SANDIA REPORT

SAND2013-1622
Unlimited Release
Printed March 2013

Strip2CubeFace User's Manual

Geoffrey E. Forden

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Approved for public release; further dissemination unlimited.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government, nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, make any warranty, express or implied, or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represent that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof, or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof, or any of their contractors.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.osti.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd.
Springfield, VA 22161

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2013-1622
Unlimited Release
Printed March 2013

Strip2CubeFace User's Manual

Geoffrey E. Forden
International Nuclear Threat Reduction
Sandia National Laboratories
P.O. Box 5800
Albuquerque, New Mexico 87185-MS1373

Abstract

Commercial Off The Shelf (COTS) software for producing linked virtual tours based on 360 degree panoramas are becoming more and more available. However, the best current products for taking the images, stitching them into 360 degree panoramas, and then linking them together into complex virtual tours require different and incompatible input and output formats. This program is designed to bridge the gap between the iPix Interactive Studio export format, which consists of a single JPEG with the six faces of a cube connected horizontally, with the six individual JPEGs needed to be imported into Panotour Pro software. This report describes how to use the software program Strip2CubeFace, which takes the cube-strip JPEG exported from iPix Studio and coverts it into six JPEGs representing the six cube faces that Panotour Pro imports. As such, it represents a necessary link between the two COTS software programs key to making virtual tours quickly and easily. It becomes one member of the suite of software programs known as "RaPP-TOURS" or Rapid Processing of PanoTours Software necessary to simulate managed access and other permission requesting arms control-type training exercises.

CONTENTS

1. Introduction 7
2. Installing Strip2CubeFace 9
3. Running Strip2CubeFace Software 11
4. References 15
Appendix: Source Code 17
Distribution 25

FIGURES

Figure 1. Example of an export output format from the iPix Interactive Studio 7
Figure 2. Output of Strip2CubeFace resulting from processing image in Figure 1 7
Figure 3. Dialog box for running Strip2CubeFace. 11
Figure 4. Dialog for changing the input directory, where the files to be converted are located. Setting this will change the default directory for input files until the next time the "Change Input Directory" button is pushed. 12
Figure 5. Dialog resulting from pushing the "Browse" button. 12

NOMENCLATURE

iPix Interactive Studio	A program for "stitching" images together into a 360 degree panorama
JPEG	A commonly used method of compression for digital image storage.
Panotour Pro	A program for linking together a series of 360 degree panoramas into a virtual tour.

1. INTRODUCTION

Commercial Off The Shelf software for producing linked virtual tours based on 360 degree panoramas are becoming more and more available. However, the best current products for taking the images, stitching them into 360 degree panoramas, and then linking them together into complex virtual tours require different input and output formats. This program is designed to bridge the gap between the iPix Interactive Studio[1] export format, which consists of a single JPEG with the six faces of a cube connected horizontally (see Figure 1), with the six individual JPEGs needed to be imported into Panotour Pro software[2]; see

Figure 1. Example of an export output format from the iPix Interactive Studio

Figure 2. Output of Strip2CubeFace resulting from processing image in Figure 1

2. INSTALLING STRIP2CUBEFACE

Copy the installation package (strip2CubeFace_pkg.exe) to the directory where you would like to run it. Double click on the executable. A command window will pop up as the program installs the MATLAB COMPILER RUNTIME library. This software has been licensed to install this library from MathWorks without the user needing a license. After this has run, it will have created an executable, strip2CubeFace.exe, in that same directory. Double click on this executable to run the program. The executable strip2CubeFace_pkg.exe may now be deleted if desired.

3. RUNNING STRIP2CUBEFACE SOFTWARE

Double click on the executable, strip2CubeFace.exe. A dialog box will appear for running the conversion program. See Figure 3. The Dialog box consists of several boxes for inputting text and buttons for changing directories or finding the file to be converted.

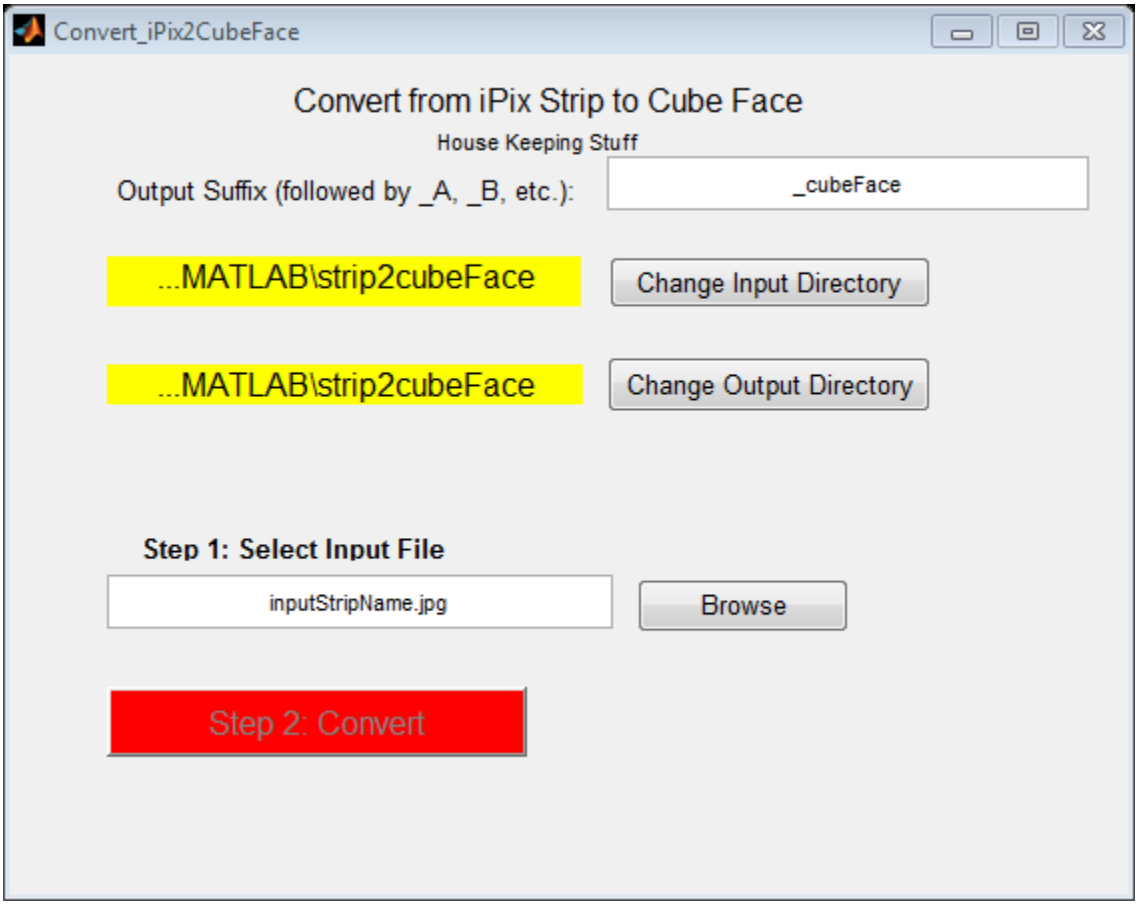

Figure 3. Dialog box for running Strip2CubeFace.

At the top of the dialog box, is a text box with the first part of the string used as the suffix identifier. The default output file name is the input file name concatenated with" _cubeface_A" through "_cubeface_F." The "_cubeface" part may be changed by overwriting the text box in the upper right hand corner but the output files will always end with one of the final suffixes "_A" through "_F."

The input directory, shown in the top-most yellow box, may be changed by pushing the "Change Input Directory" button. Figure 4 shows the dialog resulting from pushing the Change Input Directory button. Once changed, the selected folder becomes the default directory for future input files until the "Change Input Directory" button is pushed again. A similar dialog box appears when the "Change Output Directory" button is pushed.

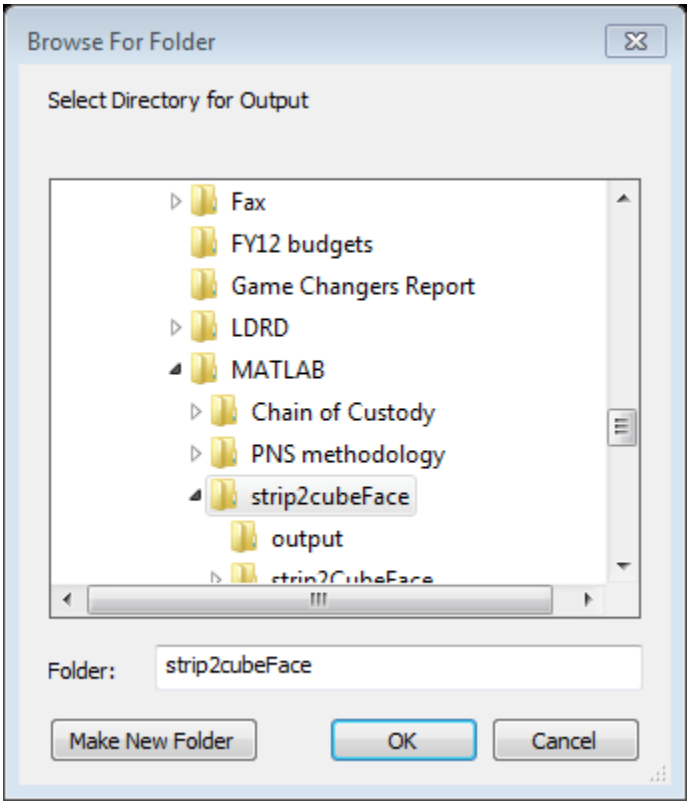

Figure 4. Dialog for changing the input directory, where the files to be converted are located. Setting this will change the default directory for input files until the next time the "Change Input Directory" button is pushed.

The input file may be selected by either browsing for it (by pushing the "Browse" button beside the "Step 1: Select Input File" text box, see) or directly typing in the new file name into that text box. Once a new file has been selected, the red "Step 2: Convert" button becomes enabled. Pushing that button converts the input file into the six output files with the different suffixes selected before.

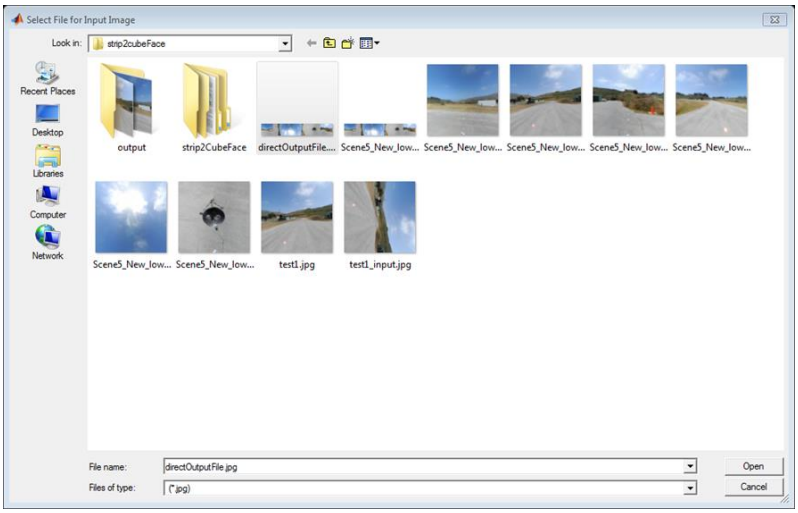

Figure 5. Dialog resulting from pushing the "Browse" button.

The six new cube face images, ending in “_A” through “_F” may now be directly imported into Panotour Pro.

4. REFERENCES

1. iPix Interactive Studio 2.0 Product Sheet, available on line at http://www.ipix.com/img/document_files/IPX-IS.pdf, last accessed 11/13/12
2. Kolor Panotour Pro 1.8, http://www.kolor.com/panotour-pro-profesionnal-360-virtual-tour-software-home.html, last accessed 11/13/12

APPENDIX: SOURCE CODE

```
function varargout = Convert_iPix2CubeFace(varargin)
%
% Author: Geoff Forden, geforde@sandia.gov
%
% This program takes a "cube strip" of images generated by iPix Studio and
% splits them into the appropriate six cube faces for importing into
% Panotour Pro
%
% CONVERT_IPIX2CUBEFACE MATLAB code for Convert_iPix2CubeFace.fig
%      CONVERT_IPIX2CUBEFACE, by itself, creates a new CONVERT_IPIX2CUBEFACE
or raises the existing
%      singleton*.
%
%      H = CONVERT_IPIX2CUBEFACE returns the handle to a new
CONVERT_IPIX2CUBEFACE or the handle to
%      the existing singleton*.
%
%      CONVERT_IPIX2CUBEFACE('CALLBACK',hObject,eventData,handles,...) calls
the local
%      function named CALLBACK in CONVERT_IPIX2CUBEFACE.M with the given
input arguments.
%
%      CONVERT_IPIX2CUBEFACE('Property','Value',...) creates a new
CONVERT_IPIX2CUBEFACE or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before Convert_iPix2CubeFace_OpeningFcn gets
called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to Convert_iPix2CubeFace_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
%*************************************************************************
****
% here is the list of handles USER variables:
% handles.user.suffix
% handles.user.outputDirectory
% handles.user.inputDirectory
% handles.user.inputFileName
% handles.user.stripImage
% handles.user.inputFile
% handles.user.inputPathFileName
% handles.user.outPut1
%*************************************************************************
****

% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Convert_iPix2CubeFace

% Last Modified by GUIDE v2.5 24-Oct-2012 12:37:43
```

```
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Convert_iPix2CubeFace_OpeningFcn, ...
                   'gui_OutputFcn',  @Convert_iPix2CubeFace_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Convert_iPix2CubeFace is made visible.
function Convert_iPix2CubeFace_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Convert_iPix2CubeFace (see VARARGIN)

% Choose default command line output for Convert_iPix2CubeFace
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Convert_iPix2CubeFace wait for user response (see UIRESUME)
% uiwait(handles.figure1);
%Initialize GUI variables
handles.user.suffix='_cubeFace';
set(handles.DisplayText_outputSuffix,'String',handles.user.suffix);
currentDirectory=pwd;
[n,m]=size(currentDirectory);
handles.user.outputDirectory=currentDirectory;
displayDirectory=horzcat('...',currentDirectory(m-20:end));
set(handles.text_OutputDirectory,'String',displayDirectory);

handles.user.inputDirectory=currentDirectory;
set(handles.text_inputDirectory,'String',displayDirectory);

handles.user.inputFileName=horzcat('inputStripName','.jpg');
set(handles.edit_inputFileName,'String',handles.user.inputFileName);
guidata(hObject, handles);


% --- Outputs from this function are returned to the command line.
```

```
function varargout = Convert_iPix2CubeFace_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;



function edit_inputFileName_Callback(hObject, eventdata, handles)
% hObject    handle to edit_inputFileName (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit_inputFileName as text
%        str2double(get(hObject,'String')) returns contents of
edit_inputFileName as a double
user_string = get(hObject,'String');

PathName=handles.user.inputDirectory;
inputFile=horzcat(PathName,'\',user_string);
handles.edit_inputFileName=inputFile;
set(handles.pushbutton_convert,'Enable','on');
guidata(hObject, handles);

% --- Executes during object creation, after setting all properties.
function edit_inputFileName_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit_inputFileName (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton_browse.
function pushbutton_browse_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton_browse (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
start_path=handles.user.inputDirectory;
DefaultName=horzcat(start_path,'\*.jpg');
DialogTitle='Select File for Input Image';
[FileName,PathName,FilterIndex] = uigetfile('*.jpg',DialogTitle,DefaultName);
handles.user.inputDirectory=PathName;
inputFile=horzcat(PathName,'\',FileName);
handles.user.inputPathFileName=inputFile;
handles.user.inputFileName=FileName;
```

```
%load image
stripImage=imread(inputFile);
handles.user.stripImage=stripImage;

% write new path for input file
[n,m]=size(PathName);
displayDirectory=horzcat('...',PathName(m-20:end));
set(handles.text_inputDirectory,'String',displayDirectory);
set(handles.pushbutton_convert,'Enable','on');
guidata(hObject, handles);

%write the file name into the control panel
set(handles.edit_inputFileName,'String',FileName);

% --- Executes on button press in pushbutton_changeOutputDirectory.
function pushbutton_changeOutputDirectory_Callback(hObject, eventdata,
handles)
% hObject    handle to pushbutton_changeOutputDirectory (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
start_path=handles.user.outputDirectory;
folder_name = uigetdir(start_path,'Select Directory for Output');
handles.user.outputDirectory=folder_name;
% cd(folder_name);
[n,m]=size(folder_name);
displayDirectory=horzcat('...',folder_name(m-20:end));
set(handles.text_OutputDirectory,'String',displayDirectory);
guidata(hObject, handles);

function DisplayText_outputSuffix_Callback(hObject, eventdata, handles)
% hObject    handle to DisplayText_outputSuffix (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of DisplayText_outputSuffix
as text
%        str2double(get(hObject,'String')) returns contents of
DisplayText_outputSuffix as a double
user_string = get(hObject,'String');
handles.user.suffix=user_string;
guidata(hObject, handles);

% --- Executes during object creation, after setting all properties.
function DisplayText_outputSuffix_CreateFcn(hObject, eventdata, handles)
% hObject    handle to DisplayText_outputSuffix (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
% --- Executes on button press in pushbutton_convert.
function pushbutton_convert_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton_convert (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.pushbutton_convert,'Enable','off');
fileName=handles.user.inputFileName;
inputFile=imread(fileName);
handles.user.inputFile=inputFile;
[inputN, inputM]=size(inputFile);
outPut1=inputFile(:,1:inputN,:);
outPut2=inputFile(:,inputN+1:2*inputN,:);
outPut3=inputFile(:,2*inputN+1:3*inputN,:);
outPut4=inputFile(:,3*inputN+1:4*inputN,:);
outPut5=inputFile(:,4*inputN+1:5*inputN,:);
outPut6=inputFile(:,5*inputN+1:6*inputN,:);

handles.user.outPut1=outPut1;
handles.user.outPut2=outPut2;
handles.user.outPut3=outPut3;
handles.user.outPut4=outPut4;
handles.user.outPut5=outPut5;
handles.user.outPut6=outPut6;

ImageOut1=outPut1;
ImageOut2=rotateSquare(outPut2,+1);
ImageOut3=rotateSquare(rotateSquare(outPut3,-1),-1);
ImageOut4=rotateSquare(outPut4,-1);
ImageOut5=outPut5; %rotateSquare(outPut5,-1);
ImageOut6=outPut6;

inputFileDesignation=handles.user.inputFileName;
outputFileNameWithoutExtension=inputFileDesignation(1:end-4);

outName1=horzcat(handles.user.outputDirectory,'\',outputFileNameWithoutExtens
ion,handles.user.suffix,'_A.jpg');
imwrite(ImageOut1,outName1,'jpg','Quality',100);

outName2=horzcat(handles.user.outputDirectory,'\',outputFileNameWithoutExtens
ion,handles.user.suffix,'_B.jpg');
imwrite(ImageOut2,outName2,'jpg','Quality',100);

outName3=horzcat(handles.user.outputDirectory,'\',outputFileNameWithoutExtens
ion,handles.user.suffix,'_E.jpg');
imwrite(ImageOut3,outName3,'jpg','Quality',100);

outName4=horzcat(handles.user.outputDirectory,'\',outputFileNameWithoutExtens
ion,handles.user.suffix,'_D.jpg');
imwrite(ImageOut4,outName4,'jpg','Quality',100);

outName5=horzcat(handles.user.outputDirectory,'\',outputFileNameWithoutExtens
ion,handles.user.suffix,'_F.jpg');
imwrite(ImageOut5,outName5,'jpg','Quality',100);
```

```
outName6=horzcat(handles.user.outputDirectory,'\',outputFileNameWithoutExtens
ion,handles.user.suffix,'_C.jpg');
imwrite(ImageOut6,outName6,'jpg','Quality',100);


guidata(hObject, handles);


% --- Executes on button press in pushbutton_changeInputDirectory.
function pushbutton_changeInputDirectory_Callback(hObject, eventdata,
handles)
% hObject    handle to pushbutton_changeInputDirectory (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
start_path=handles.user.inputDirectory;
folder_name = uigetdir(start_path,'Select Directory for Input');
handles.user.inputDirectory=folder_name;
% cd(folder_name);
[n,m]=size(folder_name);
displayDirectory=horzcat('...',folder_name(m-20:end));
set(handles.text_inputDirectory,'String',displayDirectory);
guidata(hObject, handles);

function ImageOut=rotateSquare(imageIn,PorM)
%
% Author: Geoff Forden, geforde@sandia.gov
%
%this routine takes a square image and rotates it by either 0, +90, or -90
%degrees for use in PanoTour Pro
%
ImageOut=imageIn;
[n,m,mm]=size(imageIn);
if( n ~= m )
    return;
end

if(PorM == 0)
    ImageOut=imageIn;
end

if(PorM == +1)
    % rotate the image in the Plus direction
    for i=1:n
        for j=1:m
            inew=m-j+1;
            ImageOut(j,i,:)=imageIn(i,inew,:);
        end
    end
end

if(PorM == -1)
        % rotate the image in the Minus direction
    for i=1:n
        for j=1:m
```

```
            ImageOut(j,i,:)=imageIn(m-i+1,j,:);
        end
    end
end
```

DISTRIBUTION

10	MS1373	Geoffrey Forden	6821
1	MS0899	Technical Library	9536 (electronic copy)
1	MS0161	Legal Technology Transfer Center	11500

Sandia National Laboratories