For this assignment I had to modify my matlab code slightly for the video:

**function** **[**vid**]** **=** video2bin**(**fin**,** fout**)**

% Construct output file name from input

**if** **(**nargin **<** 2**)**

fout **=** **[**fin**(**1**:**max**(**strfind**(**fin**,** '.'**))),** 'bin'**];**

**end**

% Path correction

fin **=** **[**pwd**,**'\'**,**'resources\'**,**fin**];**

fout **=** **[**pwd**,**'\'**,**'output\'**,**fout**];**

fprintf**(**'Input file = %s\n'**,** fin**);**

fprintf**(**'Output file = %s\n'**,** fout**);**

% read the video file

vid **=** VideoReader**(**fin**);**

M **=** vid**.**Height**;**

N **=** vid**.**Width**;**

Fs **=** vid**.**FrameRate**;**

colors **=** vid**.**BitsPerPixel **/** 8**;**

% Write the header

% ndim = 3 (video)

% nchan = colors

% dim0 = M

% dim1 = N

% dim2 = Fs

fid **=** fopen**(**fout**,** 'wb'**);**

fwrite**(**fid**,** **[**3**,** colors**,** M**,** N**,** Fs**],** 'int'**);**

% Loop over pixel(i = row, j = col) -> R, G, B

**while** hasFrame**(**vid**)**

x **=** readFrame**(**vid**);**

**for** i **=** 1**:**M

**for** j **=** 1**:**N

**for** k **=** 1**:**colors

fwrite**(**fid**,** x**(**i**,** j**,** k**),** 'float'**);**

**end**

**end**

**end**

**end**

% Release the file handler

fclose**(**fid**);**

**end**

**function** **[]** **=** bin2video**(**fin**,** fout**)**

% Construct output file name from input

**if** **(**nargin **<** 2**)**

fout **=** **[**fin**(**1**:**max**(**strfind**(**fin**,** '.'**))),** 'mp4'**];**

**end**

% Path correction

fin **=** **[**pwd**,**'\'**,**'output\'**,**fin**];**

fout **=** **[**pwd**,**'\'**,**'output\'**,**fout**];**

fprintf**(**'Input file = %s\n'**,** fin**);**

fprintf**(**'Output file = %s\n'**,** fout**);**

% Read the header

% ndim = 3 (video)

% nchan = colors

% dim0 = M

% dim1 = N

% dim2 = Fs

fid **=** fopen**(**fin**,** 'rb'**);**

ndim **=** fread**(**fid**,** 1**,** 'int'**);**

colors **=** fread**(**fid**,** 1**,** 'int'**);**

M **=** fread**(**fid**,** 1**,** 'int'**);**

N **=** fread**(**fid**,** 1**,** 'int'**);**

Fs **=** fread**(**fid**,** 1**,** 'int'**);**

%Write the data

size\_frame **=** M **\*** N **\*** colors**;**

vid **=** VideoWriter**(**fout**,** 'MPEG-4'**);**

vid**.**FrameRate **=** Fs**;**

open**(**vid**);**

% Matlab is stupid, so we can only do this wacky control structure

**[**a**,** **~]** **=** fread**(**fid**,** size\_frame**,** 'float'**);**

**while** size**(**a**,** 1**)** **~=** 0 % ~=, really?

x **=** zeros**(**M**,** N**,** colors**);**

**for** i **=** 1**:**M

**for** j **=** 1**:**N

**for** k **=** 1**:**colors

% 3 dimensional array access with 1-based indexing

x**(**i**,** j**,** k**)** **=** a**(**...

**(**i **-** 1**)** **\*** N **\*** colors **+** ...

**(**j **-** 1**)** **\*** colors **+** ...

k**);**

**end**

**end**

**end**

% Convert to uint8 just like images

x **=** uint8**(**x**);**

writeVideo**(**vid**,** x**);**

% Matlab is stupid, read the next frame

**[**a**,** **~]** **=** fread**(**fid**,** size\_frame**,** 'float'**);**

**end**

% Release the file handlers

fclose**(**fid**);**

close**(**vid**);**

**end**

C++ Code:

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

**typedef** std**::**vector**<**float**>** dsd**;**

struct dsh **{** int ndim**,** nchan**,** dim0**,** dim1**,** dim2**;** **};**

//--------------------------------------------------------------------------------------+

// Reads a full binary file into reference passed header and vector<float> |

//--------------------------------------------------------------------------------------+

bool getData**(**const std**::**string file**,** dsh**&** header**,** dsd**&** data**)** **{**

float temp**;**

std**::**fstream fin**(**file**,** std**::**ios**::**in **|** std**::**ios**::**binary**);**

**if** **(!**fin**)** **{**

std**::**cout **<<** "Error fetching: " **<<** file **<<** std**::**endl**;**

fin**.**close**();**

**return** **false;**

**}**

fin**.**read**(reinterpret\_cast<**char**\*>(&**header**),** **sizeof(**dsh**));**

**while** **(!**fin**.**eof**())** **{**

fin**.**read**(reinterpret\_cast<**char**\*>(&**temp**),** **sizeof(**float**));**

data**.**push\_back**(**temp**);**

**}**

fin**.**close**();**

**return** **true;**

**}**

void parta**()** **{**

int n**;**

dsh h**;**

dsd d1**,** d2**;**

const std**::**string

f1 **=** "src\\output\\f1.bin"**,**

f2 **=** "src\\output\\f2.bin"**,**

f3 **=** "src\\output\\f3.bin"**;**

// Read f1 and f2

**if** **(!**getData**(**f1**,** h**,** d1**))** **{** **return;** **}**

**if** **(!**getData**(**f2**,** h**,** d2**))** **{** **return;** **}**

// Create a coalesced form

h**.**nchan **=** 2**;**

n **=** d1**.**size**();**

float**\*** d3 **=** **new** float**[**2 **\*** n**];**

**for** **(**int i **=** 0**;** i **<** n**;** **++**i**)** **{**

d3**[(**2 **\*** i**)]** **=** d1**[**i**];**

d3**[(**2 **\*** i**)** **+** 1**]** **=** d2**[**i**];**

**}**

// Write out the coalesced file

std**::**fstream fout**(**f3**,** std**::**ios**::**out **|** std**::**ios**::**binary **|** std**::**ios**::**trunc**);**

fout**.**write**(reinterpret\_cast<**char**\*>(&**h**),** **sizeof(**dsh**));**

fout**.**write**(reinterpret\_cast<**char**\*>(**d3**),** **sizeof(**float**)** **\*** 2 **\*** n**);**

fout**.**close**();**

// Free up dynamically allocated memory

**delete[]** d3**;**

**}**

void partb**()** **{**

int n**;**

dsh h**;**

dsd d1**;**

const std**::**string

f1 **=** "src\\output\\xylophone.bin"**,**

f2 **=** "src\\output\\xylophone\_gray.bin"**;**

// Read f1

**if** **(!**getData**(**f1**,** h**,** d1**))** **{** **return;** **}**

// Create a grayscale version

h**.**nchan **=** 1**;**

n **=** d1**.**size**()** **/** 3**;**

float**\*** d2 **=** **new** float**[**n**];**

**for** **(**int i **=** 0**;** i **<** n**;** **++**i**)** **{**

d2**[**i**]** **=**

**(**0.2989 **\*** d1**[(**3 **\*** i**)** **+** 0**])** **+** // Red

**(**0.5870 **\*** d1**[(**3 **\*** i**)** **+** 1**])** **+** // Green

**(**0.1140 **\*** d1**[(**3 **\*** i**)** **+** 2**]);** // Blue

**}**

// Write out f2

std**::**fstream fout**(**f2**,** std**::**ios**::**out **|** std**::**ios**::**binary **|** std**::**ios**::**trunc**);**

fout**.**write**(reinterpret\_cast<**char**\*>(&**h**),** **sizeof(**dsh**));**

fout**.**write**(reinterpret\_cast<**char**\*>(**d2**),** **sizeof(**float**)** **\*** n**);**

fout**.**close**();**

// Free up dynamically allocated memory

**delete[]** d2**;**

**}**

int main**()** **{**

parta();

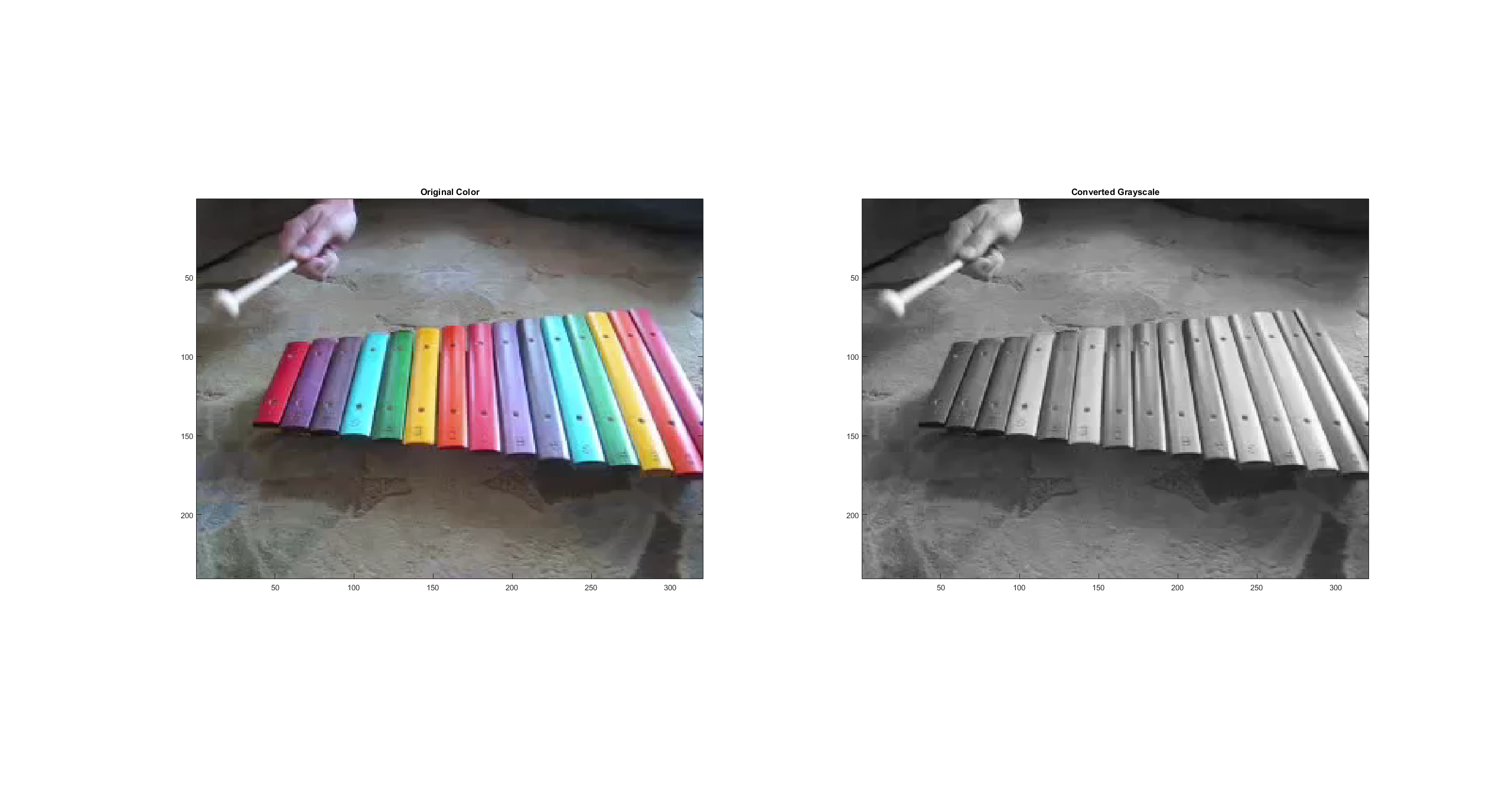
partb**();**

system**(**"pause"**);**

**return** 0**;**

**}**

Side by side comparison of the first frame for the video:



Code to generate this figure:

clear all**;**

color **=** VideoReader**(**'xylophone.mp4'**);**

gray **=** VideoReader**(**'xylophone\_gray.mp4'**);**

x **=** readFrame**(**color**);**

y **=** readFrame**(**gray**);**

figure**;**

subplot**(**1**,** 2**,** 1**);**

imagesc**(**x**);**

axis image**;**

title**(**'Original Color'**);**

subplot**(**1**,** 2**,** 2**);**

imagesc**(**y**);**

axis image**;**

title**(**'Converted Grayscale'**);**