

COMPUTER VISION LAB EXERCISE 4

ΚΩΝΣΤΑΝΤΙΝΟΣ ΚΑΡΑΪΣΚΟΣ
AM: 1072636

February 27, 2024

Ερώτημα 1

Η συνάρτηση παίρνει ως είσοδο, δύο εικόνες εκ των οποίων η μία αποτελεί την εικόνα βάσης και η άλλη την γεωμετρικά παραμορφωμένη εικόνα την οποία προσπαθούμε να ευθυγραμμίσουμε ως προς την εικόνα βάσης. Ακόμα δίνουμε ως ορίσματα τον αριθμό επιπέδων της Gaussian πυραμίδας που θα θέλαμε να δημιουργήσουμε, τον αριθμό των φορών που θέλουμε να επαναληφθεί ο ecc ή ο lk, ανά επίπεδο, το είδος του γεωμετρικού μετασχηματισμού που ψάχνουμε και τέλος το αρχικό μητρώο γεωμετρικής παραμόρφωσης των αρχικών εικόνων κάθε επιπέδου.

Καλούμε τον αλγόριθμο `ecc_lk_alignment` για την εικόνα `lena_color.tiff`, αφού πρώτα την φέρουμε σε grayscale και εφαρμόζουμε έναν γεωμετρικό μετασχηματισμό πάνω της.

```
img=rgb2gray(imread("lena_color.tiff"));
sh_y=0.1;
sh_x=0.1;
A=[1 sh_x 0;
   sh_y 1 0;
   0 0 1];

tform=affine2d(A);
wImg=imwarp(img,tform,'cubic','FillValues',1);

[results results_lk MSE,rho,MSELK]= ecc_lk_alignment...
(wImg,img,1,2,'affine',eye(2,3));
```

Αποτελέσματα ECC:

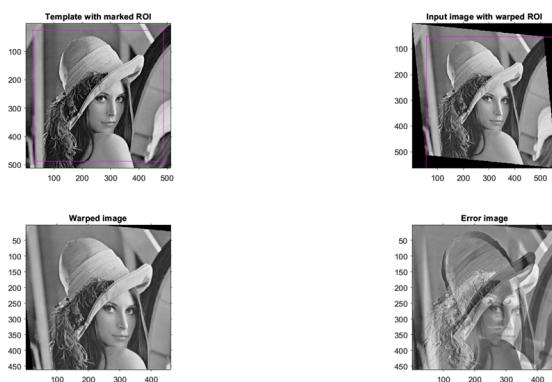


Figure 1: ECC output

Αποτελέσματα LC:

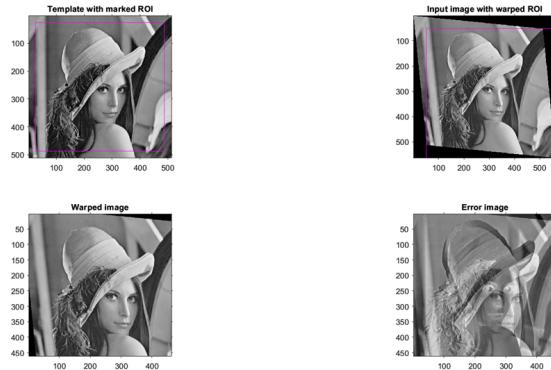


Figure 2: LK output

Και για τον ECC και τον LK, γίνεται plot το template με την περιοχή ενδιαφέροντος, η γεωμετρικά παραμορφωμένη εικόνα με την αντίστοιχη περιοχή ενδιαφέροντος, η παραμορφωμένη εικόνα και η διαφορά του template με την image. Για τα συγκεκριμένα ορίσματα, δεν μπορούμε να παρατηρήσουμε κάποια ορατή διαφορά μεταξύ των δύο αλγορίθμων.

Αποτελέσματα PSNR:

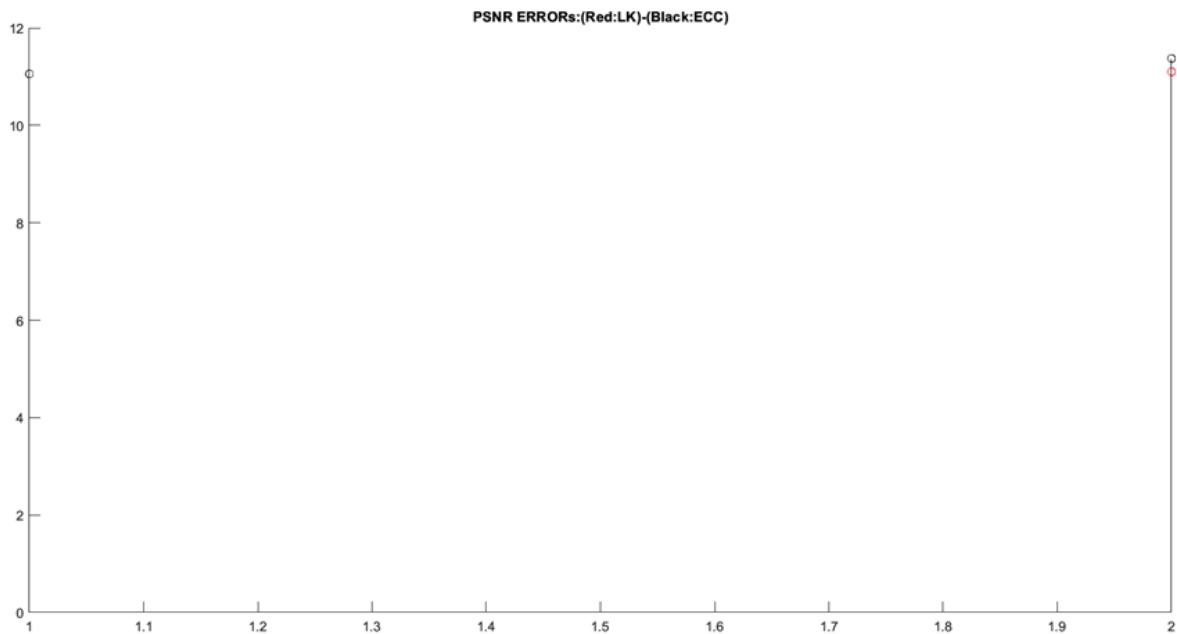


Figure 3: ECC-LK PSNR

Είναι το PSNR ανάλογα με τον αριθμό της επανάληψης για κάθε αλγόριθμο. Βλέπουμε ότι, στην δεύτερη επανάληψη που φαίνεται και για τους δύο αλγορίθμους, ο ECC έχει μικρότερο PSNR από ότι ο LK.

Ερώτημα 2

- *spatial_interp*: Υλοποιεί μια δισδιάστατη χωρική παρεμβολή στην εικόνα εισόδου IN. Οι συντεταγμένες NX, NY προβάλλονται μέσω της warp και προκύπτει ένα νέο subpixel σύστημα συντεταγμένων, του οποίου οι τιμές των subpixel υπολογίζονται μέσω διγραμμικής παρεμβολής τις εικόνα εισόδου IN.
- *image_jacobian*: Η συνάρτηση αυτή υπολογίζει το Jacobian μητρώο G της γεωμετρικά παραμορφωμένης εικόνα με βάση τις wrk παραμέτρους.
- *warp_jacobian*: Η συνάρτηση αυτή υπολογίζει την Jacobian J ως προς τις παραμέτρους της γεωμετρικής παραμόρφωσης warp. Αν η γεωμετρική παραμόρφωση είναι homography ή Euclidean η Jacobian J εξαρτάται από τις τιμές των παραμέτρων, αν είναι affine ή translation όχι.
- *param_update*: Η συνάρτηση αυτή ενημερώνει τις παραμέτρους του γεωμετρικού μετασχηματισμού, προσθέτοντας τις τιμές συσχέτισης του DELTA_P στο WARP_IN.

Ερώτημα 3

Παρακάτω βλέπουμε τους χρόνους εκτέλεσης για τα 4 βίντεο. Παρατηρούμε ότι για τα βίντεο υψηλής ανάλυσης ο χρόνος εκτέλεσης είναι μεγαλύτερος. Αυτό γιατί τα βίντεο υψηλής ανάλυσης είναι 256x256 ενώ τα βίντεο χαμηλής ανάλυσης είναι 64x64, άρα ο αλγόριθμος χρειάζεται περισσότερο χρόνο για τα πρώτα.

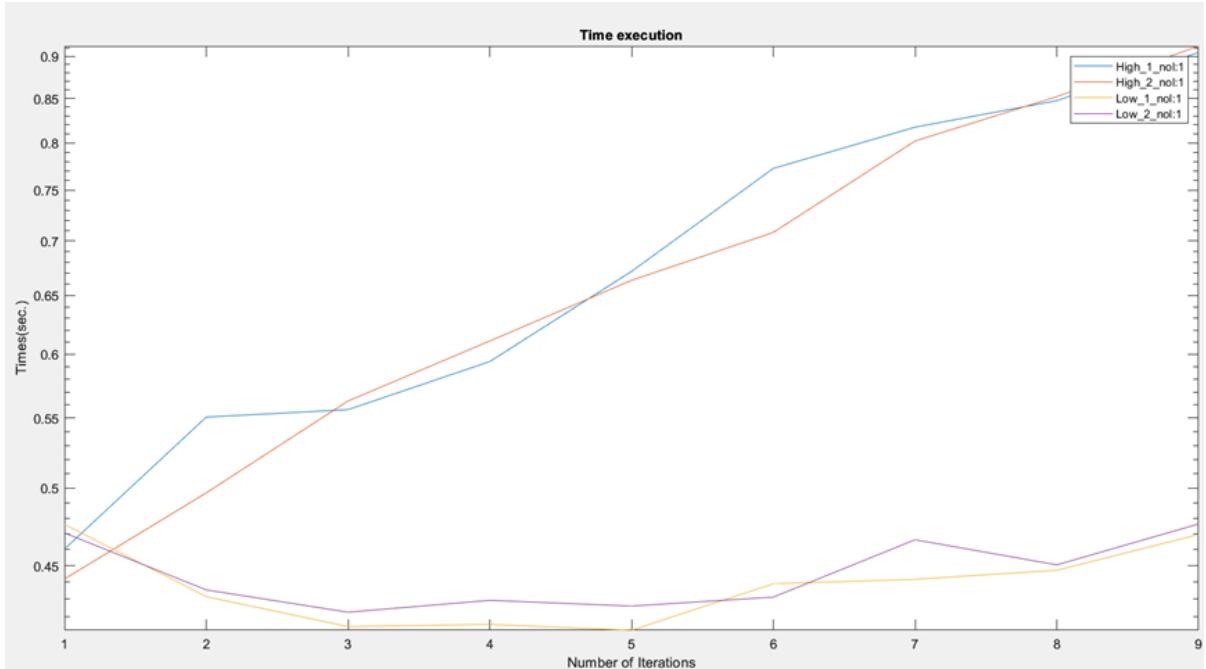


Figure 4: Iterations vs Time Graph

Παρατηρούμε ότι το PSNR στον ECC αυξάνεται αρκετά όσο ανεβαίνει ο αριθμός των επαναλήψεων, δηλαδή αυξάνεται και η απόδοση του αλγορίθμου. Αυτό φαίνεται και στα βίντεο χαμηλής ανάλυσης αλλά εμφανέστερα στα υψηλής ανάλυσης. Από τα γραφήματα παρατηρούμε ότι ο ECC έχει καλύτερη απόδοση από τον LK.

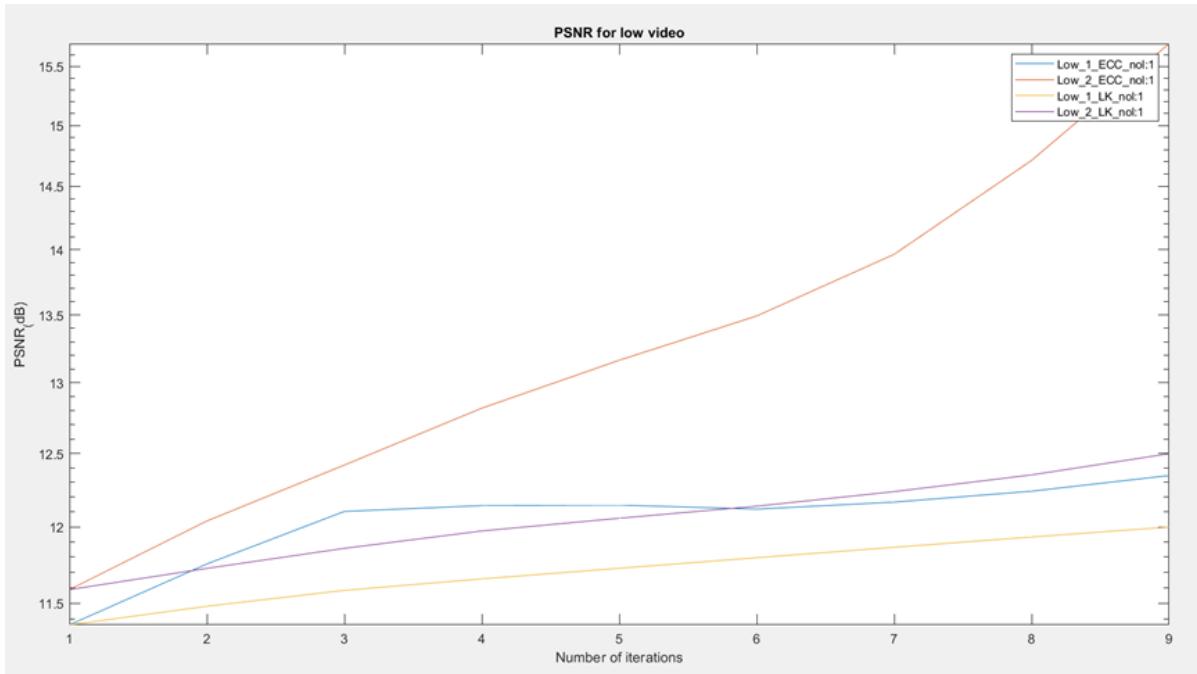


Figure 5: PSNR for Low Video

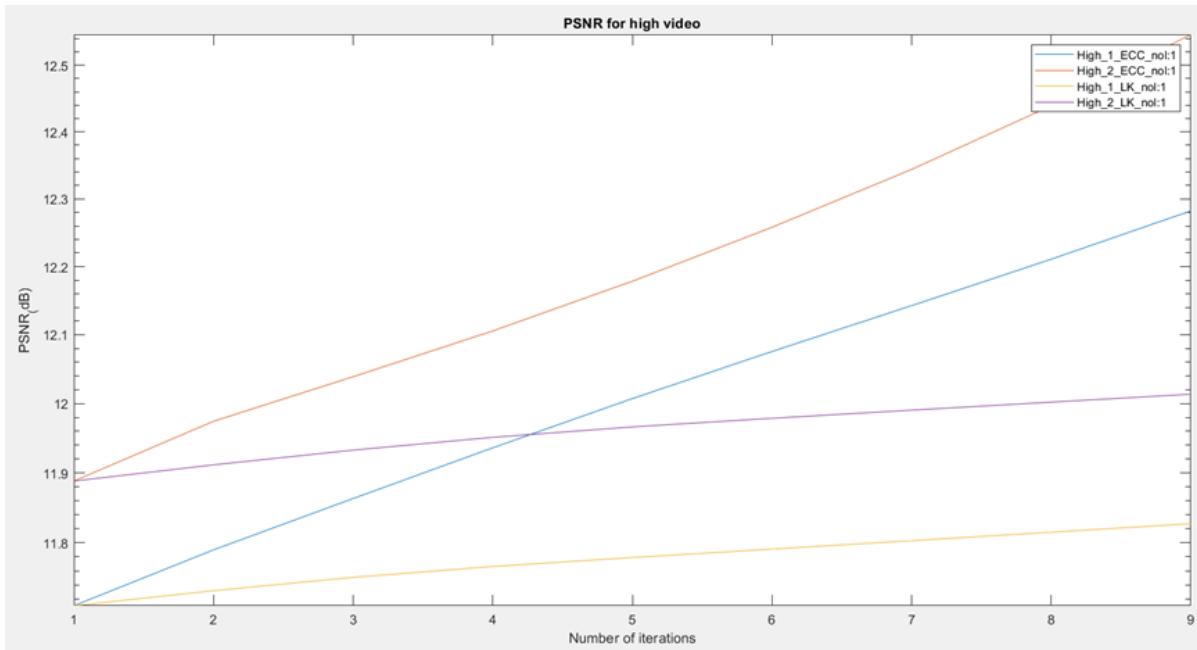


Figure 6: PSNR for High Video

Ερώτημα 4

Προκειμένου να εξετάσουμε την απόδοση των αλγορίθμων θα τρέξουμε τον αλγόριθμο ανά ζεύγη frames για τα βίντεο υψηλής και χαμηλής ανάλυσης. Θα ορίσουμε το img να είναι 10 frame μετά από το frame που ορίζουμε ως temp, γιατί καθώς θα ορίσουμε 1 επίπεδο πυραμίδας, η μετατόπιση μεταξύ διαδοχικών frames δεν είναι αρκετή ώστε να παρατηρηθεί διαφορά στην απόδοση μεταξύ του ECC και LK, καθώς οι γραφικές παραστάσεις των PSNR για τα βίντεο ταυτίζονται.

Παρακάτω δίνουμε τις γραφικές παραστάσεις των PSNR των βίντεο:

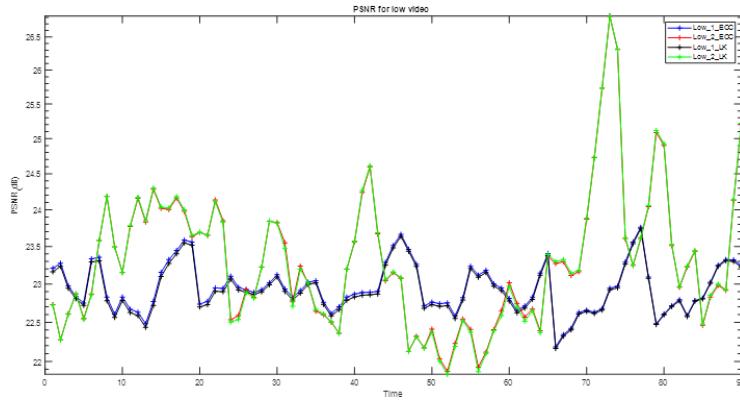


Figure 7: PSNR for Low Quality Videos - 1 level Pyramid

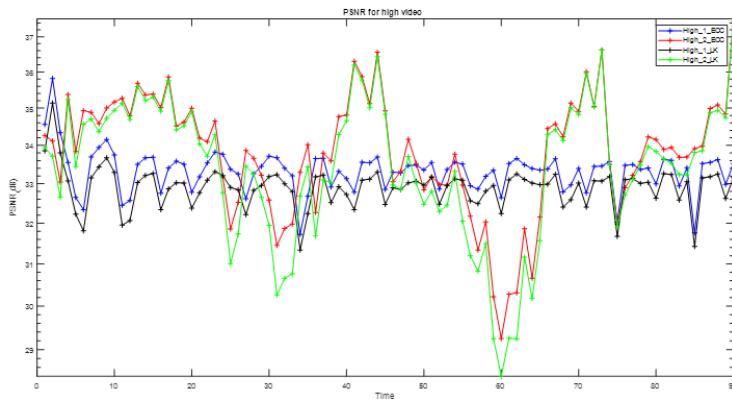


Figure 8: PSNR for High Quality Videos - 1 level Pyramid

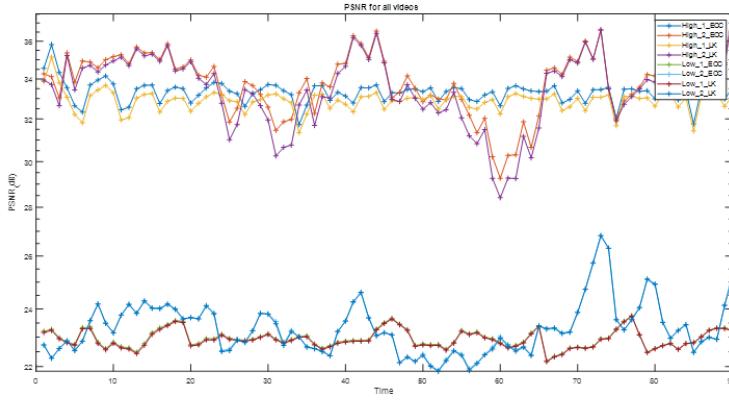


Figure 9: PSNR for all Videos - 1 level Pyramid

Με μεγαλύτερη ευχρίνεια στις γραφικές παραστάσεις για τα υψηλής ευχρίνειας βίντεο, βλέπουμε ότι το PSNR του ECC είναι μεγαλύτερο από ότι του LK και για τα δύο βίντεο, καθώς και για το βίντεο με PSNR τη μαύρη και μπλε γραμμή βλέπουμε ότι έχουμε μικρότερες διακυμάνσεις στην τιμή του PSNR σε σχέση με ο άλλο βίντεο. Γενικά, όσο μεγαλύτερη είναι η τιμή του PSNR, τόσο καλύτερη είναι και η ευθυγράμμιση καθώς έχουμε μικρότερο MSE. Στη γενική περίπτωση παρατηρούμε ότι ο ECC αποδίδει καλύτερα σε σχέση με τον LK.

Για 2 επίπεδα πυραμίδας, βλέπουμε ότι ο ECC δίνει εμφανώς πολύ καλύτερα αποτελέσματα σε σχέση με τον LK.

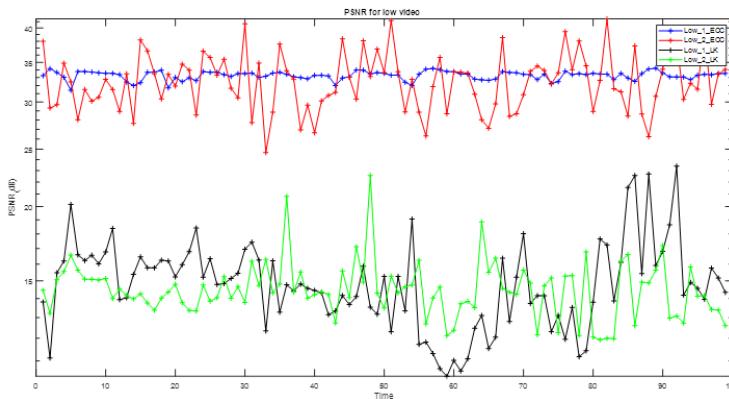


Figure 10: PSNR for Low Quality Videos - 2 levels Pyramid

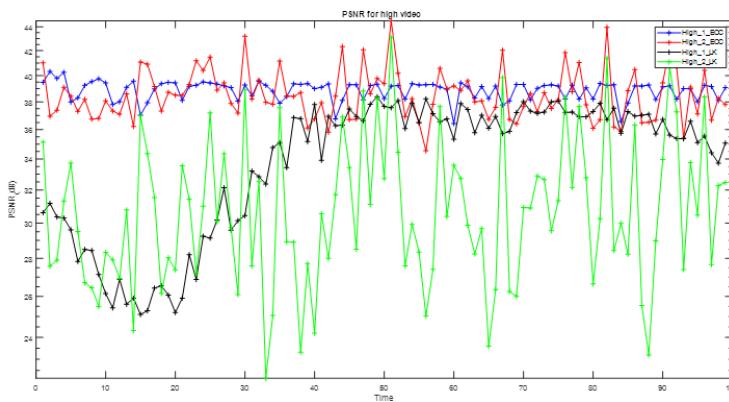


Figure 11: PSNR for High Quality Videos - 2 levels Pyramid

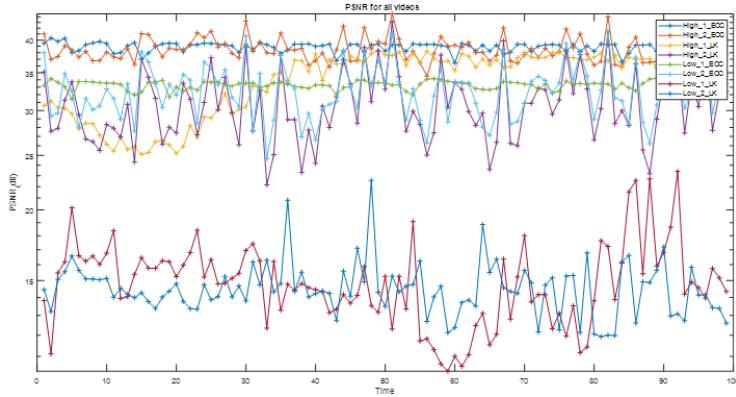


Figure 12: PSNR for all Videos - 2 levels Pyramid

Ερώτημα 5

Για να εξετάσουμε την απόδοση των αλγορίθμων ευθυγράμμισης παρουσία φωτομετρικών παραμορφώσεων, ακολουθούμε την ίδια διαδικασία με το προηγούμενο ερώτημα, και επιπλέον εισάγουμε στην εικόνα εισόδου temp, img και τέλος και στις δύο μαζί τυχαίο ομοιόμορφο θόρυβο που παίζει το ρόλο της φωτομετρικής παραμόρφωσης.

Παρακάτω φαίνονται οι γραφικές παραστάσεις για τα PSNR των βίντεο, με φωτομετρικά παραμορφωμένη την εικόνα εισόδου temp.

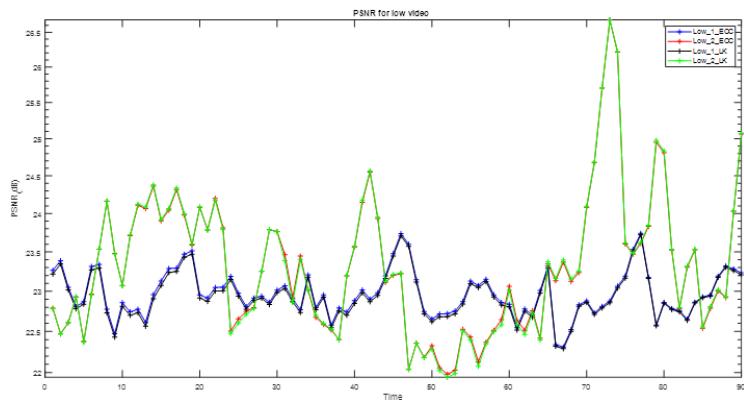


Figure 13: PSNR for Low Quality Videos with photometric distortion on temp image

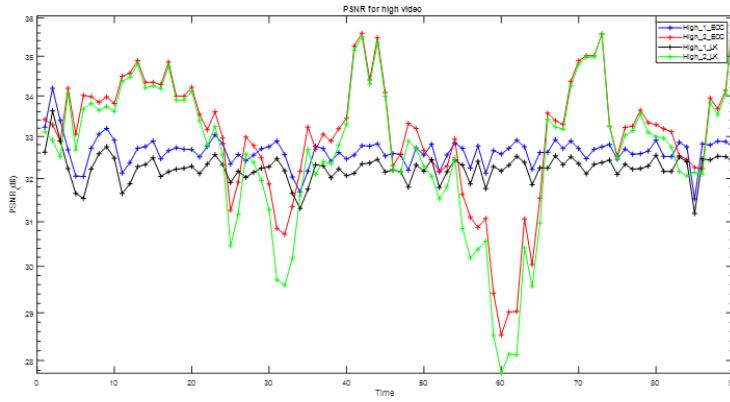


Figure 14: PSNR for High Quality Videos with photometric distortion on temp image

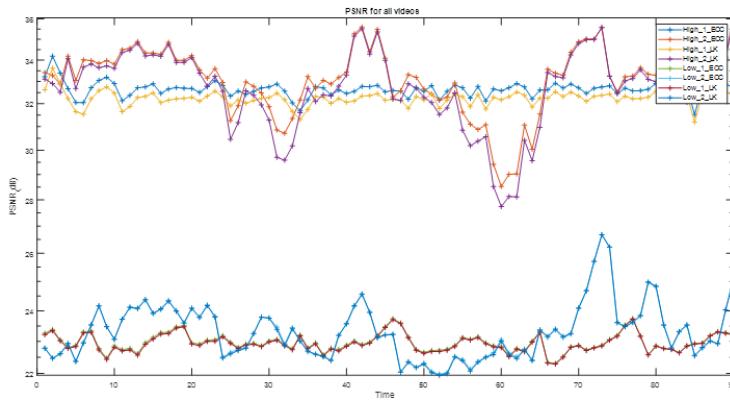


Figure 15: PSNR for All Videos with photometric distortion on temp image

Παρακάτω φαίνονται οι γραφικές παραστάσεις για τα PSNR των βίντεο, με φωτομετρικά παραμορφωμένη την εικόνα εισόδου img.

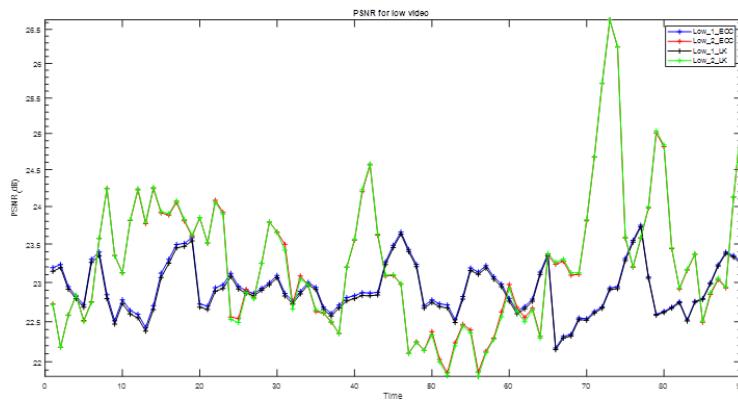


Figure 16: PSNR for Low Quality Videos with photometric distortion on img

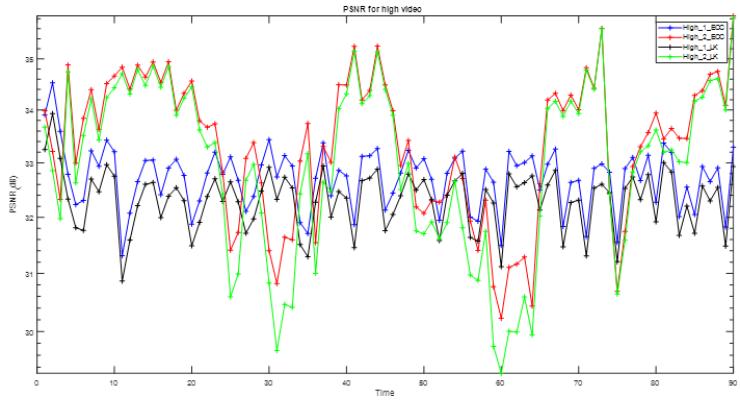


Figure 17: PSNR for High Quality Videos with photometric distortion on img

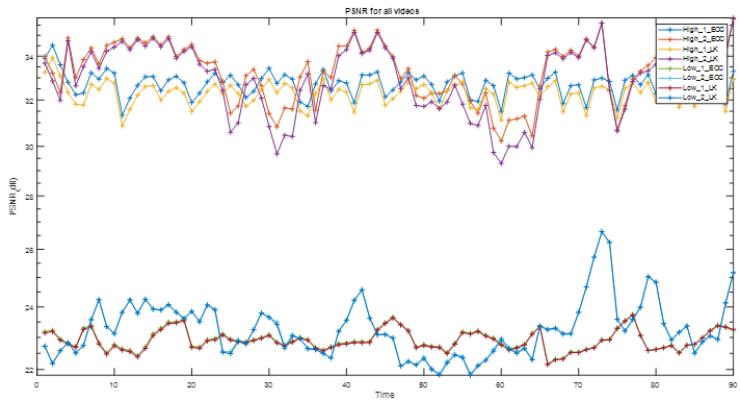


Figure 18: PSNR for All Videos with photometric distortion on img

Παρακάτω φαίνονται οι γραφικές παραστάσεις για τα PSNR των βίντεο, με φωτομετρικά παραμορφωμένες και τις δύο εικόνες εισόδου.

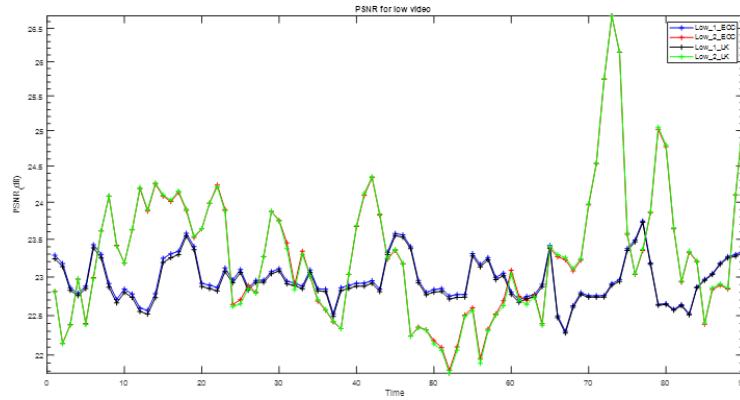


Figure 19: PSNR for Low Quality Videos with photometric distortion on both inputs

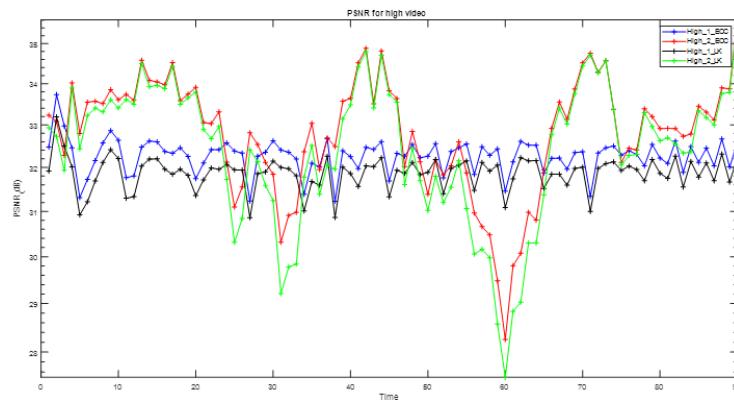


Figure 20: PSNR for High Quality Videos with photometric distortion on both inputs

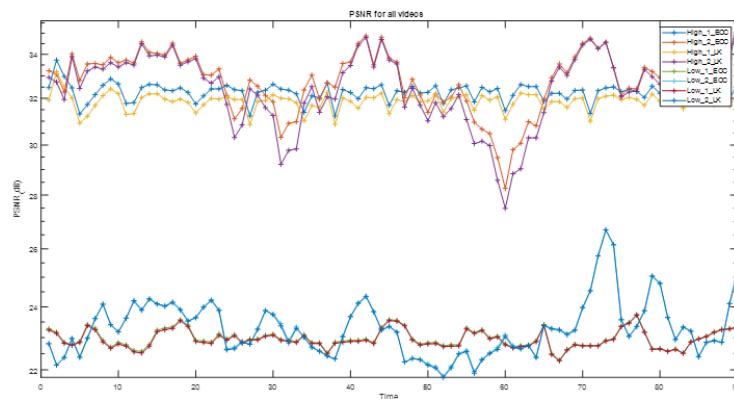
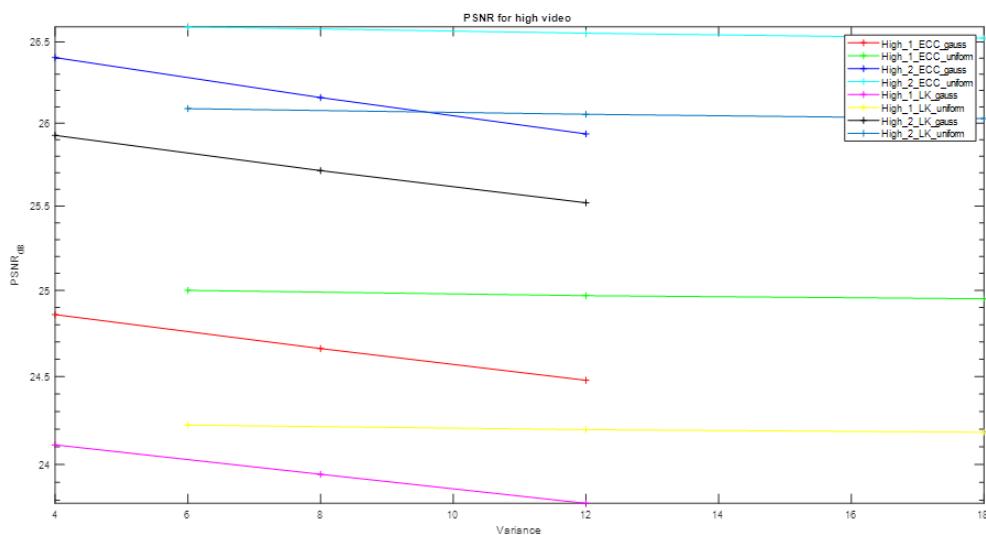
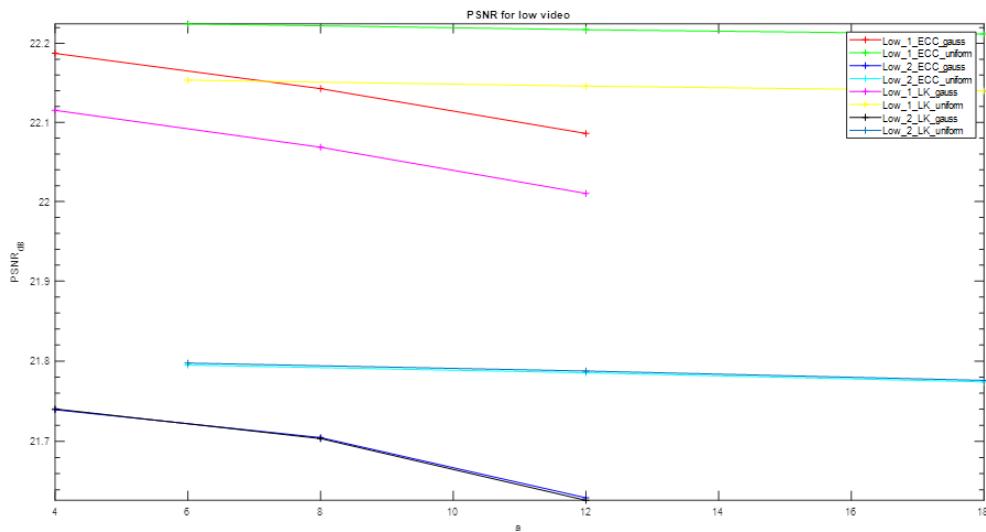


Figure 21: PSNR for All Videos with photometric distortion on both inputs

Όπως φαίνεται και από τις γραφικές παραστάσεις, ο ECC έχει καλύτερη απόδοση σε κάθε περίπτωση φωτομετρικής παραμόρφωσης που εισάγουμε σε κάποια ή και στις δύο εικόνες εισόδου. Συγκριτικά με το προηγούμενο ερώτημα, μπορούμε να δούμε ότι η απόδοση των αλγορίθμων δεν μεταβάλλεται σημαντικά, κάτιο το οποίο μας οδηγεί στο συμπέρασμα ότι οι αλγόριθμοι είναι αρκετά εύρωστοι ως προς τις φωτομετρικές παραμορφώσεις.

Ερώτημα 6

Από τις γραφικές παραστάσεις βλέπουμε ότι οι δύο αλγόριθμοι δεν έχουν και πολύ μεγάλη διαφορά ως προς το PSNR. Στα βίντεο χαμηλής ανάλυσης η διαφορά στο PSNR είναι κατά μέγιστο 0.5 ενώ στα βίντεο υψηλής ανάλυσης είναι περίπου 2.5. Θα περιμέναμε ίσως ο ECC να αποδίδει καλύτερα παρουσία Gaussian και Uniform θορύβου, με βάση και τις παρατηρήσεις μας από τα προηγούμενα ερωτήματα ωλά δεν μπορούμε να το εξακριβώσουμε. (Έτρεξα το πείραμα 100 φορές με noi=5 για κάθε βίντεο, γιατί με μεγαλύτερους αριθμούς από αυτά κράσαρα).



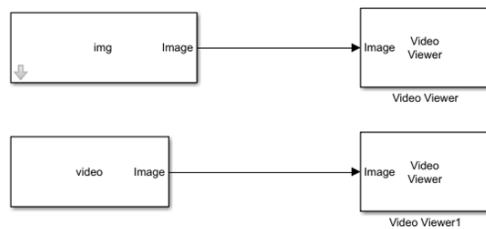
Mέρος Β

Ερώτημα B2

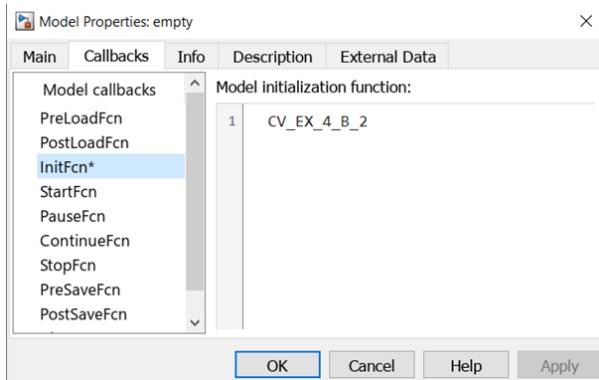
Ανοίγουμε την εικόνα και δημιουργούμε ένα βίντεο των 2 frames.

```
% Question B_2
clear all;
load('img.mat');
img = uint8(img);
frames = 2;
% Struct containing all the frames
video(frames) = struct('cdata',[],'colormap',[]);
% insert frames to img1
video(1) = im2frame(img,gray(256));
video(2) = im2frame(img,gray(256));
```

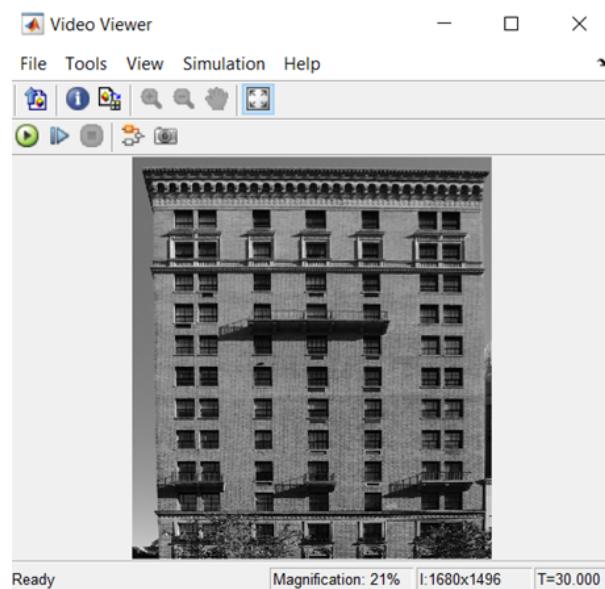
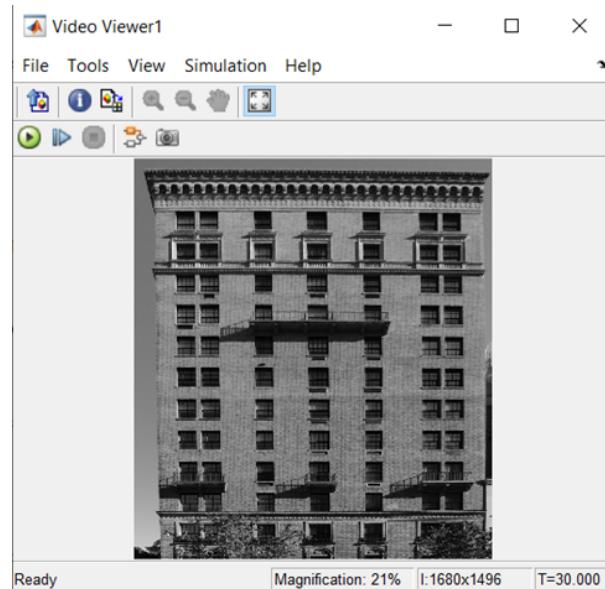
Ανοίγουμε το αρχείο *empty.mdl* και δημιουργούμε τα ζητούμε συστήματα.



Στο model properties, ορίζουμε πριν την εκτέλεση του μοντέλου να εκτελεστεί ο κώδικας σε Matlab που θα αρχικοποιήσει τις μεταβλητές για τα μοντέλα:



Εξοδοι μοντέλου:

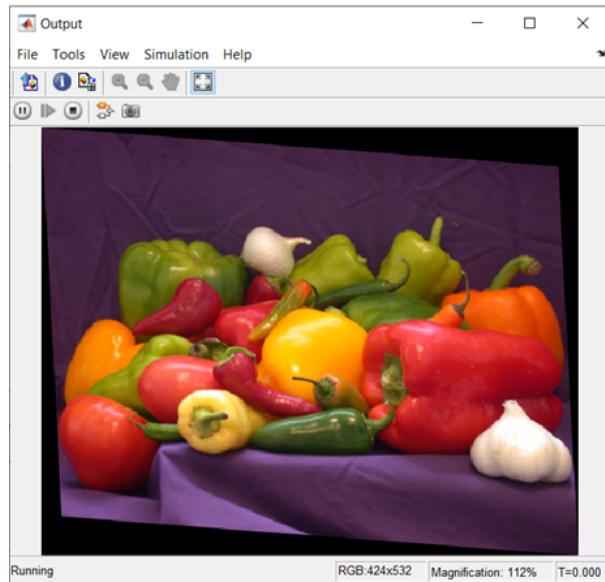


Ερώτημα Β3

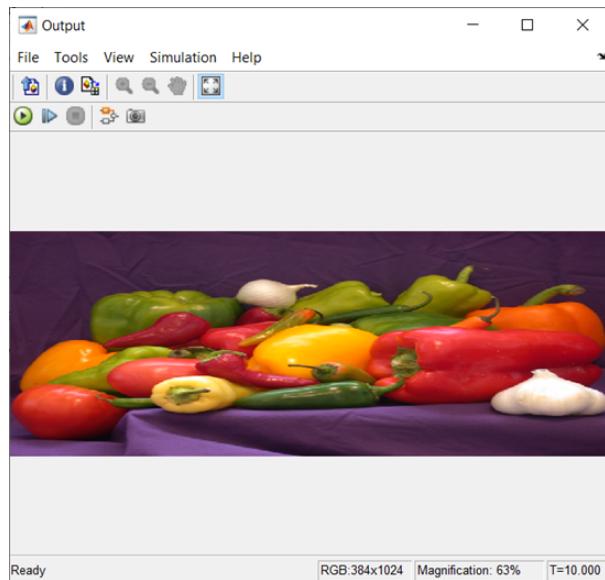
1. Έστω μία μικρή περιοχή δS στο S . Με τον μετασχηματισμό συγγένειας W , η περιοχή μετατρέπεται σε $\delta S' = W(S)$. Ο μετασχηματισμός κλιμακώνει την περιοχή κατά $|det(A)|$, άρα έχουμε $\delta S' = |det(A)| * \delta S$. Ολοκληρώνοντας για όλη την περιοχή S , παίρνουμε το εμβαδόν της. Άρα, η περιοχή $W(S)$, υπολογίζεται από την ολοκλήρωση $\delta S'$ στο $W(S)$, το οποίο ισούται με $|det(A)|$ φορές το ολοκλήρωμα του δS στο S . Άρα, $W(S) = |det(A)| * (\text{εμβαδόν } S)$.

Η ορίζουσα του A , αποτελεί γεωμετρικά την κλιμάκωση μίας περιοχής συμπεριλαμβανομένης και της αλλαγής στον προσανατολισμό. Αν $det(A) > 0$ τότε ο αρχικός προσανατολισμός διατηρείται, αλλιώς αν $det(A) < 0$ ο προσανατολισμός αντιστρέφεται. Η απόλυτη τιμή $|det(A)|$ μας δίνει τον παράγοντα κλιμάκωσης με βάση τον οποίο μία περιοχή μεγεθύνεται ή σμικρύνεται, αγνοώντας μεταβολές στον προσανατολισμό.

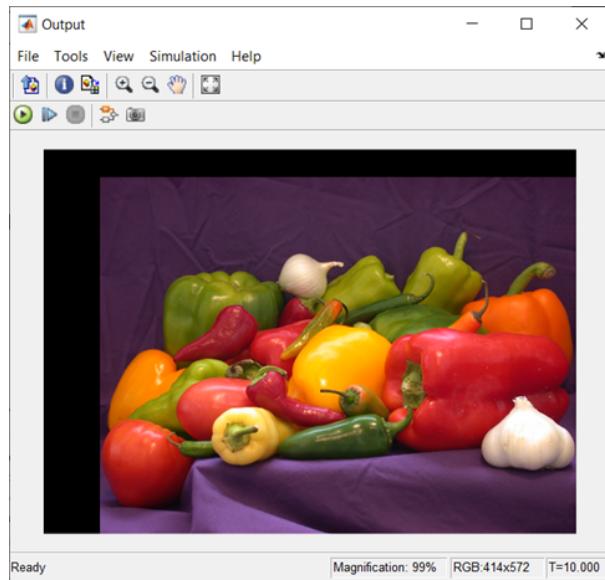
2. Για να πετύχουμε οριζόντια στρέβλωση κατά 20 pixels, θέτουμε την είσοδο $S = [-10 \ 10]$, η οποία είναι οι τιμές στρέβλωσης γραμμής και στήλης και την Maximum shear value = 20, που ορίζει τη μέγιστη στρέβλωση. Για να πετύχουμε κατακόρυφη στρέβλωση κατά 40 pixels, θέτουμε την είσοδο $S = [-20 \ 20]$ και την Maximum shear value = 40.



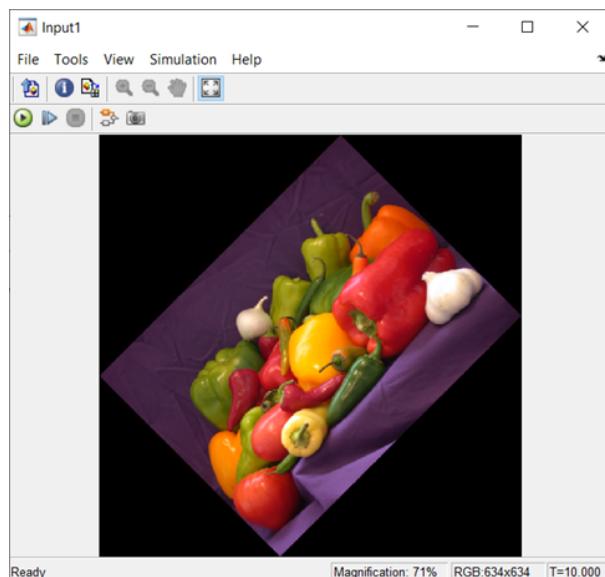
Δημιουργία μοντέλου Geometric Transformations/Resize το οποίο διπλασιάζει το μήκος της αρχικής εικόνας. Για να γίνει αυτό, ορίζουμε το Resize factor in% = [100 200], όπου το πρώτο στοιχείο αναφέρεται στις γραμμές και το δεύτερο στις στήλες.



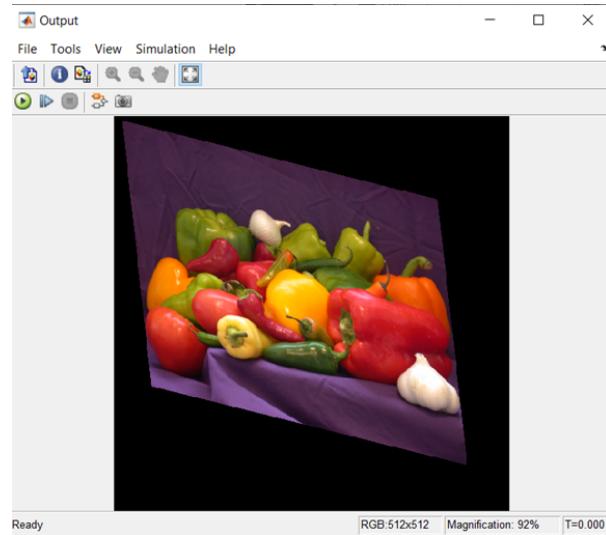
Μοντέλο Geometric Transformations/Translate το οποίο μετακινεί την εικόνα στους πάνω/χάτω, δεξιά/αριστερά. Ορίζουμε την τιμή Offset = [30 60], το οποίο σημαίνει ότι η εικόνα θα μετακινηθεί κατά 30 pixel στον άξονα y και 60 στον x. Αν θέλαμε να διατηρήσουμε το μέγεθος της εικόνας μετά την μετακίνηση, θα έπρεπε να ορίσουμε το Output size after translation = same as input image.



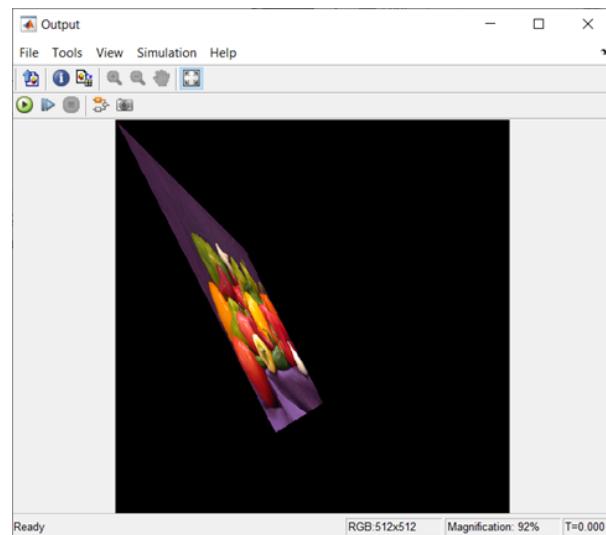
Μοντέλο Geometric Transformations/ Rotate το οποίο περιστρέφει την εικόνα. Ορίζουμε την παράμετρο Angle (radians) = $\pi/4$ για να περιστρέψουμε την εικόνα κατά 45 μοίρες.



3. Αρχικά, αντικαθιστούμε το block Apply Geometric Transformation με το block Warp, γιατί μας εμφανίζει μήνυμα ότι από την έκδοση R2015b, το πρώτο αντικαταστάθηκε με το Warp block. Ακόμα, αντικαθιστούμε το block Image from Workspace με το block Image from File, και ορίζουμε το File Name = peppers.png. Θέτουμε την παράμετρο Constant value = [0.8 0.2; 0.1 0.9; 10 5;], δηλαδή ορίζουμε το μητρώο μετασχηματισμού. Τέλος, στο block Warp θέτουμε το μέγεθος της εξόδου σε custom, ώστε μετά το σχηματισμό να φαίνεται όλη η εικόνα. Αν αφήναμε το μέγεθος της εξόδου ίσο με το μέγεθος της εισόδου, θα χανόταν μέρος της εικόνας στην έξοδο. Παρατηρούμε ότι οι ευθείες παραμένουν παράλληλες και μετά το μετασχηματισμό.

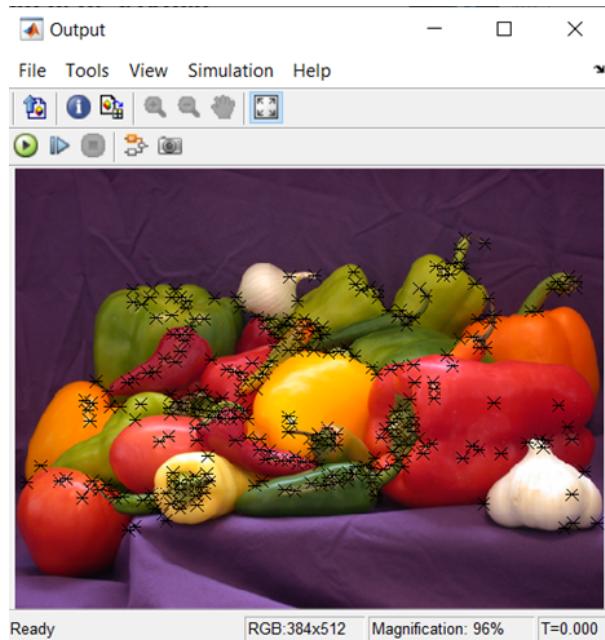


4. Αρχικά και σε αυτή την περίπτωση κάνουμε τις ίδιες αλλαγές στο μοντέλο, που κάνουμε και για τον affine μετασχηματισμό. Έπειτα, θέτουμε την παράμετρο Constant value = [0.53 0.53 0.001; 0.27 0.53 0; 100 100 0.5] που αποτελεί το 3×3 μητρώο μετασχηματισμού μας και θέτουμε στο block Warp το μέγεθος εξόδου σε custom με τιμή [200, 200, 512, 512], ώστε η έξοδος να έχει μέγεθος 512x512 (ορίζουμε να ξεκινήσει από το σημείο (200,200) γιατί από εκεί περίπου και πάνω είναι μόνο μαύρο φόντο). Πάρατηρούμε ότι μετά την εφαρμογή του μετασχηματισμού, οι ευθείες παρέμειναν ευθείες όλα οι πάνω και κάτω δεν είναι πλέον παράλληλες.

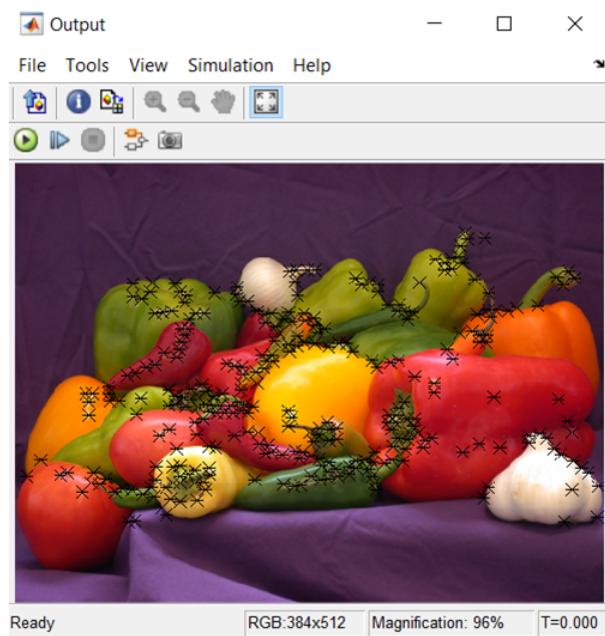


B.4 Ανίχνευση και Απεικόνιση Γωνιών σε Εικόνα

Harris Stephens Corner Detection: βασίζεται στις ιδιοτιμές του μητρώου αυτοσυσχέτισης, υπολογίζει το μέτρο απόχρισης της γωνίας χρησιμοποιώντας την ορίζουσα του μητρώου αυτοσυσχέτισης και το ίχνος του και είναι αρκετά ευαίσθητος στο θόρυβο.



Shi Tomasi Corner Detection: Βασίζεται και αυτός στις ιδιοτιμές του μητρώου αυτοσυσχέτισης όπως ο Harris-Stephens αλλά χρησιμοποιεί την ιδιοτιμή με την ελάχιστη τιμή. Χρησιμοποιεί και μία παράμετρο για εξισορρόπηση μεταξύ κορυφών και ακμών, πράγμα που τον αρκετά ανθεκτικό στο θορύβο.



Rosen Drummond Corner Detection: χρησιμοποιεί την ένταση φωτεινότητας του Pixel για να καθορίσει αν αποτελεί σημείο γωνίας ή όχι. Προκειμένου να αποτελεί σημείο γωνίας, θα πρέπει ένα αριθμός π γειτονικών pixel να έχουν μια σημαντική διαφορά στη φωτεινότητα σε σχέση με το Pixel που μας ενδιαφέρει. Ο αλγόριθμος είναι αρκετά ανθεκτικός στο θόρυβο.

