

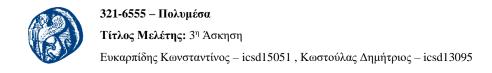
321 - 6555 Πολυμέσα

Διδάσκων: Καρύμπαλη Ειρήνη

3^η Askhoh

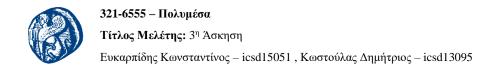
Εργαστηριακοί Συνεργάτες: Καρύμπαλη Ειρήνη

Ευκαρπίδης Κωνσταντίνος icsd15051 Κωστούλας Δημήτριος icsd13095



Περιεχόμενα

1	METATPOITH RGB VIDEO ΣΕ GRAYSCALE	. 3
2	DPCM ΚΩΔΙΚΟΠΟΊΗΣΗ	. 4
3	Епаnaahuh DPCM гia тмнма тоу video	. 5
4	APXEIO "WORKSPACE,MAT"	. 6



1 Μετατροπή RGB video σε grayscale

Στο 1° ερώτημα της $3^{\eta\varsigma}$ άσκησης μας ζητείται να μετατρέψουμε το δοθέν video από έγχρωμο σε grayscale. Ω ς παραμέτρους περνάμε το μονοπάτι του βίντεο και επιστρέφουμε το αλλαγμένο βίντεο.

```
function [a] = convert( path )
%πέρασμα αρχείου στον buffer
v=VideoReader(path);
%διάβασμα αρχείου απο τον buffer
video = read(v);
%αναπαραγωγή video πριν την επεξεργασία
implay(video);
[rows,columns,~,frames] = size(video);
a = zeros(rows,columns,frames);
 for i = 1:frames
     %Μετατροπή κάθε frame σε grayscale
     a(:,:,i) = rgb2gray(video(:,:,:,i));
 end
 %αναπαραγωγή μετά την μετατροπή σε grayscale
 implay(uint8(a));
end
```

Εικόνα 1.1 : Συνάρτηση μετατροπής

Όπως βλέπουμε αρχικά με την VideoReader συνάρτηση , η οποία αποτελεί μία stream συνάρτηση, αποθηκεύουμε στον buffer την ροή. Έπειτα την διαβάζουμε απο αυτόν σε μία μεταβλητή ώστε να είναι διαχειρίσημη και τρέχουμε μία επανάληψη για όσα frame έχει το βιντεο. Στην επανάληψη γίνεται μετατροπή του κάθε frame σε grayscale . Αναπαράγουμε το βίντεο πριν και μετά την μετατροπή.

2 DPCM κωδικοποίηση

Η DPCM κωδικοποίηση αφορά την διαφορά των τιμών κάθε στοιχείου απο το προηγούμενο. Στην δικιά μας περίπτωση έχουμε τα frames , τα οποία αποετελούν vectors. Συγκρίνουμε δηλαδή κάθε στοιχείο του vector με το αντίστοιχο στο προηγούμενο frame και κρατάμε την διαφορά τους. Έτσι ξεκινάμε αφήνωντας ανέπαφο τον 1° frame. Ως παραμέτρους παίρνει το grayscale video και επιστρέφει το λόγο συμπίεσης του.

```
\underline{\text{function}} [ 1 ] = pcm(a)
    %αρχικοποιηση πινακων
 [rows,columns,frames] = size(a);
 b = zeros(rows, columns, frames);
 c = zeros(rows,columns);
 %περασμα lou frame στον καινουριο πινακα
 b(:,:,1) = a(:,:,1);
for i = 2 : frames
    %υπολογισμός διαφορών μεταξύ διαδοχικων πλαισιων
    b(:,:,i) = a(:,:,i) - a(:,:,i-1);
end
for i = 1 : rows
    for j = 1 : columns
         %υπολογισμος bits κβαντισμού για τις διαφορές/pixel
         c(i,j) = ceil(log2((max(b(i,j,2:frames)) - min(b(i,j,2:frames)) + 1)));
    end
end
%λόγος συμπιεσης
1 = (rows*columns*8*frames) / (rows*columns*8 + (frames-1)*sum(c(:)));
end
```

Εικόνα 2.1 : DPCM κωδικοποίηση

Αρχικά δημιουργούμε 2 πίνακες. Ο πρώτος (b) είναι για να κρατάει τις διαφορές και ο δέυτερος είναι για να υπολογίζουμε το πλήθος bytes που χρειαζόμαστε (πάνω όριο του λογαρίθμου με βάση το 2 της διαφοράς max — min +1). Τέλος υπολογίζουμε το λόγο συμπίεσης για να δούμε το όφελος αυτής της κωδικοποίησης.

3 Επανάληψη DPCM για τμήμα του video

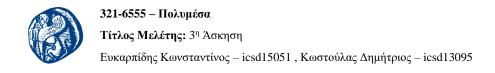
Στο τρίτο ερώτημα χρησιμοποιούμε την συνάρτηση του προηγούμενου ερωτήματος αλλά όχι για όλα τα frames του βίντεο. Το σπάμε σε κομμάτια των 30 frames. Αυτό που παρατηρούμε είναι οτι οι λόγοι συμπίεσεις είναι αυξημένοι σε σχέση με ολόκληρο το video. Το γεγονός αυτό παρατηρείται γιατί ο αριθμός των bits θα είναι πολύ μικρότερος αφού έχουμε εισάγει ένα μικρό μέρος του βιντεο και όχι ολόκληρο.

```
logos_1_30 = pcm(gray_video(:,:,1:30));
logos_31_60 = pcm(gray_video(:,:,31:60));
logos_61_90 = pcm(gray_video(:,:,61:90));
logos_91_120 = pcm(gray_video(:,:,91:120));
logos_121_145 = pcm(gray_video(:,:,121:145));
```

Εικόνα 3.1 : Κλήση συνάρτησης για μέρος των συνολικών frames

```
logos_121_145 3.3747
logos_1_30 2.2779
logos_31_60 2.0069
logos_61_90 2.8499
logos_91_120 3.5459
logos_all_frames 1.8733
```

Εικόνα 3.2 : Σύγκριση λόγων συμπίεσης



4 Αρχείο "workspace.mat"

Σαυτό το workspace έχουμε αποθηκεύσει ,όπως μας ζητήθηκε τα παρακάτω δεδομένα.

- Gray_video : Επιστρεφόμενο αποτέλεσμα από την συνάρτηση μετατροπής
- Logos_n_m : Οι λόγοι για κάθε τμήμα των συνολικών frames
- Logos_all_frames : Λόγος συμπίεσης του grayscale video (2° ερώτημα)