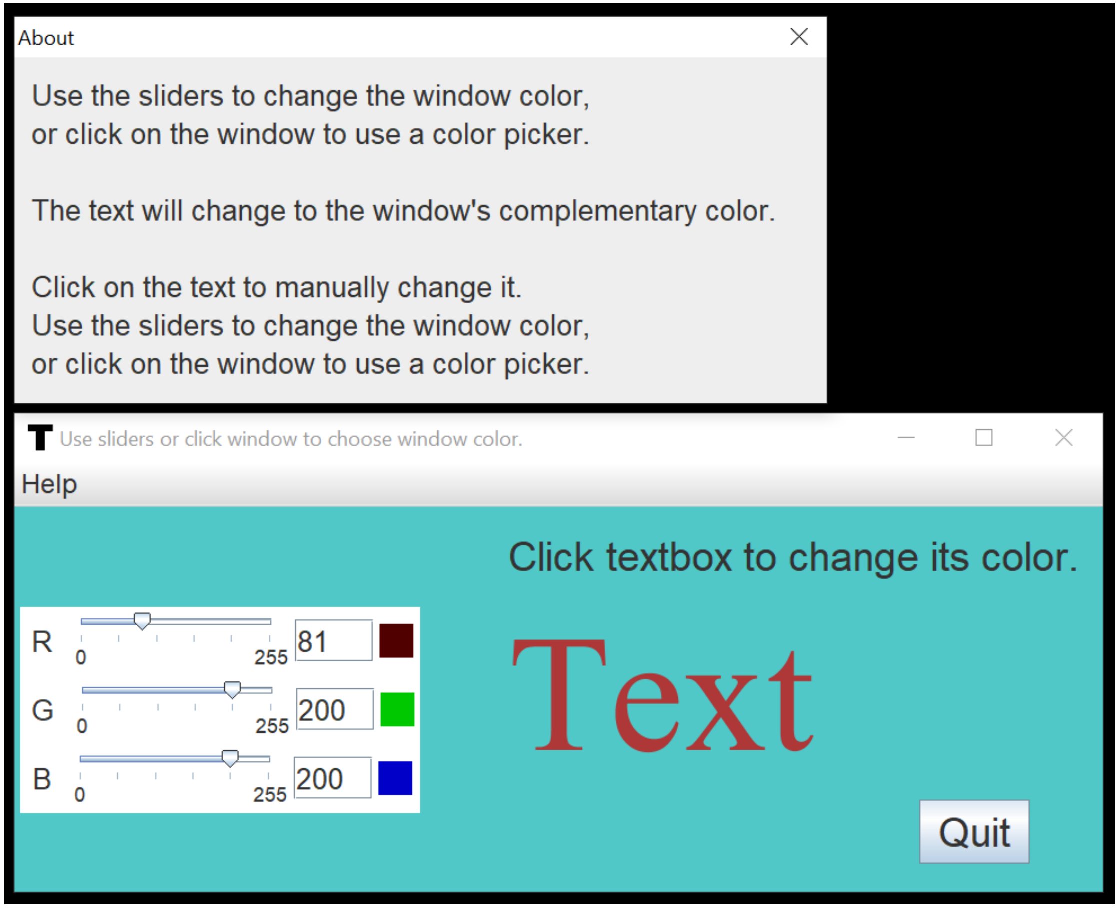
c) **A color contrast program**: If the text color is some triple (R,G,B), a legible color for the background is that color subtracted from the maximum (here assuming max=1):

$$(R, G, B) \rightarrow (1 - R, 1 - G, 1 - B) \qquad (2.1)$$

- Some color combinations are more pleasing than others; e.g., a pink background and forest green foreground, or a green background and mauve foreground.

- Fig. 2.2 shows a small Java program in operation:
  →Link to TextColor src.zip

  →Link to textcolor.exe



About

Use the sliders to change the window color,
or click on the window to use a color picker.

The text will change to the window's complementary color.

Click on the text to manually change it.
Use the sliders to change the window color,
or click on the window to use a color picker.

Use sliders or click window to choose window color.
Help
Click textbox to change its color.
R 0 255 81
G 0 255 200
B 0 255 200
Text
Quit

Fig. 2.2: Program to investigate colors and readability.

- Fig. 2.3, shows a "colour wheel", with opposite colors equal to (1-R, 1-G, 1-B)

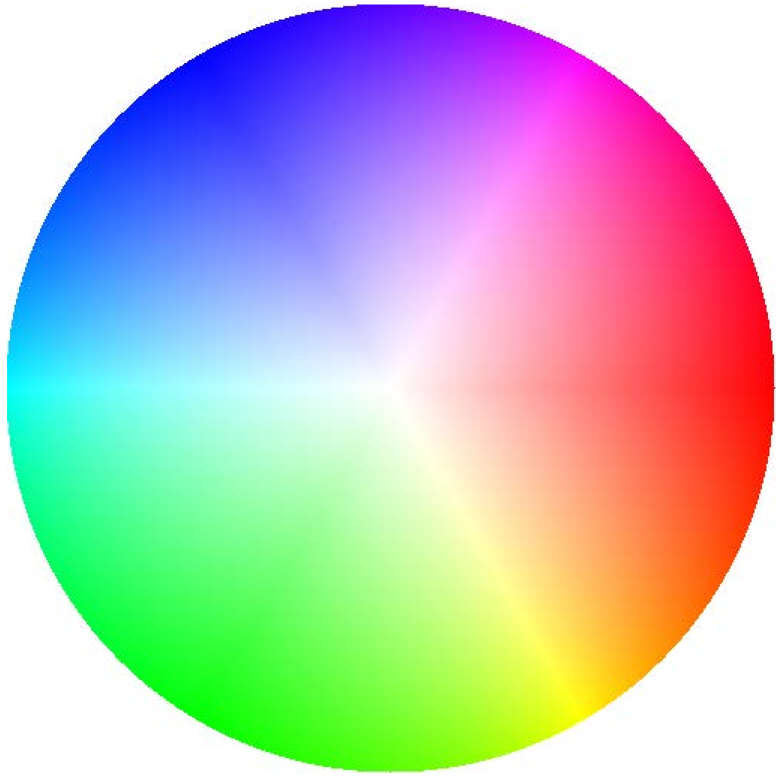


Fig. 2.3: Colour wheel

Sprite Animation

- **The basic idea**: Suppose we have an animation figure, as in Fig. 2.4 (a). Now create a 1-bit mask *M*, as in Fig. 2.4 (b), black on white, and accompanying sprite *S*, as in Fig. 2.4 (c).

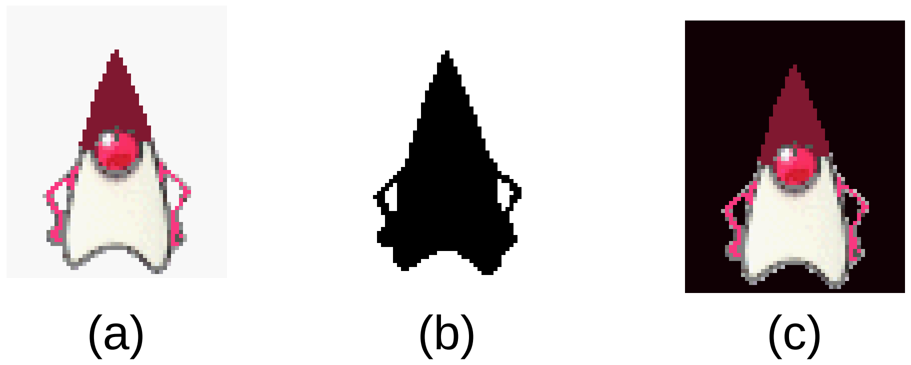


Fig. 2.4: Sprite creation: Original, mask image *M*, and sprite *S* (*"Duke" figure courtesy of Sun Microsystems*.)

- We can overlay the sprite on a colored background *B*, as in Fig. 2.5 (a) by first **AND**ing *B* and *M*, and then **OR**ing the result with *S*, with final result as in Fig. 2.5 (e).

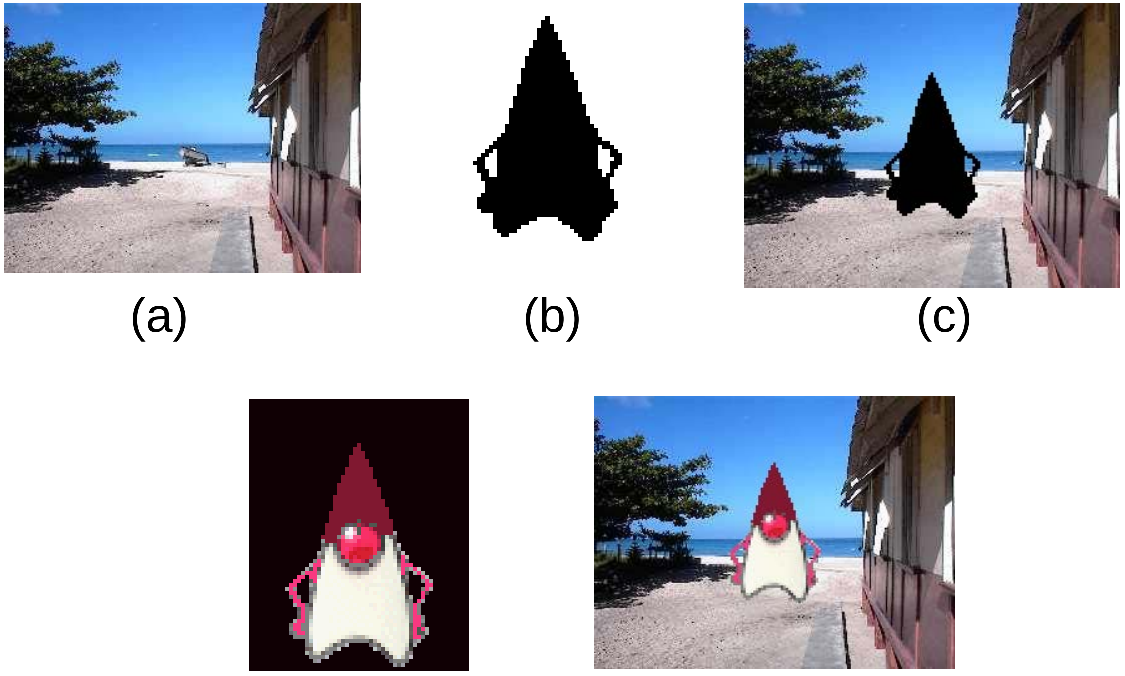


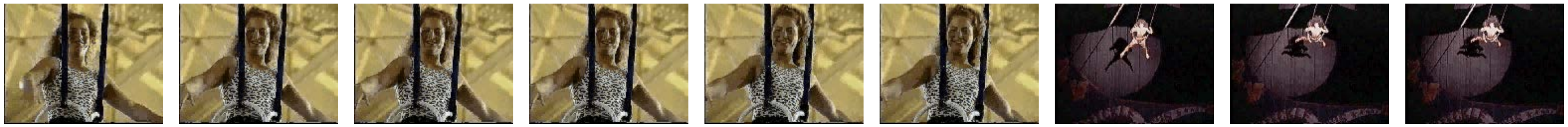
Fig. 2.5: Sprite animation: (a): Background *B*. (b): Mask *M* (c): *B* **AND** *M*. (d): Sprite *S* (e): *B* **AND** *M* **OR** *S*

Video Transitions

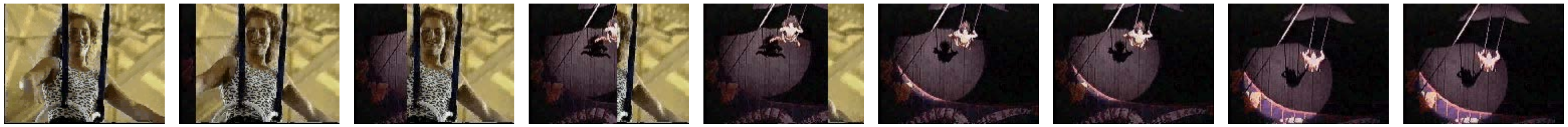
- **Video transitions**: to signal "scene changes".

- Many different types of transitions:

1. **Cut**: an abrupt change of image contents formed by abutting two video frames consecutively. This is the simplest and most frequently used video transition.



2. **Wipe**: a replacement of the pixels in a region of the viewport with those from another video. Wipes can be left-to-right, right-to-left, vertical, horizontal, like an iris opening, swept out like the hands of a clock, etc.



3. **Dissolve**: replaces every pixel with a mixture over time of the two videos, gradually replacing the first by the second. Most dissolves can be classified as two types: **cross dissolve** and **dither dissolve**.

Type I: Cross Dissolve

- Every pixel is affected gradually. It can be defined by:

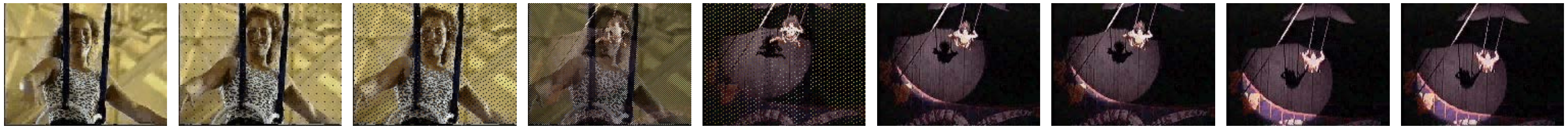
$$\mathbf{D} = (1 - \alpha(t)) \cdot \mathbf{A} + \alpha(t) \cdot \mathbf{B} \qquad (2.2)$$

where **A** and **B** are the color 3-vectors for video **A** and video **B**. Here, $a(t)$ is a transition function, which is often linear:

$$\alpha(t) = k \cdot t, \quad \text{with } k \cdot t_{max} \equiv 1 \qquad (2.3)$$

Type II: Dither Dissolve

- Determined by $a(t)$, increasingly more and more pixels in video A will abruptly (instead of gradually as in Type I) change to video B.



- Fade-in and fade-out are special types of Type I dissolve: video A or B is black (or white). Wipes are special forms of Type II dissolve in which changing pixels follow a particular geometric pattern.

- Build-your-own-transition: Suppose we wish to build a special type of wipe which slides one video out while another video slides in to replace it: a *slide* (or *push*).

(a) Unlike a wipe, we want each video frame not be held in place, but instead move progressively farther into (out of) the viewport.

(b) Suppose we wish to slide Video_L in from the left, and push out Video_R. Figure 2.6 shows this process:

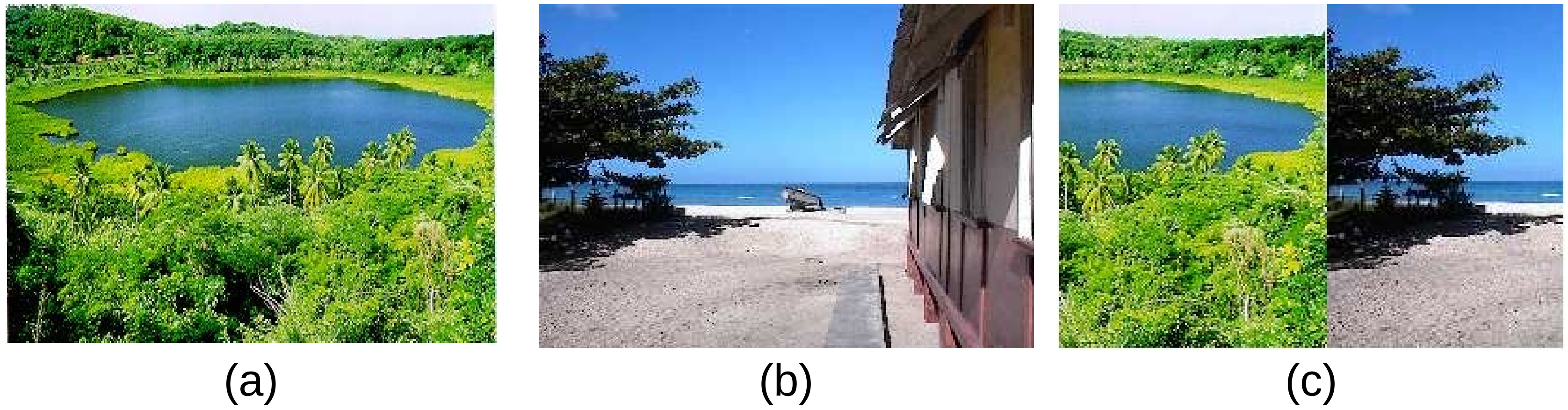


Fig. 2.6: (a): Video_L. (b): Video_R. (c): Video_L sliding into place and pushing out Video_R.

Slide Transition (Cont'd)

- As time goes by, the horizontal location x_T for the transition boundary moves across the viewport from $x_T = 0$ at $t = 0$ to $x_T = x_{max}$ at $t = t_{max}$. Therefore, for a transition that is linear in time,

$$x_T = (t/t_{max})x_{max}$$

- So for any time *t* the situation is as shown in Fig. 2.7 (a). Let's assume that dependence on *y* is implicit since we use the same *y* as in the source video. Then for the red channel (and similarly for the green and blue), $R = R(x, t)$.

- Suppose that we have determined that pixels should come from Video$_L$.

  Then the *x*-position x_L in the unmoving video should be, $x_L = x + (x_{max} - x_T)$, where x is the position we are trying to fill in the viewport, x_T is the position in the viewport that the transition boundary has reached, and x_{max} is the maximum pixel position for any frame.

- From Fig. 2.7(b), we can calculate the position x_L in Video$_L$'s coordinate system as the sum of the distance *x*, in the viewport, plus the difference $x_{max} - x_T$.

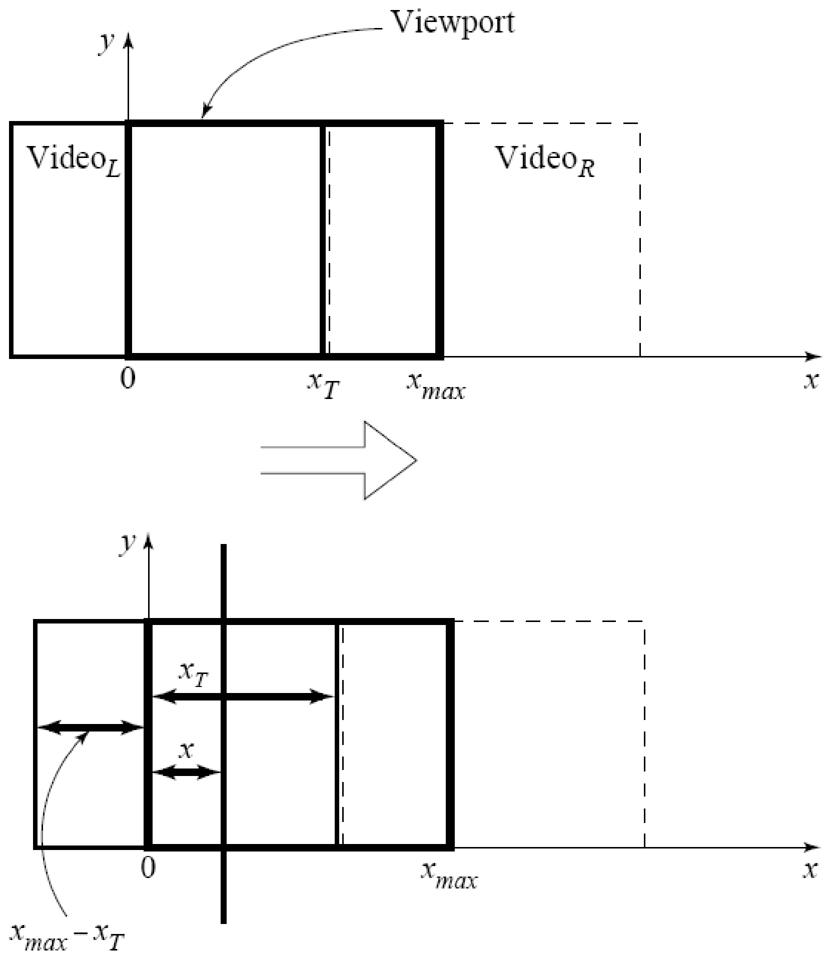


Fig. 2.7:

a) Geometry of Video$_L$ pushing out Video$_R$.

b) Calculating position in Video$_L$ from where pixels are copied to the viewport.

Slide Transition (Cont'd)

- Substituting the fact that the transition moves linearly with time, $x_T = x_{max}(t/t_{max})$, a pseudocode solution in shown in Fig. 2.8.

Fig 2.8: Pseudocode for slide video transition

```
1: for $t$ in $0 .. t_{max}$ do
2:     for $x$ in $0 .. x_{max}$ do
3:         if $(\frac{x}{x_{max}} < \frac{t}{t_{max}})$ then
4:             $R = R_L(x + x_{max} * [1 - \frac{t}{t_{max}}],\ t)$
5:         else
6:             $R = R_R(x - x_{max} * \frac{t}{t_{max}},\ t)$
```

Table 2.1: Uncompressed video sizes.

Standard Definition Video		
640×480 full color	=	922 kB/frame
@ 30 frames/sec	=	28 MB/sec
	=	221 Mb/sec
× 3,600 sec/hour	=	100 GB/hour
High Definition Video		
1,920×1,080 full color	=	6.2 MB/frame
@ 30 frames/sec	=	187 MB/sec
	=	1.5 Gb/sec
× 3,600 sec/hour	=	672 GB/hour

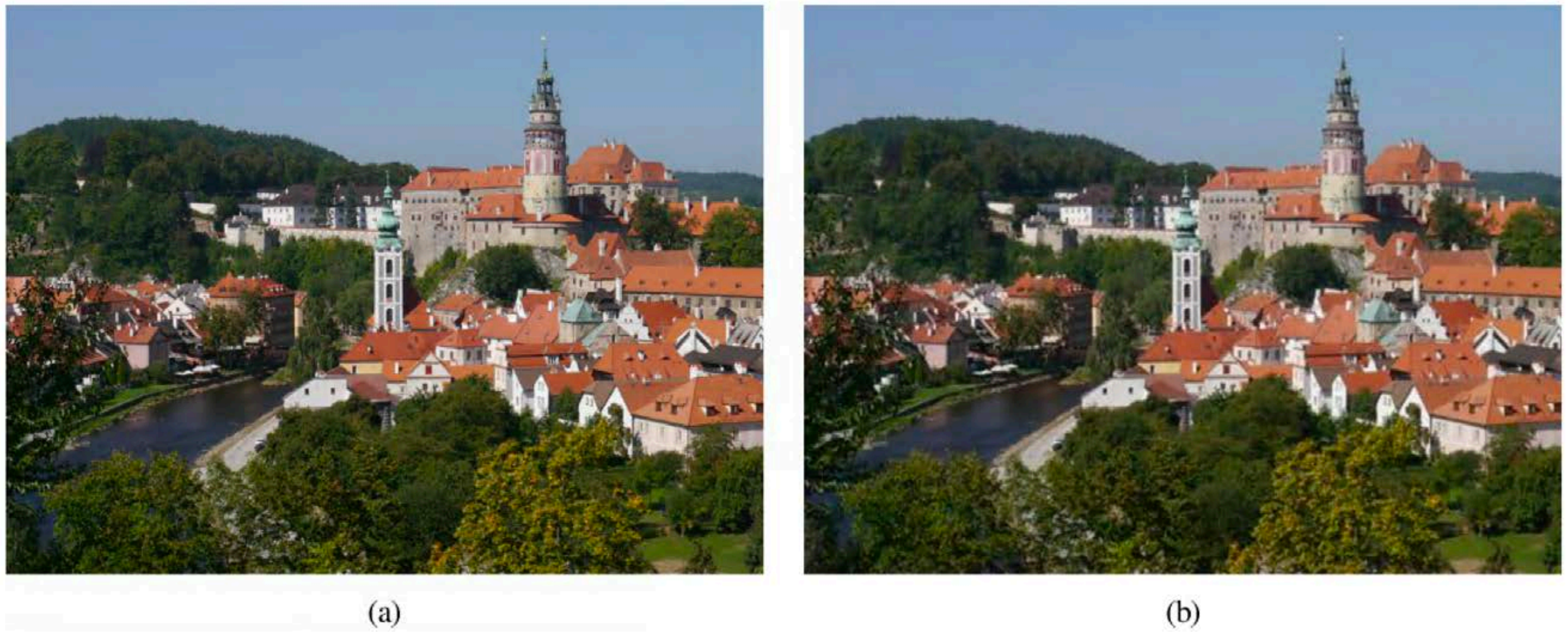


Fig. 2.9: JPEG compression: (a) original uncompressed image; (b) JPEG compression with Quality Factor *qf* = 75 (the typical default).

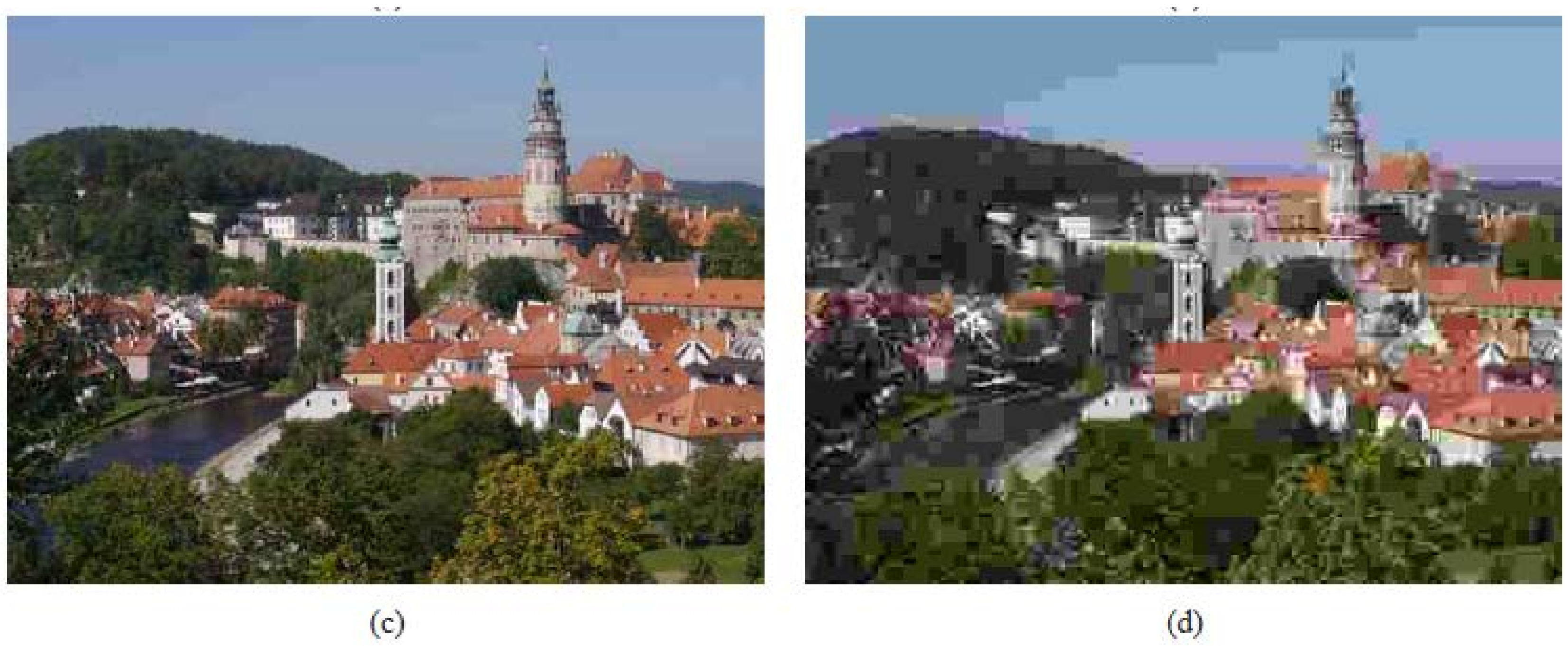
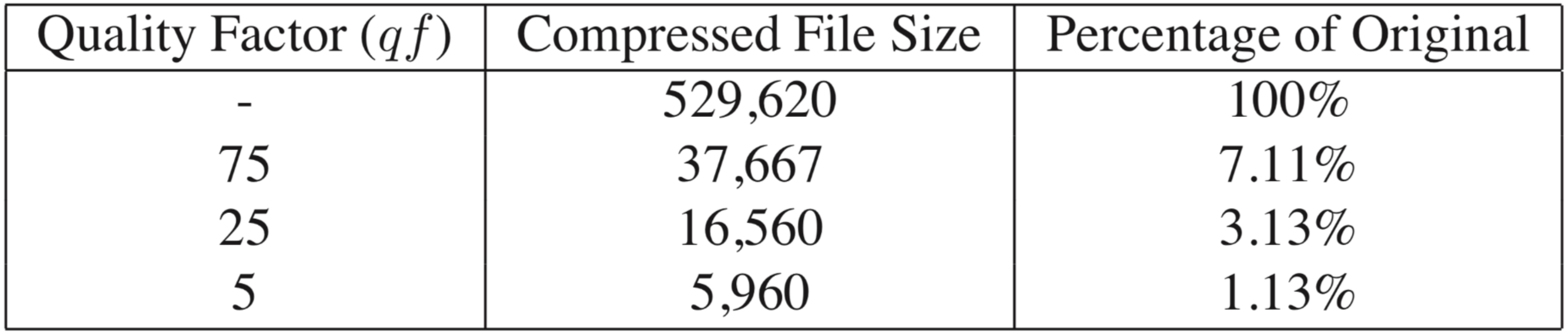


Fig. 2.9: JPEG compression: (c) Quality Factor *qf* = 25 and (d) Quality Factor *qf* = 5.

Table 2.2: JPEG file sizes (bytes) and percentage size of data for JPEG compression with Quality Factor *qf* = 75, 25, and 5.



Quality Factor (qf)	Compressed File Size	Percentage of Original
-	529,620	100%
75	37,667	7.11%
25	16,560	3.13%
5	5,960	1.13%

- How expensive image and video processing is in terms of CPU cycles?

- Suppose we have an image whose pixels we wish to darken, by a factor of 2:

Algorithm 2.1: Darken by a factor of 2

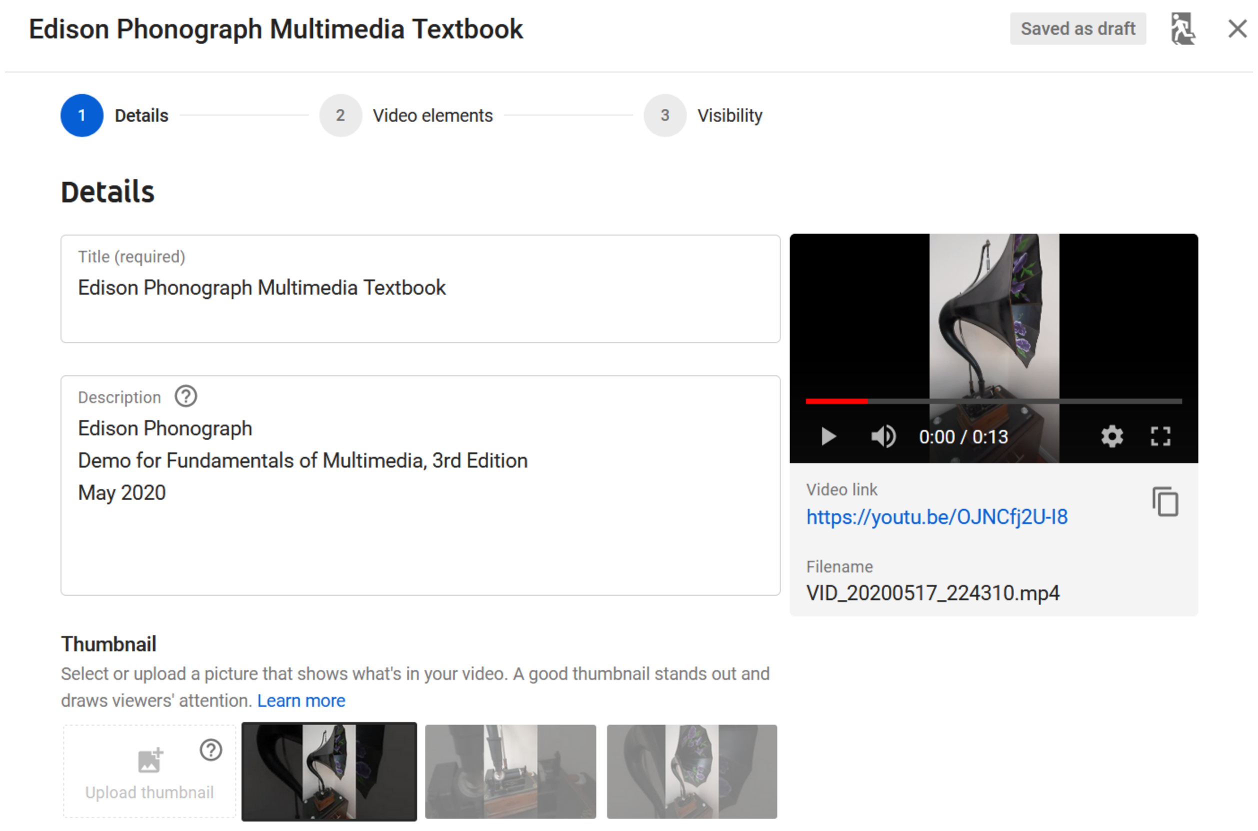
```
for $x = 1$ to $columns$ do
    for $y = 1$ to $rows$ do
        image[x,y].red /= 2
        image[x,y].green /= 2
        image[x,y].blue /= 2
```

On a RISC machine, 12 instructions per pixel, i.e., per 3 bytes.

2.4 Multimedia Production

- Multimedia production can easily involve an art director, graphic designer, production artist, producer, project manager, writer, user interface designer, sound designer, videographer, and 3D and 2D animators, as well as programmers.

2.5 Multimedia Sharing and Distribution



Edison Phonograph Multimedia Textbook

Saved as draft

Details

Title (required)
Edison Phonograph Multimedia Textbook

Description
Edison Phonograph
Demo for Fundamentals of Multimedia, 3rd Edition
May 2020

0:00 / 0:13

Video link
https://youtu.be/OJNCfj2U-I8

Filename
VID_20200517_224310.mp4

Thumbnail
Select or upload a picture that shows what's in your video. A good thumbnail stands out and draws viewers' attention. Learn more

Fig. 2.10: The webpage for uploading a YouTube video. The video, titled "Edison Phonograph Multimedia Textbook 3rd Edition" is open to all users (Privacy settings: Public) and can be searched in the YouTube homepage using the title or tags. Note that the video thumbnails are automatically generated by YouTube.

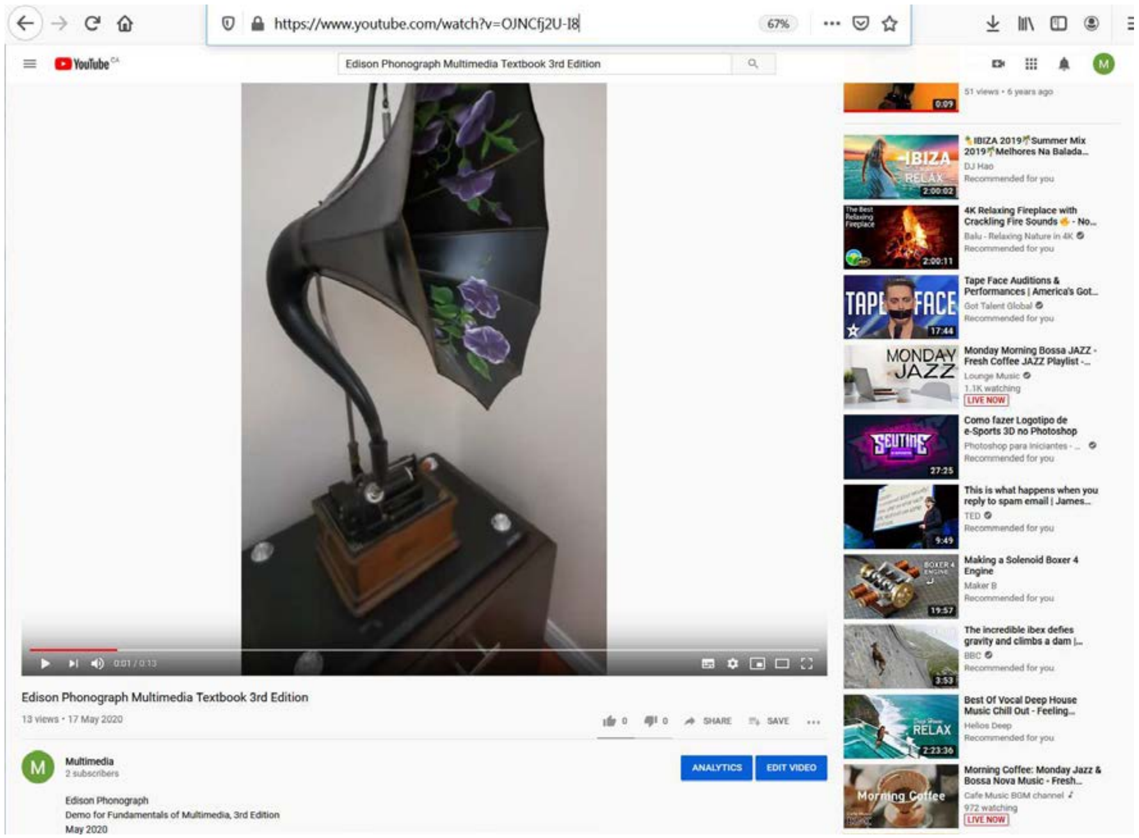

https://www.youtube.com/watch?v=OJNCfj2U-I8
67%
YouTube
Edison Phonograph Multimedia Textbook 3rd Edition
0:01 / 0:13
Edison Phonograph Multimedia Textbook 3rd Edition
13 views • 17 May 2020
SHARE
SAVE
Multimedia
2 subscribers
ANALYTICS
EDIT VIDEO
Edison Phonograph
Demo for Fundamentals of Multimedia, 3rd Edition
May 2020
IBIZA 2019 Summer Mix 2019 Melhores Na Balada...
4K Relaxing Fireplace with Crackling Fire Sounds - No...
Tape Face Auditions & Performances | America's Got...
Monday Morning Bossa JAZZ - Fresh Coffee JAZZ Playlist -...
Como fazer Logotipo de e-Sports 3D no Photoshop
This is what happens when you reply to spam email | James...
Making a Solenoid Boxer 4 Engine
The incredible ibex defies gravity and climbs a dam |...
Best Of Vocal Deep House Music Chill Out - Feeling...
Morning Coffee: Monday Jazz & Bossa Nova Music - Fresh...
LIVE NOW

Fig. 2.11: The YouTube page for the video uploaded. The list of related videos are shown on the right side, and users can post their comments, as well.

2.6 Some Useful Editing and Authoring Tools

- One needs real vehicles for showing understanding principles of and creating multimedia. And straight programming in C++ or Java is not always the best way of showing your knowledge and creativity.

- Some popular authoring tools include the following:
  - **Adobe Premiere**
  - **HTML5 Canvas**
  - **Adobe Director**
  - **Adobe XD**

- **Hint for studying this section**: Hands-on work in a laptop or Lab environment, with reference to the text.

Adobe Premiere

- Used for video editing and exporting. Features a "score" authoring metaphor, placing video clips in horizontal tracks that resemble a musical score.

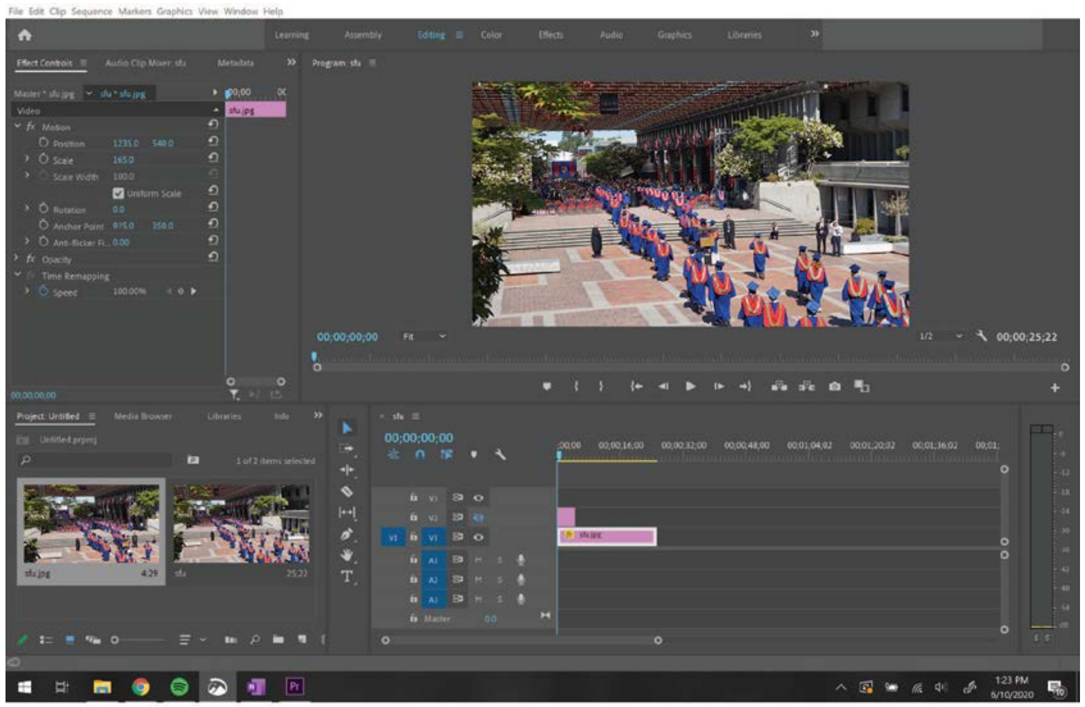
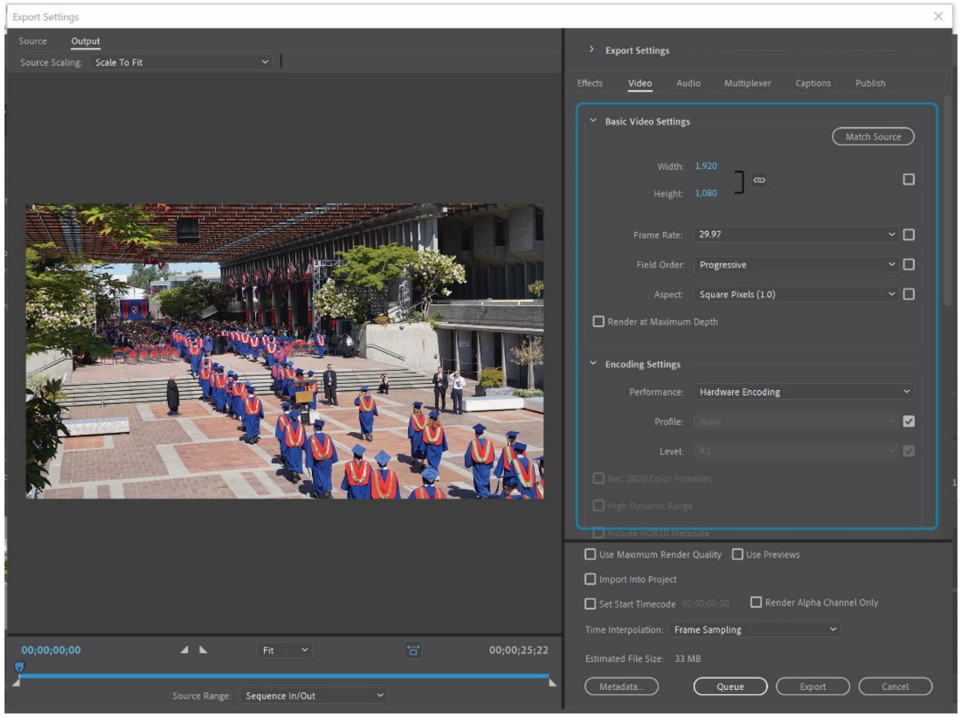


Fig. 2.12: Adobe Premiere Screen.

Adobe Premiere

- As a professional video editor, Premiere offers many controls for exporting and compressing the final product.



Source Scaling: Scale To Fit

Basic Video Settings
Width: 1,920
Height: 1,080
Frame Rate: 29.97
Field Order: Progressive
Aspect: Square Pixels (1.0)
Render at Maximum Depth

Encoding Settings
Performance: Hardware Encoding
Profile: Main
Level: 4.1
Rec. 2020 Color Primaries
High Dynamic Range
Include HDR10 Metadata

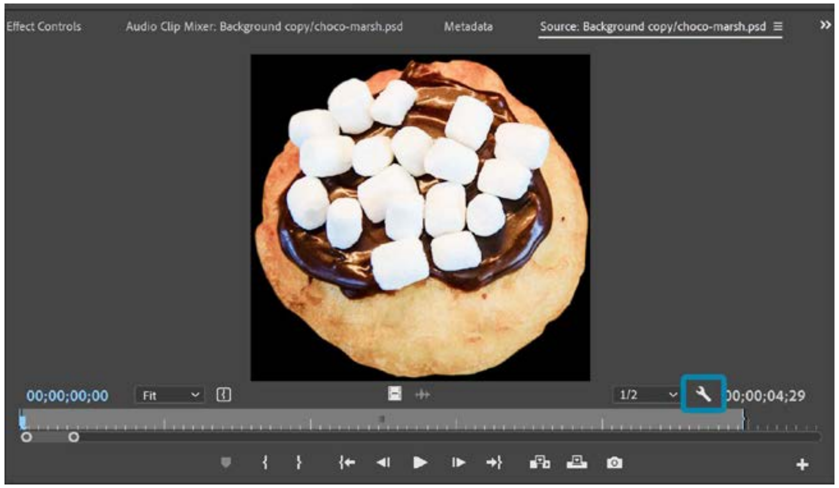
Use Maximum Render Quality
Use Previews
Import Into Project
Set Start Timecode 00;00;00;00
Render Alpha Channel Only
Time Interpolation: Frame Sampling
Estimated File Size: 33 MB

00;00;00;00
Fit
00;00;25;22
Source Range: Sequence In/Out

Fig. 2.13: Adobe Premiere export setting screen.

Adobe Premiere

- Premiere allows imports of .psd (Photoshop files), which can have alpha channels which can crop specific pixel in the picture.


Effect Controls
Audio Clip Mixer: Background copy/choco-marsh.psd
Metadata
Source: Background copy/choco-marsh.psd
00;00;00;00
Fit
1/2
00;00;04;29

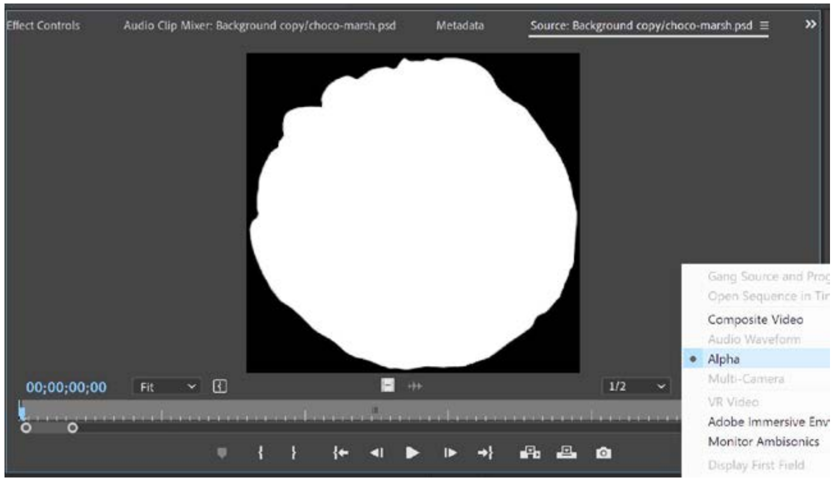


Fig. 2.14: Adobe Premiere clip viewer.

HTML5 Canvas

- HTML5 canvas is an element used to draw animated graphics on a webpage.
- The <canvas> element is a container for graphics.
  - A rectangular area that generated by HTML code with width and height attributes.
  - Using JavaScript, we can actually draw the graphical objects.

Create a Canvas

```
<canvas id="canvas" width="300" height="300" style="background-color: red"> </canvas>

var canvas = document.getElementById('canvas');
var ctx = canvas.getContext('2d');
```

- The background color is transparent by default; here we make it red to be visible.
- The canvas is filled by grids, acting as an x-y coordinate system.

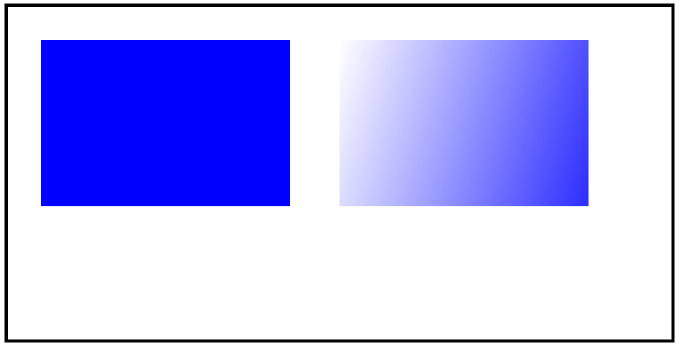
Draw Graphical Objects

- You can draw simple shapes, text, images, path or other objects on the canvas.

```
// a rectangle with filled color
ctx.fillRect(x, y, width, height)

// a rectangle with border
ctx.strokeRect(x, y, width, height)

// clear a rectangle area
ctx.clearRect(x, y, width, height)
```



Transform and Animation

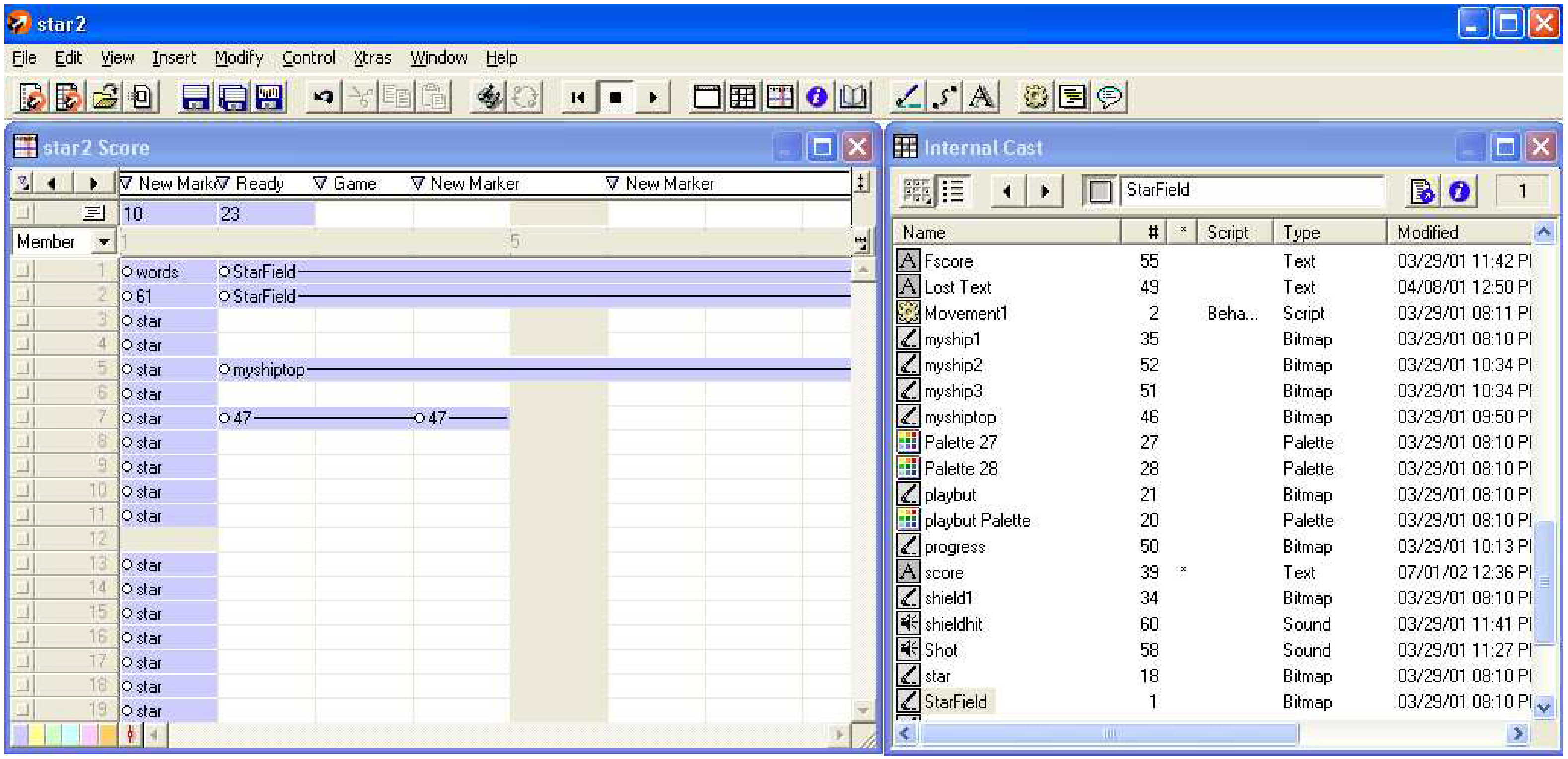
- The shapes of the graphics can be transformed by
  - Translate
  - Rotate
  - Scale

Steps to Create an Animation

1. clear the canvas: `ctx.clearRect(x, y, width, height)`
2. save the current state of canvas: `ctx.save()`
3. draw the graphics
4. restore the canvas state: `ctx.restore()`

Adobe Director

- Director is a complete environment for creating interactive “movies”.



	New Mark	Ready	Game	New Marker		New Marker
	10	23				
Member	1			5		
1	words	StarField				
2	61	StarField				
3	star					
4	star					
5	star	myshiptop				
6	star					
7	star	47	47			
8	star					
9	star					
10	star					
11	star					
12						
13	star					
14	star					
15	star					
16	star					
17	star					
18	star					
19	star					

Name	#	*	Script	Type	Modified
Fscore	55			Text	03/29/01 11:42 PI
Lost Text	49			Text	04/08/01 12:50 PI
Movement1	2		Beha...	Script	03/29/01 08:11 PI
myship1	35			Bitmap	03/29/01 08:10 PI
myship2	52			Bitmap	03/29/01 10:34 PI
myship3	51			Bitmap	03/29/01 10:34 PI
myshiptop	46			Bitmap	03/29/01 09:50 PI
Palette 27	27			Palette	03/29/01 08:10 PI
Palette 28	28			Palette	03/29/01 08:10 PI
playbut	21			Bitmap	03/29/01 08:10 PI
playbut Palette	20			Palette	03/29/01 08:10 PI
progress	50			Bitmap	03/29/01 10:13 PI
score	39	*		Text	07/01/02 12:36 PI
shield1	34			Bitmap	03/29/01 08:10 PI
shieldhit	60			Sound	03/29/01 11:41 PI
Shot	58			Sound	03/29/01 11:27 PI
star	18			Bitmap	03/29/01 08:10 PI
StarField	1			Bitmap	03/29/01 08:10 PI

Director Windows

- Main window
  - Stage, on which the action takes place.
- Cast window
  - Consists of resources a movie may use, such as bitmaps, sounds, vector-graphics shapes, Flash movies, digital videos, and scripts.
- Score window
  - Organized in horizontal lines, each for one of the sprites, and vertical frames.

Animation

- Animation is created by showing slightly different images over time.
  - Using different cast members in different frames.
  - To control this process more easily, Director permits combining many cast members into a single sprite.

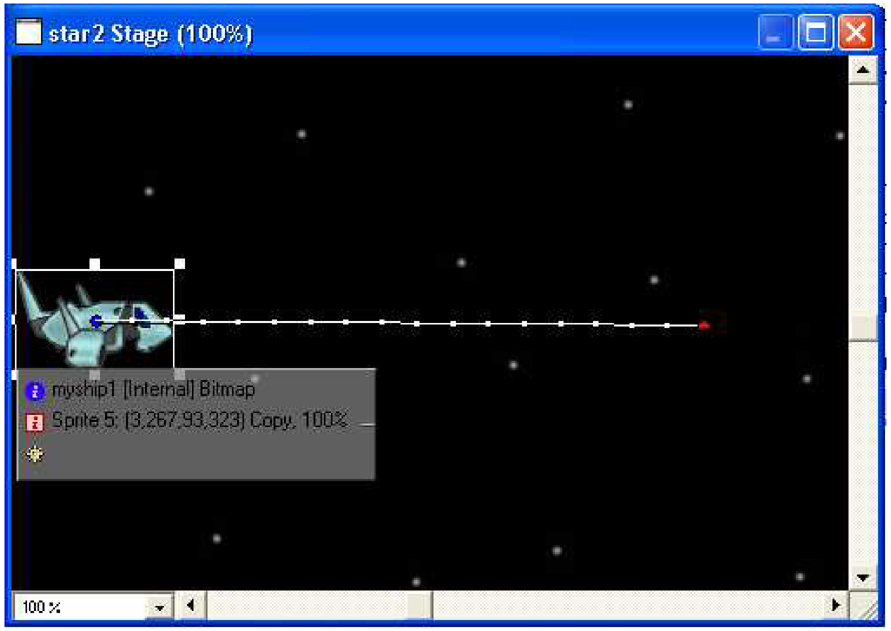

star2 Stage (100%)
myship1 [Internal] Bitmap
Sprite 5: (3,267,93,323) Copy, 100%
100 %

Fig. 2.17: A tweened sprite. Because several instances may be used for a single cast member, each instance is called a sprite. Typically, cast members are raw media, whereas sprites are objects that control where, when, and how cast members appear on the stage and in the movie.

Adobe XD

- Used for making prototypes of websites and mobile apps without any code.

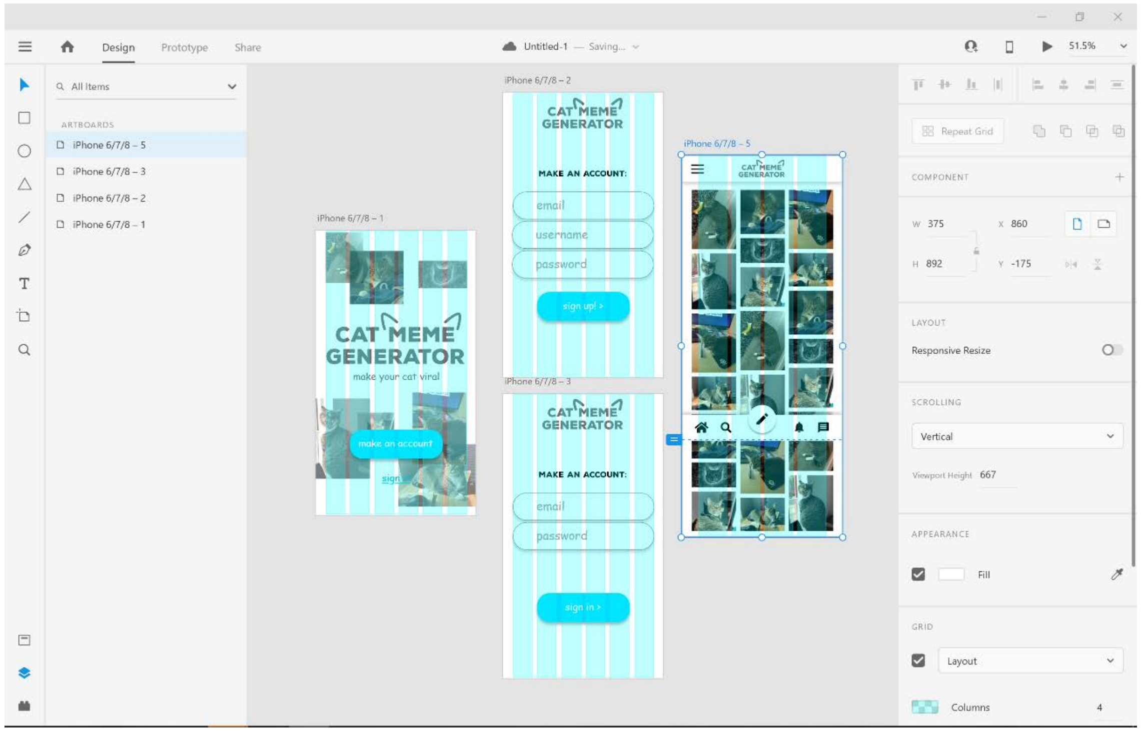
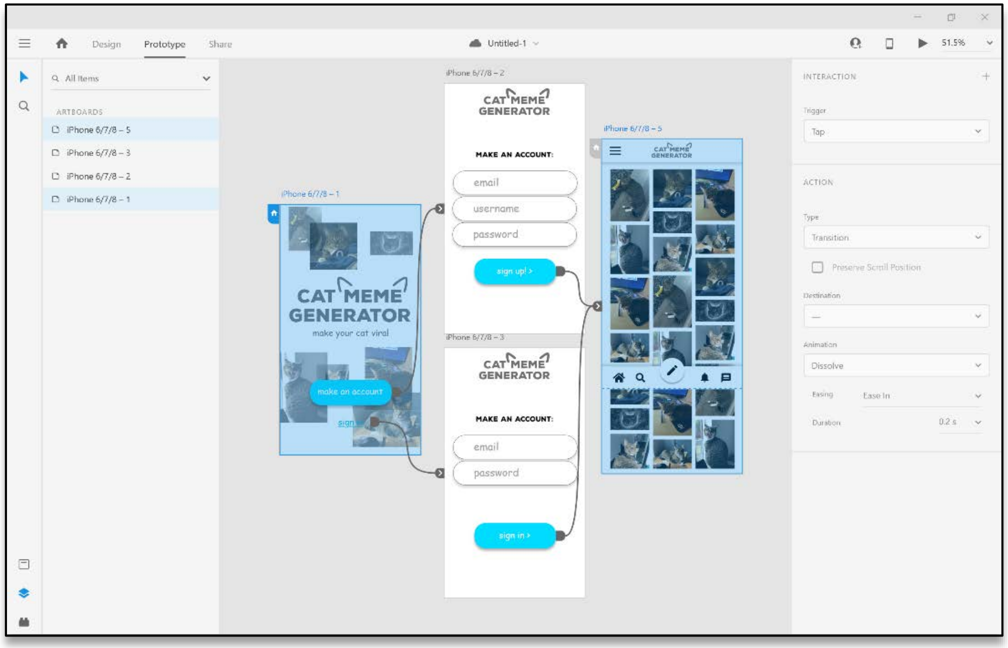


Fig. 2.18: The Design Mode in Adobe XD.

Adobe XD

- Screens are linked by making elements clickable, giving the "feel" of a real app.
- This is commonly used for UI design to be finalized before sent for programming.



INTERACTION

Trigger: Tap

ACTION

Type: Transition

Preserve Scroll Position

Destination: —

Animation: Dissolve

Easing: Ease In

Duration: 0.2 s

Fig. 2.20: The Prototype Mode in Adobe XD.