%%

function varargout = showMaskAsOverlay(opacity, mask, overlaycolor, varargin)

narginchk(1,5);

deleMasks = true; %Default

imgax = [];

for ii = 1:numel(varargin)

if isempty(varargin{ii}),continue,end;

if ishandle(varargin{ii})&&varargin{ii}~=0

imgax = varargin{ii};

else

deleMasks = varargin{ii};

end

end

if isempty(imgax)

imgax = imgca; %Default

end

if isa(deleMasks,'double'),deleMasks = logical(deleMasks);end

fig = ancestor(imgax,'figure');

validateattributes(opacity, {'double','logical'},{'nonempty'},mfilename, 'opacity', 1);

validateattributes(deleMasks, {'logical'},{'nonempty'}, mfilename, 'deleMasks', 5);

if nargin == 1

overlay = findall(gcf,'tag','opaqueOverlay');

if isempty(overlay)

error('SHOWMASKASOVERLAY: No opaque mask found in current figure.');

end

mask = get(overlay,'cdata');

newmask = max(0,min(1,double(any(mask,3))\*opacity));

set(overlay,'alphadata',newmask);

figure(fig);

%tofront(get(fig,'name'));

return

else

validateattributes(mask, {'double','logical'},{'nonempty'}, mfilename, 'mask', 2);

end

% If the user doesn't specify the color, use red.

DEFAULT\_COLOR = [1 0 0];

if nargin < 3

overlaycolor = DEFAULT\_COLOR;

elseif ischar(overlaycolor)

switch overlaycolor

case {'y','yellow'}

overlaycolor = [1 1 0];

case {'m','magenta'}

overlaycolor = [1 0 1];

case {'c','cyan'}

overlaycolor = [0 1 1];

case {'r','red'}

overlaycolor = [1 0 0];

case {'g','green'}

overlaycolor = [0 1 0];

case {'b','blue'}

overlaycolor = [0 0 1];

case {'w','white'}

overlaycolor = [1 1 1];

case {'k','black'}

overlaycolor = [0 0 0];

otherwise

disp('Unrecognized color specifier; using default.');

overlaycolor = DEFAULT\_COLOR;

end

end

figure(fig);

if isempty(imgax)

tmp = imhandles(fig);

else

tmp = imhandles(imgax);

end

if isempty(tmp)

error('There doesn''t appear to be an image in the current figure.');

end

try

a = imattributes(tmp(1));

catch

error('There doesn''t appear to be an image in the current figure.');

end

imsz = [str2num(a{2,2}),str2num(a{1,2})]; %#ok

if ~isequal(imsz,size(mask(:,:,1)))

error('Size mismatch');

end

if deleMasks

%delete(findall(fig,'tag','opaqueOverlay'))

delete(findall(imgax,'tag','opaqueOverlay'))

end

overlaycolor = im2double(overlaycolor);

% Ensure that mask is logical

mask = logical(mask);

if size(mask,3) == 1

newmaskR = zeros(imsz);

newmaskG = newmaskR;

newmaskB = newmaskR;

newmaskR(mask) = overlaycolor(1);

newmaskG(mask) = overlaycolor(2);

newmaskB(mask) = overlaycolor(3);

elseif size(mask,3) == 3

newmaskR = mask(:,:,1);

newmaskG = mask(:,:,2);

newmaskB = mask(:,:,3);

else

beep;

disp('Unsupported masktype in showImageAsOverlay.');

return

end

newmask = im2uint8(cat(3,newmaskR,newmaskG,newmaskB));

hold(imgax,'on');

h = imshow(newmask,'parent',imgax);

try set(h,'alphadata',double(mask)\*opacity,'tag','opaqueOverlay');

catch

set(h,'alphadata',opacity,'tag','opaqueOverlay');

end

if nargout > 0

varargout{1} = findall(imgax,'tag','opaqueOverlay');

end

if nargout > 1

% This returns a "concatenated" rgb image of the

% color-coded overlaid mask.

tmp = get(varargout{1}(1),'cdata');

switch a{3,2}

case 'uint8'

mult = 255;

case 'uint16'

mult = 2^16;

otherwise

mult = 1;

end

r = tmp(:,:,1);

g = tmp(:,:,2);

b = tmp(:,:,3);

r(mask) = overlaycolor(1)\*mult;

g(mask) = overlaycolor(2)\*mult;

b(mask) = overlaycolor(3)\*mult;

varargout{2} = cast(cat(3,r,g,b),a{3,2});

end