Wykład podstawowy z APD nr 2
(obejmujący uwagi, dodatki i suplementy
do zajęć z zakresu od 6 do 12 tygodnia zajęć)
prowadzący : mgr inż. Artur Bernat

Na wstępie pragnę zaznaczyć, że **pierwsze zajecia** miały na celu zaprezentować w zarysie ogólnym potencjalne możliwości zastosowań środowiska Matlab (6.x) w przetwarzaniu danych 2D, 3D, a w szczególności danych reprezentujących obrazy luminancji (intensywności jasności) lub obrazy chrominancji (tj. posiadające równiez informację o barwie np.: w reprezentacji RGB).

Zajęcia od 2 do 5 tygodnia włącznie z APD, posłużyły w skrótowym przeglądzie podstawowych poleceń nawigacyjnych w środowisku Matlab oraz w przeglądzie podstawowych operatorów, składni poleceń oraz samych funkcji i poleceń obszernego zestawu toolbox'ów. W szczególności w tym początkowym etapie zajęć z APD pomocna była lista funkcji dostępnych w *Image Processing Toolbox.*

W momencie kiedy pojawiła się możliwość realizacji zarówno tych prostszych jak i tych bardziej zaawansowanych zadań przetwarzania danych obrazów 2D, w rozpiętości zajęć od 6 do 14 tygodnia zajęć, postanowiłem z zajęć na zajęcia reprezentować wybrane (fragmentaryczne) problemy do rozwiązania.

W takim razie podstawowy przegląd poleceń nawigacyjnych Matlab'a (**2-5 tydzień**), listy operandów oraz poleceń, składni poleceń i funkcji w Matlab'ie powinien w przyszłości pojawić się jako suplement do zestawu wykładów podstawowych. Celem nadrzędnym przedmiotu jest bowiem omawianie (na wykładach) oraz praktyczne testowanie (na zajęciach laboratoryjnych) algorytmów przetwarzania danych, a nie samych narzędzi jako takich.

Ponieważ temat podstawowych przekształceń obrazu omawiany w ramach **6-8 tygodnia** zajęć ma zawsze szeroki odźwięk w Internecie, to tym temacie nie pojawi się w wykładzie podstawowym żadne wznowienie.

==========9 TYDZIEŃ==========

W temacie przekształceń obrazu 2D w reprezentacji z jednego układu współrzędnych (układ kartezjański) do innego układu (układ przekształceń biegunowych) (**9 tydzień**) zostałem zainspirowany zestawem scryptów Matlaba stworzonych przez Peter'a Kovesi z Zachodniej Australii (http://www.csse.uwa.edu.au). Zasadniczo, na swoich akademickich stronach, rozważa on zagadnienia detekcji krawędzi oraz obrysów obiektów 3D widocznych na obrazach 2D pod kątem analizy zgodności fazy czy też przystawania faz w treści obrazów (*ang. phase congruency*).

Niestety nie uzyskałem z tych stron informacji co do przekształcenia odwrotnego obrazu z reprezentacji we współrzędnych biegunowych na współrzędne kartezjańskie. Przekształcany był w tygodniu 9 zajęć obraz '*cameraman.tif*', który jest dostępny w katalogu zasobów demonstracyjnych Matlab'a.

Przekształcenia dostępne w skrypcie autorstwa Peter'a Kovesi uwzględniały możliwość wyboru długości promienia rozpiętości płaszczyzny obrazu przekształcanego. Mógł on swoją rozpiętością albo obejmować obszar bezpieczny, tj. minimalny obrazu, gdzie pomijane były obszary 4 rogów obrazu, albo obszar całkowity obrazu, wówczas w reprezentacji współrzędnych biegunowych widoczne były ciemne obszary poza rozciągłością oryginalnego obrazu, jako że promień miał długość równą połowie długości przekątnej obrazu. Ponadto odwzorowanie obrazu do współrzędnych biegunowych mogło być albo liniowe albo logarytmiczne. To drugie odwzorowanie przy stosowaniu rekonstrukcji obrazu w drugą stronę może skutecznie niwelować niekorzystne

zjawisko zmiennej rozdzielczości odwozorowywanych detali, ponieważ obszary w pobliżu środka układu współrzędnych biegunowych są zazwyczaj lepiej odwozorowywane w obrazie niż te znajdujące z dala od początku układu współrzędnych. Trzecią właściwością działań w skrypcie tego autora była możliwość doboru współrzędnych startowych przekształceń.

W treści oraz funkcjonalnym wykonaniu skryptu, na użytek 9 tygodnia zajęć, przyjąłem następujące uproszczenia:

1) skrypt będzie umożliwiał realizację tylko odwozorowania liniowego (tj. względem docelowej rozpiętości promienia wodzącego)
2) skrypt będzie realizował transformację układu współrzędnych w bezpiecznym obszarze obrazu, tj z pominięciem 4 rogów obrazu (na wynikowym obrazie nie będzie ciemnych obszarów spoza obrazu)
3) punkt startowy układu współrzędnych domyślnie będzie dotyczył środka (lub przybliżonego środka) we współrzędnych rozmiarów przekształcanego obrazu.

Oto treść skryptu *polartrans.m* przekształconego do *poltrans.m* według moich modyfikacji, wprowadzonych na użytek zajęć z APD:

```
%copyright 05.Dec.2004 Artur Bernat
%modified and simplified version
%of 'polartrans.m' Matlab script,which is
%available at http://www.csse.uwa.edu.au/~pk
%provided by Peter Kovesi of Western Australia
%-----------------------------------------------------------
%pim=poltrans(im,nrad,ntheta)
%where:
%      pim -   output image in polar coord.
%      im -    image in Cartesian coord. to be transformed
%      nrad -  resolution of radius length
%      ntheta -resolution in angle per full circle
%-------inherently, by implicit:--------------------------
%      The modifications introduced by Artur Bernat
%       are the following:
%      a)scale of transformation can be only linear,
%      b)starting point is within midle of the image
%      c)the range of the image transformed
%        equals to surface of a circle fully enclosed
%        within initial rectangle of the image
function pim=poltrans(im,nrad,ntheta)
[rows,cols]=size(im);
cx=cols/2+0.5;
cy=rows/2+0.5;
rmax=min([cx-1,cols-cx,cy-1,rows-cy]);
deltatheta=2*pi/ntheta;
deltarad=rmax/(nrad-1);
[theta,radius]=meshgrid([0:ntheta-1]*deltatheta,[0:nrad-1]*deltarad);
xi=radius.*cos(theta)+cx;
yi=radius.*sin(theta)+cy;
[x,y]=meshgrid([1:cols],[1:rows]);
pim=uint8(interp2(x,y,double(im),xi,yi));
```

Rys.1 Skrypt transformacji współrz. kartezjańskich. na współrz. biegunowe obrazu intensywności.

=>

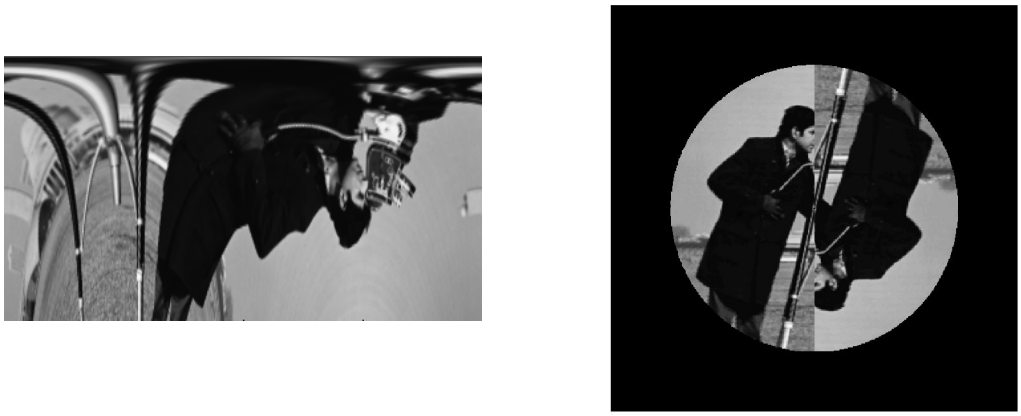
=>

Rys.2 Rezultat przekształceń obrazu intensywności jasności '*cameraman.tif*
(256x256 pikseli) na obraz we współrzędnych biegunowych
(200 gradacji długości pomienia wodzącego, rozciagającego się pionowo,
oraz 360 gradacji kąta wodzącego rozciągającego się na obrazie wynikowym poziomo)

Przekształcenie odwrotne odwzorowania obrazu do współrzędnych kartezjańskich wymagały użycia funkcji arcus tangens (*atan*), która w przeciwieństwie do funkcji tryg. sinusa i kosinusa wykazuje nieokreśloność w punktach -Π/2, Π/2 w całej domenie kąta odwozorowania <0,2Π >.

W takim razie, po wielu modyfikacjach, postanowiłem nie podać ostatecznego rozwiązania rzeszy Studentów z *APD*, a jedynie zademonstrować skrypt próbnego przekształcenia wraz z rezultatami:

=>



=>

Rys.3 Rezultat przekształceń obrazu intensywności z reprezentacji we współrz.biegunowych
(200 gradacji długości promienia wodzącego pionowo, 360 gradacji kąta wodzacego poziomo)
na obraz we współrzędnych kartezjańskich 256x256 pikseli.

Obszary ciemne na wynikowym obrazie reprezentują połacie bliskie rejonom 4 rogów obrazu w pierwotnym obrazie współrzędnych kartezjańskich, które ze względu na 'bezpieczną' długość maksymalnego promienia wodzącego przekształceń zostały pominięte.
Skrypt uruchomieniowy danego przekształcenia '*crttrans.m'* ,mojego autorstwa, należy uruchamiać wyłącznie po przekształceniach pierwotnych z użyciem skryptu '*poltrans.m'*, jako, że w skrypcie

tym przyjęte zostały trzy uproszczenia co do opcji działania i listy argumentów wejściowych, które nieprzestrzegane, mogą zniweczyć rezutat przekształcenia wtórnego.

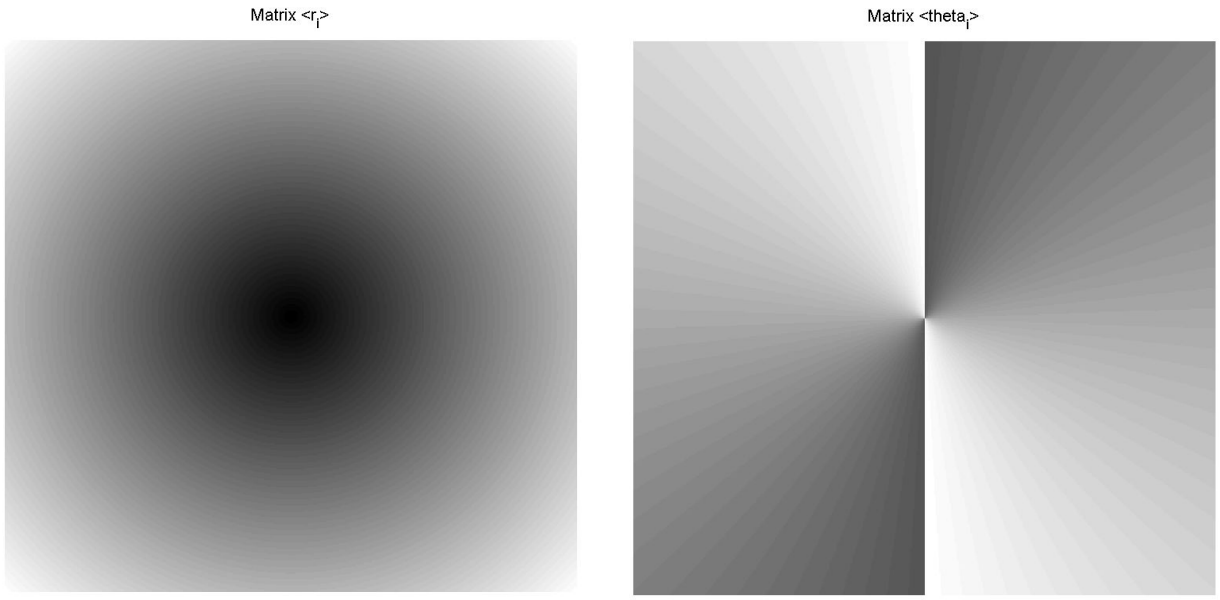
-

```
%copyright Artur Bernat 12.January.2004
%polar to Cartesian coordinates intensity image transformation
%-------------------------------------------------------
%cim=crtrans(pim,rows,cols)
%where:
%      pim -   input image in polar coords
%     1)with prior linear mapping from Cart. to polar coord.
%     2)prior Cart.coord. origin placed in centre of the image
%     3)short prior Cart. to polar coord. radius
%       fully enclosed with initial image rectangle
%
%      rows -  resolution in rows of the resulting Cart.Coord. image
%      cols - resolution in columns of the resulting Cart.Coord.image
%--------------------------------
function cim=crttrans(im,rows,cols)
[nrad,ntheta]=size(im);

delta_x=1/cols;
delta_y=1/rows;

[x,y]=meshgrid(delta_x/4+[-cols:1:cols-1]*delta_x,delta_y/4+[-rows:1:rows-1]*delta_y);

ri=(2*nrad/sqrt(2))*sqrt((x).^2+(y).^2);
theta_i=(ntheta/(2*pi))*atan((y)./(x));

theta_i=theta_i+180;

figure,imshow(uint8(255*ri/max(ri(:))));
hold on ;
title('Matrix <r_i>');
figure,imshow(uint8(255*theta_i/max(theta_i(:))));
hold on;
title('Matrix <theta_i>');
[theta,radius]=meshgrid([1:ntheta],[1:nrad]);
scale_x=max(theta_i(:))/max(theta(:));
scale_y=max(ri(:))/max(radius(:));
cim=uint8(interp2(theta,radius,double(im),theta_i,ri));
```

Rys.4 Skrypt *'crttrans.m'* wtórnego przekształcenia obrazu ze współrz. kartezjańskich na biegunowe

Ponadto, w obrazie wynikowym nastąpiło zdwojenie połówki II i III oryginalnego obrazu z charakterystycznym odwróceniem kopii, co stanowi postanowioną przeze mnie zagadkę do rozwiązania dla rzeszy Studentów z APD. Pragnę jedynie zaznaczyć, że wartości indeksowe macierzy *x,y* sprzężonych ze sobą, a stworzonych za pomocą polecenia *meshgrid* rozciągają się od wartości ujemnych poprzez zero do wartości dodatnich.

Same wartości macierzy *ri* oraz *theta_i*, po odpowiednim przeskalowaniu również poddane są wizualizacji we skrypcie (wartości analogiczne macierzy *xi* oraz *yi* stworzonych na użytek transformacji pierwotnej zostały już podane w notatkach uzupełniających z APD do 9 tygodnia zajęć)



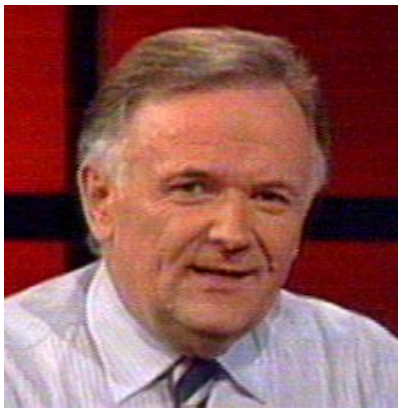
Rys.5 Wizualizacja treści macierzy r_i oraz $theta_i$, służących w odwozorowaniu wtórnym przekształacenia współrzędnych biegunowych na kartezjańskie.

Jak widać z treści rys.5 o ile całkowitą symetrią (tj.izotropowością kierunkową) charakteryzuje się macierz pierwsza, to w przypadku wartości macierzy *thetai* istnieje nieciągłość na wartościach kąta równego -Π/2, Π/2.

==========TYDZIEN 10 i 11==========

Te tygodnie zajęć z APD dotyczyły tematu, będącego moim zdaniem, zasadniczym tematem całych zajęć z APD, mianowicie rekonstrukcji stereometrii obiektów 3D, których zewnętrzny wygląd, postać została zarejestrowana, czy też utrwalona na obrazie 2D. Na zajęciach praktykowano rekonstrukcję i wizualizację kształtu różnych sfotografowanych obiektów zarówno z teksturą oryginalnego obrazu 2D nałożonego na wykres 3D tworzony np.: z wykorzystaniem funkcji *mesh* jak i bez teksturowania. Wówczas wykres 3D stworzony czy to za pomocą funkcji *mesh* czy to funkcji *surfl* (*ang. Surface Lit by virtual light source*) był wyrażony w skali głębokości.

Dość skąpe notatki uzupełniające z 10,11 tygodnia, nie uwzględniły w swojej treści wizualizacji obiektów z teksturowaniem. Ponadto nie podano praktycznej realizacji teksturowania wykresu 3D z użyciem kolorowego obrazu pierwotnego (w reprezentacji RGB) z obiektem poddawanym rekonstrukcji kształtu.



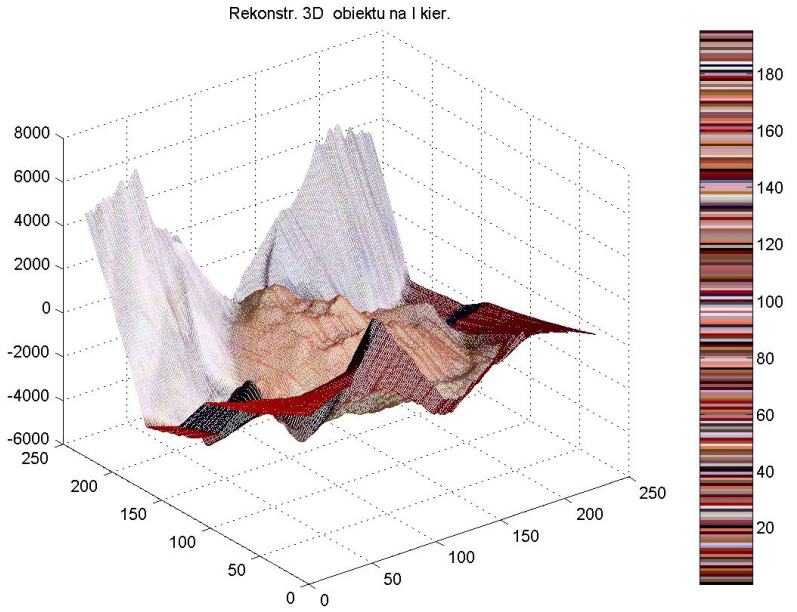
Rys.6 Pierwszy obraz, uzyskany w trakcie oglądactwa programu «*EcoMniejWięcej »*

Obraz twarzy ludzkiej z rys.6 to dane wymiaru 224x229x24bityRBG. Ponieważ mamy do czynienia z mapą kolorową, należy ją najpierw zindeksować (sposób indeksacji podano z I wykładzie podstawowym z APD), najlepiej z gradacją 196 kolorów listy indeksowej.

```
%copyright 11.Jan.2005 Artur Bernat
%revised 3D map reconstruction script
% kx- first direction,ky - second direction
function [kcx,kcy]=im2_3Drev_rgb(Mrgb)
%------------------------
%mapa naliczana metodą sum kumulacyjnych na dwóch kier.
%kąt nachylenia proporcjonalny do średniej
%luminancji obrazu w skali od 0 na 255
M=rgb2gray(Mrgb);
[indM,map196]=rgb2ind(Mrgb,196);
kx=cumsum(double(M),1);   %Ikierunek
%------------------------
M2=rot90(M,1);%rotacja macierzy M
ky=cumsum(double(M2),2); %IIkierunek
%------------------------
%kcx zarys 3D po odjęciu rampy średniej luminancji
%albo w skrócie zarys 3D na I kierunku
kcx=kx;
for j=1:size(M,2),
 y=1:size(M,1);
 [p,s]=polyfit(y',kx(:,j),1);
 cx=p(2)+p(1)*y';
 kcx(:,j)=kx(:,j)-cx;
end;
%-------------------------
%skorygowany zarys 3D na  II kierunku
kcy=ky;
for j=1:size(M2,2),
 y=1:size(M2,1);
 [p,s]=polyfit(y',ky(:,j),1);
 cy=p(2)+p(1)*y';
 kcy(:,j)=ky(:,j)-cy;
end;
%------------------
kcy=rot90(kcy,-1); %rotacja wtórna mapy danych kcy
figure,mesh(kcx,double(indM));hold on;colormap(map196);colorbar('vert');
view(3);title('Rekonstr. 3D  obiektu na I kier.');
figure,for i=1:6,subplot(2,3,i);mesh(kcx,double(indM));hold on;colormap(map196);
view(i*15,15);title(['Azymut:' num2str(i*15) ', El:45']);end;
```

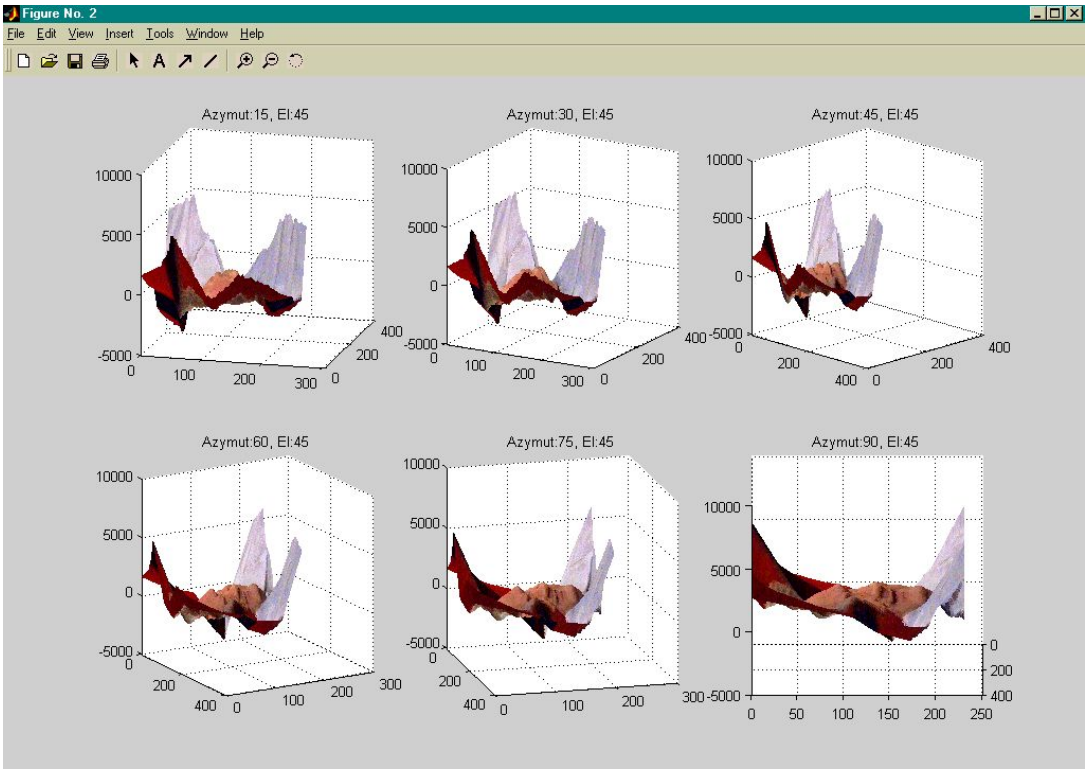
Rys.7 Przykładowy skrypt *'im2D_3Drev_rgb.m'* będący niewielką modyfikacją skryptu *'im2D_3Drev.m'* zaprezentowanego w I wykładzie podstawowym z APD.

W skrypcie o treści podanej na rys.7 najpierw wygenerowano dużego formatu wykres 3D ze zrekonstruowanym kształtem centralnych elementów twarzy, a następnie wygenerowano wykres z 6 podwykresami obrazującymi zrekonstruowany kształt pod różnym kątem azymutalnym punktu obserwatora.

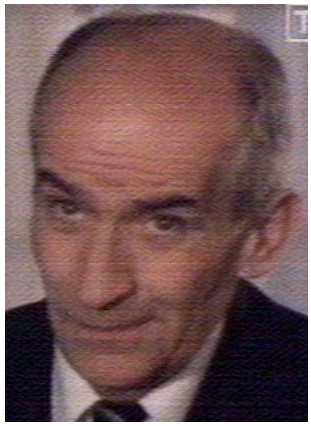


Rys.8 Zrekonstruowany kształt twarzy ludzkiej wraz z peryferyjnymi elementami obrazu typu: powierzchnia koszuly, czerwone tło studia za planem głównym obrazu. Po prawej widać charakterystyczny '*wzór*' paska zindeksowanej listy kolorów. Nie jest to jak widać '*widmo*' ciągłe barw paska głębokości, a jedynie zoptymalizowana lista 196 kolorów.

-



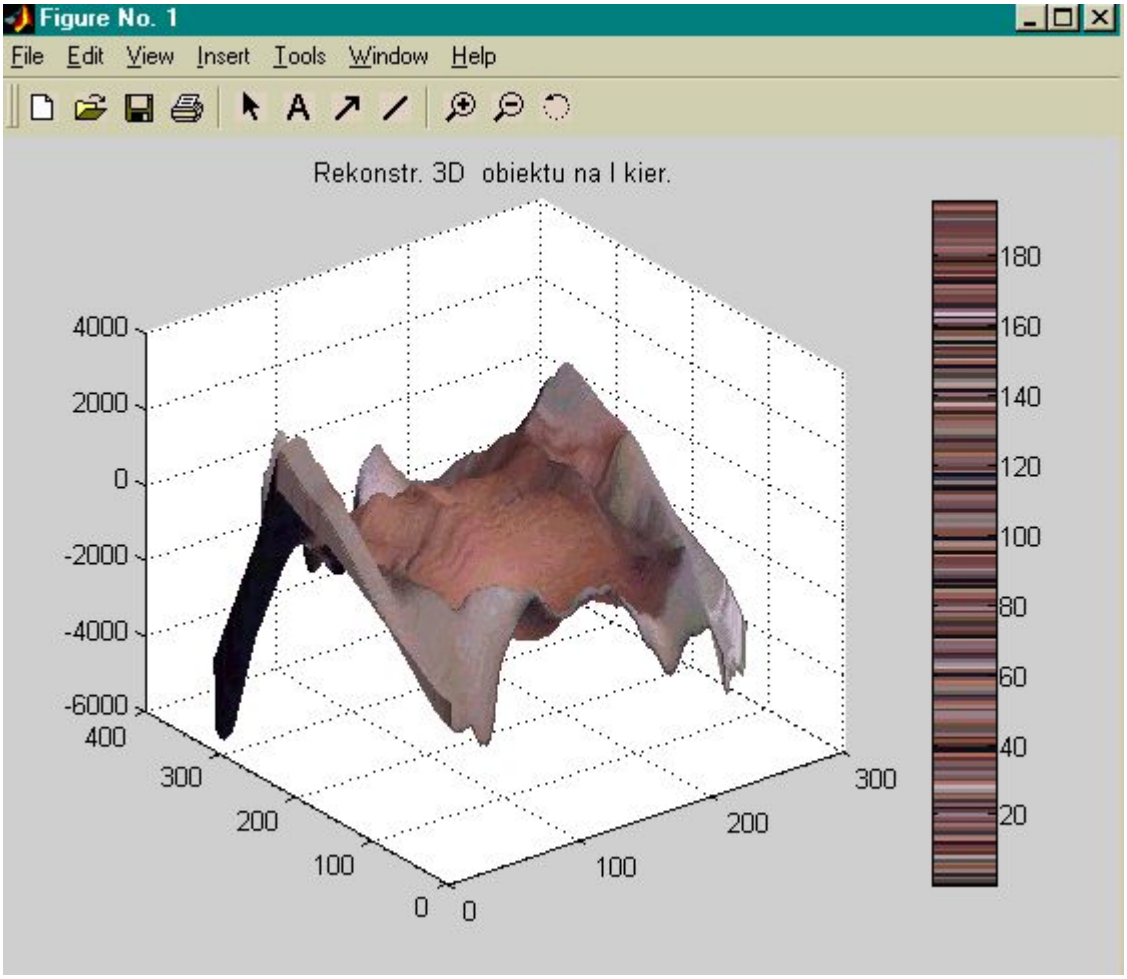
Rys9 Zrekonstruowany kształt twarzy ludzkiej przy zmiennym azymucie punktu obserwatora.



Rys.10 Drugi obraz, Louis w fabule filmu « Panowie szanujcie swoje Żony ».225x310x24bityRBG

Treść obrazu z rys.10 bezpośrednio podddana działaniu skryptu o treści z rys.7 wykazywała zbyt duże zaszumienie. W takim razie obraz poddano wstępnie wygładzaniu (czytaj. odszumianiu, z wykorzystaniem filtru konwolucji o rozmiarze maski 3x3 piksele).

-

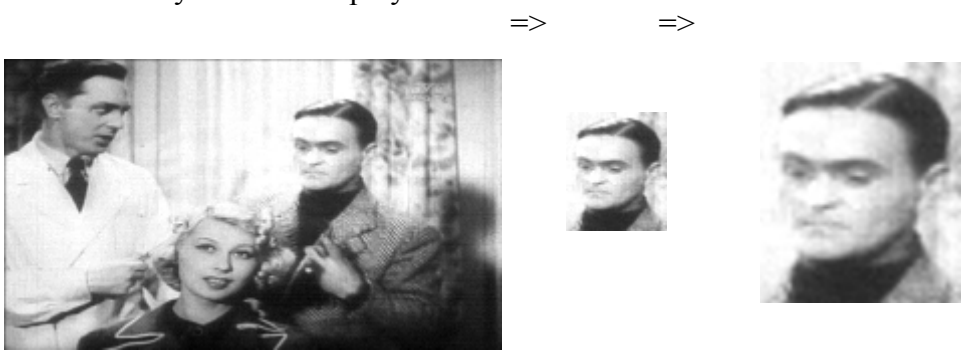


Rys.11 Wykres 3D rekonstrukcji twarzy z obrazu na rys.10 uzyskany z użyciem skryptu o treści z rys.7

Wykres wielokrotny, z 6 podwykresami 3D przy zmiennym azymucie punktu obserwatora nie będzie tutaj przytoczony, jakkolwiek trzeba stwierdzić fakt stosunkowo wiernej rekonstrukcji wysokiego, zmarszczonego czoła oraz oczodołów wraz z obowiązkowo, zawsze w stu procentach poprawnie rekonstruowanym nosem. Poprawność rekonstrukcji twarzy ludzkich utrwalonych na

obrazach 2D zależy od kierunku analizy z użyciem sum kumulacyjnych jak również w dużym stopniu od korzystnego układu elementów brzegowych obrazu, niewchodzących w skład układu elemetów składających się na twarz. Na korzystny układ elementów brzegowych obrazu mających również wpływ na globalny zarys głowy mają więc: elementy typu włosy, ubranie (koszula, sweter ich jasność, chromancja) oraz jasność i chrominancja planu dalszego obrazu.

Jeszcze tylko maleńki przykład:

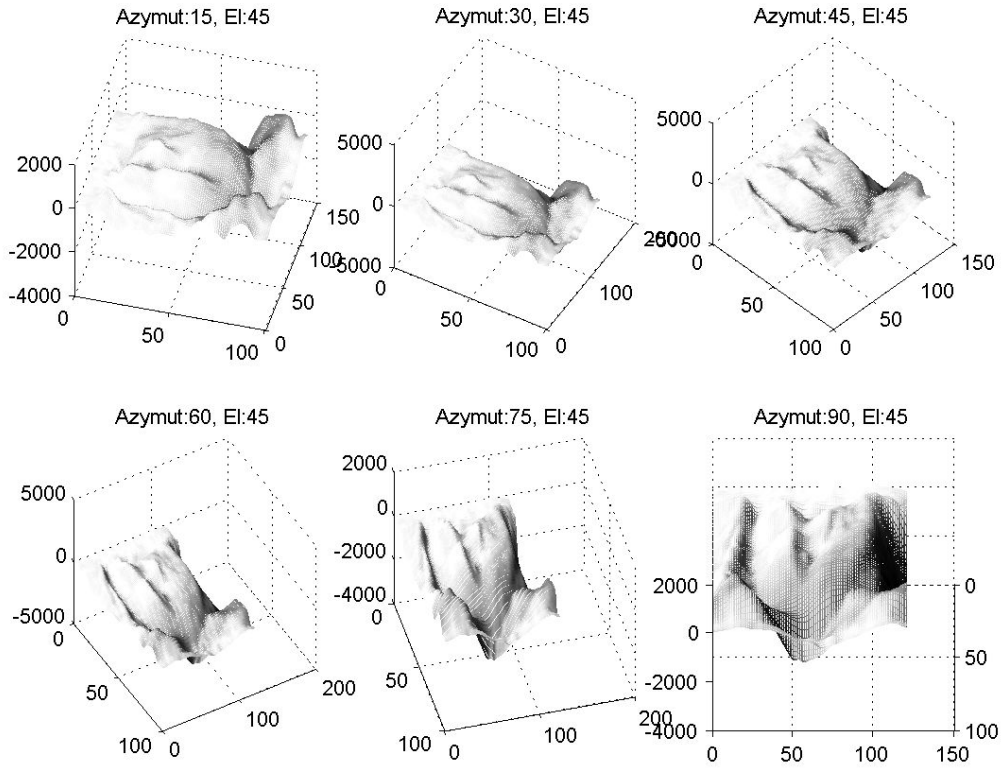


Rys.11 Wycinek twarzy Adolfa z pradawnego filmu o rozmiarach 50x59 pikseli przeskalowano do rozmaru 100x120 pikseli pozbywając się przy tym efektu pikselizacji obrazu twarzy.

-

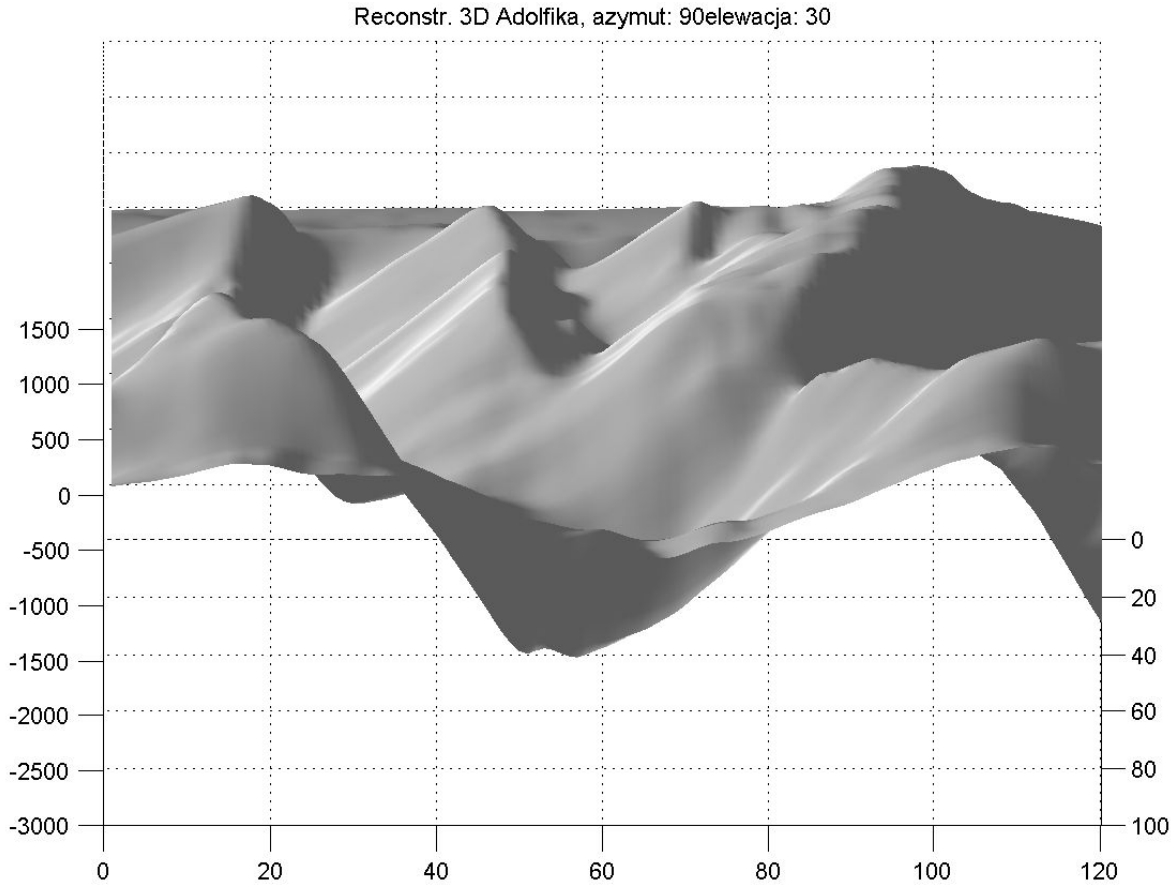
Z wykorzystaniem skryptu analogicznego w działaniu do tego z rys.7, lecz operującego na prostszych danych obrazu intensywności jasności wygenerowano złożony wykres z 6 punktami obserwatora przy zmiennym azymucie i stałym kącie elewacji:

-



Rys.12 Zrekonstruowany kształt twarzy Adolfa przy zmiennym azymucie punktu obserwatora.

Ponieważ wykres z rys.12 nie oddaje w całej okazałości istoty omawianej metody rekonstrukcji kształtu obiektów, a w szczególności twarzy ludzkich, którą jest uderzająca wierność i poprawność, zdecydowano się na zamieszczenie w ramach tego wykładu podstawowego z APD powierzchni 3D poddanej wizualizacji z wykorzystaniem funkcji *surfl*.
-

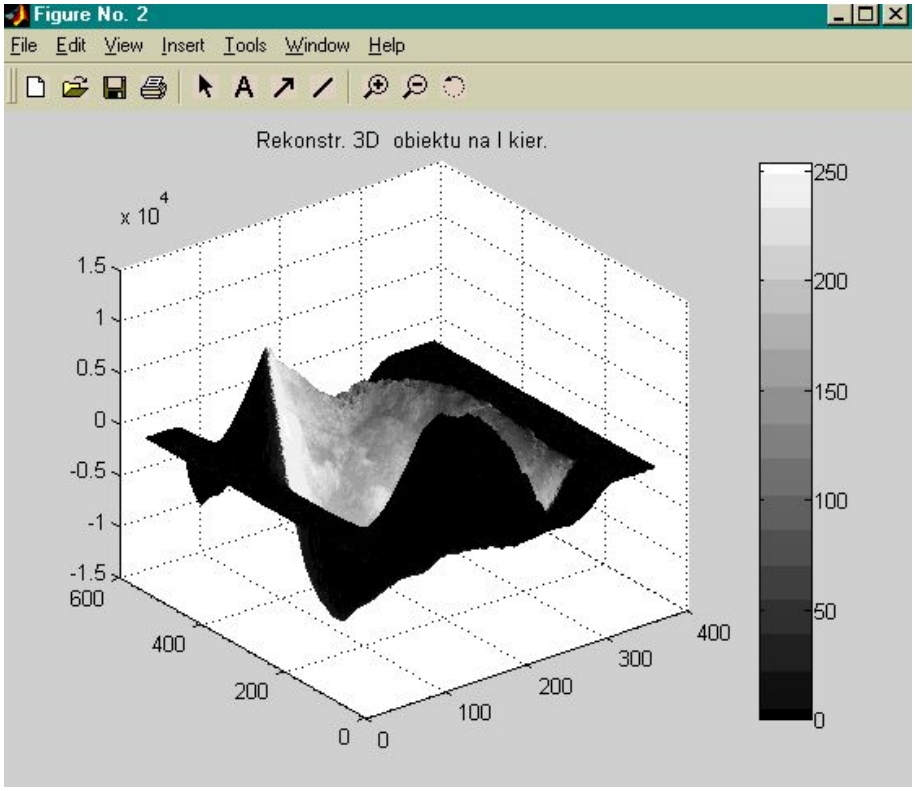


Rys.13 Wykres 3D zarysu twarzy Adolfa w formie 'odlewu'

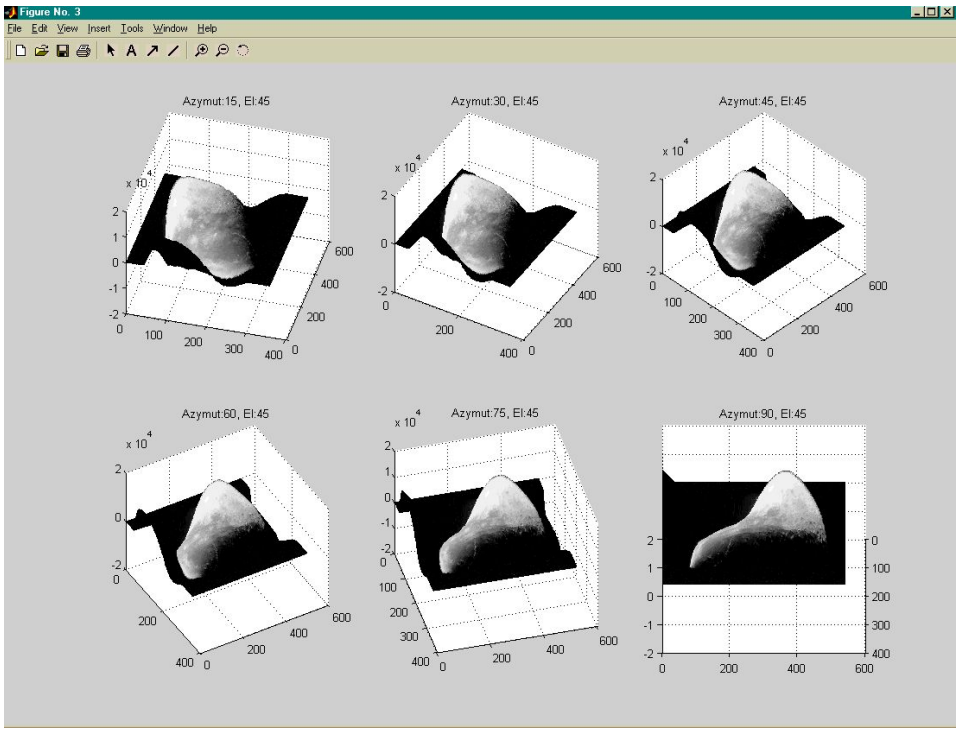
Jak widać nieznaczne odchylenie sfotografowanej twarzy na kierunku azymutalnym oraz w pionie na planie filmu nie jest w stanie, w omawianej metodzie, zakłócić procesu rekonstrukcji twarzy z tak dobrze uwidocznionymi elementami brzegowymi twarzy jak podbródek, a czasami nawet wyższe partie czoła, włosy na głowie czy uszy.

W omawianym na zajęciach (10,11 tygodnia) przykładzie rekonstrukcji oświetlonej półkuli Księżyca, dodanie elementu 'nateksturowania' ujawnia globalna tendencję do rekonstrukcji sfery, z lokalnymi jednak istotnymi niekształceniami wynikającymi z różnego współczynnika odbicia.
-

Rys.14 Obraz '*moon.tif*' z zasobów demonstracyjnych Matlab'a



Rys.15 Wykres podstawowy próby rekonstrukcji 3D obiektu z fotografii '*moon.tif*' z wykorzystaniem skryptu analogicznego do tego z rys.7 wraz z teksturowanem,



Rys.16 Zrekonstruowany kształt obiektu przy zmiennym azymucie punkcie obserwatora

Jak widać do poprawnej rekonstrukcji globalnego zarysu Księżyca brakuje w tej konkretnej

sytuacji dodatkowych przesłanek, które by 'wymuszały' poprawny sferoidalny kształt oświetlonej półkuli. Ponadto, na wskutek istnienia różnego współczynnika odbicia światła w rejonach bardziej górzystych na granicy światło-cienia oraz na połaciach nizinnych planety, całość rekonstruowanej planety bardziej przypomina twór stożkowaty niż sferoidalny.

Na zajęciach 10,11 tygodnia zajęć podano również ogólne 'recepty' i założenia na wykorzystywanie zestawu prostych funkcji i poleceń w zadaniu tworzenia animacji, z przeznaczeniem zapisu ich treści na dysk.
Oto skrótowe przypomnienie:

M=aviread('nazwa.avi'); <= dokonuje wczytania pod zmienną M środowiska Matlab'a filmu avi, najlepiej o standarcie kompresji « Indeo3 », albo »Indeo5 » albo też « Cinepak »(są to standardowe kodery platformy Win2000/XP)

movie2avi(M,'nazwa.avi','Fps',16,'Compression','Indeo5',Colormap,paleta_kolorow);
<=to przykład typowej składni polecenia zapisu animacji *M* na dysk z 16 klatkami odtwarzania na sekundę , przy standarcie kompresji « Indeo5 », oraz z paletą kolorów *paleta_kolorów.*

img=frame2im(M(5)); <=polecenie konwersji 5 klatki animacji *M* do formatu obrazu *img*

M(10)=im2frame(img,paleta)<=polecenie wstawienia obrazu (intensywności jasności albo kolorowego indeksowanego z paletą kolorów *paleta*) na miejsce 10 klatki animacji *M.*

M(i)=getframe; <=polecenie wstawienia obrazu z bieżacego okna graficznej prezentacji danych pod i-tą klatkę animacji M

Przykładowy skrypt 'składu animacji' na podstawie wygenerowanych wcześniej danych 3D *dane* oraz obrazu 2D *obraz* może przyjąć następującą treść:

```
%---------------------------------
function M=mov(dane,paleta,obraz,nfrm)
figure,mesh(dane,double(obraz));
colormap(paleta);
step=round(360/nfrm);
for i=1:nfrm,
   view(step*i,45);
   M(i)=getframe;
end;
movie(M);
%-----------------
```

Rys.17 Treść skryptu tworzenia animacji przy zadanej danej 3D, obrazie pierwotnym 2D, palecie reprezentacji kolorów oraz żadanej liczbie klatek obejmujących pełny kąt obrotu wykresu 3D.

==========TYDZIEŃ 12==========

W 12 tygodniu zajęć przedstawiono problem związany z układem akwizycji danych o matrycy CCD kamery typu WebCam generującej na treści obrazu 2D ciemne pasy, o zaniżonej w ich obrębie jasności obrazu. Funkcja względnych zmian średniej jasności tych pionowych pasów na obrazie przypomina zmienność funkcji expontencjalnej z wykładnikiem o ujemnej wartości. Ponadto współczynniki takiej funkcji dopasowania, jeśli nawet możliwe byłoby ich wyznaczenie będą miały niezależne wartości dla pasów parzystych i nieparzystych. Sam obraz wykazuje ponadto

w detalach silnie lokalne zmiany jasności (również przejaśnienia), a do tego trzeba uwzględnić zmienność zjawiska w zależności tego czy obraz akwizycji występuje w wymiarze podstawowym, podwójnym, albo półówkowym.

W takim razie wprowadzono istotną modyfikację w zasadzie działania algorytmu korekcji luminancji obrazu, biorąc pod uwagę również obraz odniesienia otrzymywany z układu akwizycji danych w wyniku rozstrojenia układu optycznego. Na podstawie obrazu odniesienia, po obliczeniu średniej wartości luminancji w jego obrębie można dokonać przeskalowania całości obrazu z wartościami mnożników w postaci macierzy 2D, która wymiarami będzie przystawać do wymiarów obrazu korygowanego.

W przypadku pikseli nie wykazujących istotnych odchyleń od wartości średniej w obrazie, wartosci mnożnikow będą bliskie jedności. W przypadku jednak wyraźnych odstępstw sredniej luminancji od wartosci średniej na obrazie odniesienia wartości generowane w tablicy 2D monożników mogą osiągać wartości rzędu 1.2-1.3. W końcowej operacji po-elementowego mnożenia tablicy obrazu intensywności luminancji z tablicą 2D mnożników, przyczyni się to do rozjaśnienia ciemnych obszarów obrazu pierwotnego.

W przypadku obrazu w reprezentacji RGB, nalezałoby zwrócić baczniejszą uwage na drobne zmiany barwy, jako że przy skalowaniu jedna lub więcej ze składowych RGB mających początkowy duży udział w tworzeniu barwy obrazu może ulec w wyniku skalowania nasyceniu. Być może właściwsze byłoby dokonywanie korekcji w przestrzenii HSI (ang.Hue Saturation Intensity) niedopuszczając do nadmiernych zniekształceń danych.

Po za zajęciami 12 tygodnia eksperymentowano ze stworzonym algorytmem, modyfkując go w ten sposób, aby możliwa była korekcja filmów avi nagrywanych w celu dynamicznej wizualizacji obrazów z próbek taśm ściernych.

Otóż ciekawy sposób działania algorytmu o treści skryptów zasadniczo zgodnych z tymi podanymi na zajęciach 12 tygodnia zajęć z APD uzyskuje się jeśli za argument pełniący rolę obrazu odniesienia w lokalnej korekcji luminancji poda się obraz z pierwszej klatki animacji. W przypadku ruchomej treści animacji avi, otrzymujemy detektor typu 'żabie oko' regujące zmianą treści wynikowej (czytaj: skorygowanej) animacji na dynamiczne zmiany obrazu, podczas gdy obrazy statyczne dają w wyniku jednolitą mapę o wartości luminancji odpowiadającej średniej luminancji w obrazie oryginalnym.