Option Explicit

'This set of macros works for any camera - qImaging or not.

'The resizing code works regardless of any beginning difference in proportion.

'Wingmachine 1: Values between 10,000 and 19,999

'Wingmachine 2: Values between 20,000 and 29,000

'Wingmachine 3: Values between 30,000 and 39,000

'Set machine specific information here:

Public Const minVal = 20001

Public Const maxVal = 29999

Public Const asciiPrefix = "WM2"

Function mySplit(txt As String)

Dim i As Long, a() As String, n As Long

ReDim a(1 To Len(txt))

For i = 1 To Len(txt)

If (Mid$(txt, i, 1) = ",") Then

n = n + 1 : a(n) = Mid$(txt, i, 1)

Debug.Print n

Debug.Print a(n)

End If

Next

ReDim Preserve a(1 To n)

mySplit = a

End Function

'This file is written for wingmachine 2

Function getapoint (x As Single,y As Single)

Dim points%

IpMeasShow(1)

'Clear any measurements from previous images

IpMeasTag(MEAS\_ALL, 1)

IpMeasDelete(MEAS\_TAG)

IpMeasTool(MEAS\_POINT)

IpLocZoomShow(1)

IpLocZoomMove(1057, 75) 'Moves local zoom window to top right of screen, next to image window. GC

IpLocZoomSet(IP\_LZ\_ZOOM, 200)

IpLocZoomSet(IP\_LZ\_CROSS, 1)

points = 0

While points<1

IpMeasGet(GETNUMOBJ,0,points)

Wend

'Open the data collector window

IpDcShow(4)

IpDcShow(3)

'Select which measurements to take

IpDcSelect("Image", "Name", 0)

IpDcSelect("Measurements", "MDATA\_POS", 0)

IpDcSelect("Measurements", "MDATA\_POSY", 0)

IpDcShow(1)

'Clear any measurements from previous images

IpDcUpdate(DC\_RESET)

IpDcSet(DC\_AUTO, 0)

'Allow data collector to collect measurements from current image

IpDcUpdate(DC\_FETCH)

'Save measurements to variable names

IpMeasShow(0)

ipdcset(dc\_col, 1)

ipdcget(dc\_data,1,x)

ipdcset(dc\_col, 2)

ipdcget(dc\_data,1,y)

IpDcShow(0)

End Function

Function calibrate(imagepath$,mmperpix As Single,calSet%)

'General calibration routine

Dim calname$,calpath$,dateTime$,unitPerPix As String\*20

calSet=0

Do While calSet=0 'When calSet=0 the calibration has not been set

'Local zoom window

IpLocZoomShow(1)

IpLocZoomSet(IP\_LZ\_ZOOM,200)

IpLocZoomSet(IP\_LZ\_CROSS,1)

'Take calibration image

IpAcqShow(ACQ\_SNAP,1)

IpAcqControl(800,0,IPNULL)

IpAcqShow(ACQ\_LIVE,1)

IpMacroStop("Position calibration scale in the Live Preview window, then click CONTINUE.",0)

IpAcqShow(ACQ\_LIVE,0)

IpAcqSnap(ACQ\_CURRENT)

IpAcqShow(ACQ\_SNAP,0)

'Open calibration dialog

IpSCalShow(1)

IpSCalDestroy()

IpSCalCreate()

IpSCalSetUnitName("millimeter")

IpSCalSetLong(0,SCAL\_ONIMAGE\_COLOR,RGB(0,236,255))

IpMacroStop("1. Click on the IMAGE button under PIXELS/UNIT, 2. Stretch the blue line over the scale bar, 3. Click OK in SCALING window, 4. Click CONTINUE here.",0)

IpCalGet("sYUnitPerPix",unitPerPix)

mmperpix=Val(unitPerPix)

IpSCalShow(0)

calSet=IpMacroStop("Did you successfully get the scale?",MS\_MODAL+MS\_YESNOCAN+MS\_QUEST)

'Check to see if scale was set correctly

If calSet=1 Then

If Val(unitPerPix)=1 Then

IpMacroStop("No you didn't! You forgot to click OK before Continue. Click Continue here to recalibrate",1)

calSet=0

ElseIf Val(unitPerPix)=0.1 Then

IpMacroStop("No you didn't! You forgot to position the line over the scale bar. Click Continue here to recalibrate",1)

calSet=0

Else

'Save the calibration image

dateTime=Replace(Format(Now,"m.d.yy/h.nn.AM/PM"),"/","\_")

calname="cal"+dateTime+".tif"

calname=IpTrim(calname)

calpath=imagepath+"\"+calname

IpWsSaveAs(calpath,"TIF")

End If

End If

'Reset units so landmarks are recorded in pixels

IpSCalSetUnit(1,1)

IpLocZoomShow(0)

IpAppCloseAll()

Loop

End Function

Function landmarks(lmSet%,nLm%,tempFile$,XY$)

'A modified version of the getapoint function for collecting multiple landmarks

'Jason Cassara 2/8/16

Dim i%,lmDrawn%,cirSz%

Dim obj(99) As Single,x(99) As Single,y(99) As Single

cirSz=12 'controls size of annotated circles on image

Do While lmSet=0

XY=""

For i=1 To nLm

lmDrawn=0 'reset to 0

IpLocZoomShow(1) 'Local zoom window

IpLocZoomSet(IP\_LZ\_ZOOM,200)

IpLocZoomSet(IP\_LZ\_CROSS,1)

IpMeasShow(1) 'Open measurement window

IpMeasDelete(MEAS\_ALL) 'Delete previous measurements

IpMeasTool(MEAS\_POINT) 'Select point measurement tool

'This loop prevents the function from moving on until a point has been drawn

While lmDrawn<1

IpMeasGet(GETNUMOBJ,0,lmDrawn)

Wend

IpDcShow(3) 'Open the data collector window

IpDcUpdate(DC\_RESET) 'Clear data collector

'Select which measurements to take

IpDcSelect("Image","Name",0)

IpDcSelect("Measurements","MDATA\_POS",0)

IpDcSelect("Measurements","MDATA\_POSY",0)

'Allow data collector to collect measurements from current image

IpDcSet(DC\_AUTO,0)

IpDcUpdate(DC\_FETCH)

'Save measurements to variable names

IpDcSet(DC\_COL,1)

IpDcGet(DC\_DATA,1,x(i))

IpDcSet(DC\_COL,2)

IpDcGet(DC\_DATA,1,y(i))

'Concatenate all coordinates as a string

XY = XY + Str$(x(i)) + Str$(y(i))

'Draw circle on image

obj(i)=IpAnCreateObj(GO\_OBJ\_ELLIPSE)

IpAnMove(0,x(i)-cirSz,y(i)-cirSz)

IpAnMove(5,x(i)+cirSz,y(i)+cirSz)

'Close windows

IpLocZoomShow(0)

IpMeasShow(0)

IpDcShow(0)

Next i

lmSet=IpmacroStop("Are the points entered correctly for this image?",MS\_MODAL+MS\_YESNOCAN+MS\_QUEST)

If lmSet=2 Then 'User pressed CANCEL

IpAppCloseAll()

Exit Function

End If

'Save image as temporary file, close it, then open it to force point number labels to revert

IpWsSaveAs(tempFile,"TIF")

IpDocClose()

IpWsLoad(tempFile,"TIF")

'Delete circles from image

For i=1 To nLm

ipanactivateobjID(obj(i))

IpAnDeleteObj()

Next i

Loop

End Function

Function dialogBoxx(nFields%,captions$,defaults$,dlgCancel%)

'A generic function for creating a dialog box with a specified number/list of fields

'nFields must be an integer from 1-20

'captions and defaults must each be a single string containing the desired values separated by commas. No commas are needed after the last value.

'The number of comma-separated values in captions and defaults must each = nFields

On Error GoTo ErrorHandler

Dim caps() As String

caps=mySplit(captions)

Debug.Print captions

Debug.Print caps(0)

'ReDim Preserve caps(19) 'Make an array of strings holding captions, plus blanks to fill up 20 items

Dim defs() As String

defs=mySplit(defaults)

'ReDim Preserve defs(19) 'Make an array of strings holding default values, plus blanks to fill up 20 items

Dim ar(19) As Integer 'Make an index array

Dim i%

dlgCancel=0

For i=0 To nFields-1 'Set the first nFields values in the array as 0 through nFields

ar(i)=i

' If dlgCancel<>2 Then 'Test for matching numbers of varnames, captions, defaults, and nFields. This block can be deleted if not needed, but may be useful for debugging.

' If defs(i)="" Or caps(i)="" Then

' dlgCancel=IpMacroStop("Number of fields and captions or default values do"+Chr(13)+"not match. Press OK to exit or Cancel to attempt to"+Chr(13)+"proceed anyway (Not recommended).",MS\_MODAL+MS\_OKCAN+MS\_EXCLAM)

' End If

' If dlgCancel=1 Then

' IpAppCloseAll()

' Exit Function

' End If

' End If

Next

For i=nFields To 19 'Set remaining values in the array to 100. This will move the unwanted fields outside of the visible portion of the dialog box

ar(i)=100

If dlgCancel<>2 Then 'Test for matching numbers of varnames, captions, defaults, and nFields. This block can be deleted if not needed, but may be useful for debugging.

If defs(i)<>"" Or caps(i)<>"" Then

dlgCancel=IpMacroStop("Number of fields and captions or default values do"+Chr(13)+"not match. Press OK to exit or Cancel to attempt to" \_

+Chr(13)+"proceed anyway (Not recommended).",MS\_MODAL+MS\_OKCAN+MS\_EXCLAM)

End If

If dlgCancel=1 Then

IpAppCloseAll()

Exit Function

End If

End If

Next

Begin Dialog UserDialog 400,(nFields+1)\*28+40,"Image information" 'Width,Height,Title

TextBox 180,21+(ar(0))\*28,180,21,.TextBoxA 'X,Y,Width,Height,Name

Text 30,28+(ar(0))\*28,130,14,caps(0),.TextA 'X,Y,Width,Height,Caption,Name

TextBox 180,21+(ar(1))\*28,180,21,.TextBoxB

Text 30,28+(ar(1))\*28,130,14,caps(1),.TextB

TextBox 180,21+(ar(2))\*28,180,21,.TextBoxC

Text 30,28+(ar(2))\*28,130,14,caps(2),.TextC

TextBox 180,21+(ar(3))\*28,180,21,.TextBoxD

Text 30,28+(ar(3))\*28,130,14,caps(3),.TextD

TextBox 180,21+(ar(4))\*28,180,21,.TextBoxE

Text 30,28+(ar(4))\*28,130,14,caps(4),.TextE

TextBox 180,21+(ar(5))\*28,180,21,.TextBoxF

Text 30,28+(ar(5))\*28,130,14,caps(5),.TextF

TextBox 180,21+(ar(6))\*28,180,21,.TextBoxG

Text 30,28+(ar(6))\*28,130,14,caps(6),.TextG

TextBox 180,21+(ar(7))\*28,180,21,.TextBoxH

Text 30,28+(ar(7))\*28,130,14,caps(7),.TextH

TextBox 180,21+(ar(8))\*28,180,21,.TextBoxI

Text 30,28+(ar(8))\*28,130,14,caps(8),.TextI

TextBox 180,21+(ar(9))\*28,180,21,.TextBoxJ

Text 30,28+(ar(9))\*28,130,14,caps(9),.TextJ

TextBox 180,21+(ar(10))\*28,180,21,.TextBoxK

Text 30,28+(ar(10))\*28,130,14,caps(10),.TextK

TextBox 180,21+(ar(11))\*28,180,21,.TextBoxL

Text 30,28+(ar(11))\*28,130,14,caps(11),.TextL

TextBox 180,21+(ar(12))\*28,180,21,.TextBoxM

Text 30,28+(ar(12))\*28,130,14,caps(12),.TextM

TextBox 180,21+(ar(13))\*28,180,21,.TextBoxN

Text 30,28+(ar(13))\*28,130,14,caps(13),.TextN

TextBox 180,21+(ar(14))\*28,180,21,.TextBoxO

Text 30,28+(ar(14))\*28,130,14,caps(14),.TextO

TextBox 180,21+(ar(15))\*28,180,21,.TextBoxP

Text 30,28+(ar(15))\*28,130,14,caps(15),.TextP

TextBox 180,21+(ar(16))\*28,180,21,.TextBoxQ

Text 30,28+(ar(16))\*28,130,14,caps(16),.TextQ

TextBox 180,21+(ar(17))\*28,180,21,.TextBoxR

Text 30,28+(ar(17))\*28,130,14,caps(17),.TextR

TextBox 180,21+(ar(18))\*28,180,21,.TextBoxS

Text 30,28+(ar(18))\*28,130,14,caps(18),.TextS

TextBox 180,21+(ar(19))\*28,180,21,.TextBoxT

Text 30,28+(ar(19))\*28,130,14,caps(19),.TextT

OKButton 90,(nFields+1)\*28,100,21

CancelButton 210,(nFields+1)\*28,100,21

End Dialog

Dim dlg As UserDialog 'Populate dialog box with default values

dlg.TextBoxA=defs(0):dlg.TextBoxB=defs(1):dlg.TextBoxC=defs(2):dlg.TextBoxD=defs(3):dlg.TextBoxE=defs(4):

dlg.TextBoxF=defs(5):dlg.TextBoxG=defs(6):dlg.TextBoxH=defs(7):dlg.TextBoxI=defs(8):dlg.TextBoxJ=defs(9):

dlg.TextBoxK=defs(10):dlg.TextBoxL=defs(11):dlg.TextBoxM=defs(12):dlg.TextBoxN=defs(13):dlg.TextBoxO=defs(14):

dlg.TextBoxP=defs(15):dlg.TextBoxQ=defs(16):dlg.TextBoxR=defs(17):dlg.TextBoxS=defs(18):dlg.TextBoxT=defs(19)

Dialog dlg 'Show dialog box

For i=0 To nFields-1 'Get values from dialog box

defs(i)=Eval("dlg.TextBox"&Chr(i+65))

defs(i)=IpTrim(defs(i))

Next

ReDim Preserve defs(nFields-1)

'defaults=Join(defs,",")

Exit Function

ErrorHandler:

dlgCancel=1

Exit Function

End Function

'Universally calibrated - reads innate picture size and reduces everything accordingly to produce 640x480 images 7/3/09 SJS

'Uses basic dialog

'Added checking for whether file exists, wingmachine numbers, and pic ID numbers 10/23/10 SJS

'Also went through and made sure the tabbing fits proper programming style, so it's easier to ID loops.

Sub BigWingZoom()

Dim comments$, author$, namepath$, calpath$, calname$, dateandtime$, flySex$, imagename$, imagepath$, asciiname$, genotypename$, imagenumber$, message$

Dim imagenum%, outfile$, iname$, collect%, image%, scale%, goodpt%, ampm%

Dim points%, obj1%, obj2%

Dim x(2) As Single, y(2) As Single, mmperpix As Single

Dim xto As Single, yto As Single

Dim nocomment As Boolean

Dim dontWrite As Boolean

Dim okayToWrite As Boolean

Dim pixpermm$

Dim unitPerPix As String \*20

Dim picDate As Date

Dim zoomFactor As Long

Dim shrinkFactor As Single

Dim NL$

NL = Chr$(13) +Chr$(10)

' This set of commands changes the appearance of the annotated points.

IpAnSet(GO\_ATTR\_PENWIDTH, 2)

IpAnSet(GO\_ATTR\_USEASDEFAULT, 1)

IpAnSet(GO\_ATTR\_PENCOLOR, 255)

IpAnSet(GO\_ATTR\_USEASDEFAULT, 1)

zoomFactor = 100 'default value

scale = 0

' For debugging scale=1, otherwise set back to 0

Do While scale=0

'When scale = 0 the calibration has not been set

'Taking calibration picture:

IpAcqShow(ACQ\_SNAP,1)

IpAcqControl(800, 0, IPNULL)

IpAcqShow(ACQ\_LIVE, 1)

IpMacroStop("Place a calibration scale in the active video image, then click CONTINUE.",0)

IpAcqShow(ACQ\_LIVE, 0)

IpAcqSnap(ACQ\_CURRENT)

IpAcqShow(ACQ\_SNAP, 0)

'Calibrate final image reduction: this is coded so it will work for any camera that has a 4:3 aspect ratio. 7/3/09 SJS

'It is intended to produce images for use in Wings software which requires 640x480 input files. 7/3/09 SJS

Dim picInfo As IPDOCINFO

IpDocGet(GETDOCINFO, DOCSEL\_ACTIVE, picInfo)

shrinkFactor = picInfo.Width/640

Dim szcir As Single

szcir = 3\*shrinkFactor 'This determines the size of the annotated circle on the wings.

'Calculate today's date and time from system info 07/04/09 SJS

dateandtime = "18\_Jun\_79 Mon\_AM"

dateandtime = Replace(Format(Now,"m.d.yy/h.nn.AM/PM"),"/","\_")

'Open calibration dialog

IpSCalShow(1)

IpSCalDestroy()

IpSCalCreate()

IpSCalSetUnitName("millimeter")

imagepath = "ThisIsAGenericFileNameThatIsReallyLongSoTheActualFilenamesCanHaveSuperLength"

' get imagepath from user - JAC 10/10/12

scale = IpStGetString("Enter the path for images. Don't forget to put the backslash \ on the end.", imagepath, 75)

If scale = 0 Then

IpAppCloseAll()

Exit Sub

End If

imagepath = IpTrim(imagepath)

calname = asciiPrefix + "cal" + dateandtime + ".tif"

calname = IpTrim(calname)

calpath = imagepath + calname

'Reset the calibration so that futher measurements are taken in pixels, not millimeters

IpMacroStop("1. Click on the IMAGE button (pixels/unit) in the SPATIAL CALIBRATION window, 2. Position the line to measure 2 mm in the IMAGE window, 3. Click OK in SCALING window, 4. Click CONTINUE here.",0)

IpSCalMove(651, 172)

IpCalGet("sYUnitPerPix", unitPerPix)

IpSCalShow(0)

scale = IpMacroStop("Did you successfully get the scale?",MS\_MODAL+MS\_YESNOCAN+MS\_QUEST)

'Check to see if scale was set correctly - JAC 10/10/12

If scale = 1 Then

If Val(unitPerPix) = 1 Then

IpMacroStop("No you didn't! You forgot to click OK before Continue. Click Continue here to re-calibrate",1)

scale=0

Else

' save the calibration image - JAC 10/10/12

IpWsSaveAs(calpath, "TIF")

End If

End If

' The FWINGS programs locates the coordinates in pixels.

IpSCalSetUnit(1,1)

IpAppCloseAll()

'End calibration collection loop

Loop

If scale=2 Then

'If scale = 2 the user has pressed the cancel button during calibration and the macro stops.

IpAppCloseAll()

Exit Sub

End If

'If scale = 1 the calibration has been set and the program will go on.

'Debugging: Ipmacrostop(unitPerPix, MS\_MODAL+MS\_OKCAN)

'Convert pixels per mm to mms per pix 4/09/08 SJS

mmperpix = Val(unitPerPix)

'Since image is resized later, multiply mmperpix by shrink factor. 7/3/09 SJS

mmperpix = shrinkFactor\*mmperpix

pixpermm = 1/CVar(mmperpix)

'Set starting values for the information to be collected.

imagenum = minVal

imagenumber = Str(minVal)

author = "author"

imagename = "example#####.tif"

asciiname = "example.asc"

flySex = "X"

nocomment = False

dontWrite = False

'SJS 10-23-10: The following boolean is used to determine whether an image file exists before writing.

'This should prevent file overwrites.

'It starts out false until we check and make sure it's okay to write to a given filename

okayToWrite=False

'Collect is the main looper for taking multiple images.

'While Collect=1, another image will be taken.

collect = 1

Do While collect = 1

comments = " "

'If image = 0, an image will be snapped. Image will be set to 1 as

'soon as a good shot is taken.

Do While image = 0

x(1) = 0

x(2) = 0

y(1) = 0

y(2) = 0

nocomment = False

dontWrite = False

IpAcqShow(ACQ\_SNAP,1)

IpAcqControl(800, 0, IPNULL)

IpAcqShow(ACQ\_LIVE, 1)

'not needed on WM2 IpAcqControl(10, 60, IPNULL) 'I added this line to change the brightness of images from 50 to 60 - JAC

IpMacroStop("Arrange wing in the active video image, then click CONTINUE to go on.",0)

' IpAcqControl(801, 1, IPNULL)

IpAcqShow(ACQ\_LIVE, 0)

IpAcqSnap(ACQ\_CURRENT)

IpAcqShow(ACQ\_SNAP, 0)

image = IpMacroStop("Did you get a good image?",MS\_MODAL+MS\_YESNOCAN+MS\_QUEST)

Loop

'If image = 2 the user has pressed the cancel button and the macro is stopped.

If image = 2 Then

collect = IpMacroStop("Do you want to acquire another image?",MS\_MODAL+MS\_YESNO+MS\_QUEST)

End If

If collect = 1 Then

'We must reset image to 0 for the next time around.

image = 0

'The following lines are not needed any more, have been left in for reference.

'IpWsZoom(100)

'Adjust the contrast

' IpHstEqualize(EQ\_BESTFIT)

' IpLutApply()

'If nopoints = 0, the user has not yet entered any FWINGS coordinates

'Find default zoom factor - adaptable to any size camera 7/3/09 SJS

IpDocGet(INF\_ZOOMFACTOR, 0, zoomFactor)

'Get coords for FWINGS

goodpt = 0

Do While goodpt = 0

'IpOutputShow(1)

'Click to zoom in to accurately place point 1

getapoint(x(1),y(1))

IpWsZoom(200)

If (x(1)<150) Then xto=0 Else xto=x(1)-50

If (y(1)<100) Then yto=0 Else yto=y(1)-50

IpWsPan(xto,yto)

'Place point 1

getapoint(x(1),y(1))

'Reset zoom to default level to place point 2 7/3/09 SJS

IpWsZoom(zoomFactor)

'IpWsZoom(50)

'Draw point 1

obj1 = IpAnCreateObj(GO\_OBJ\_ELLIPSE)

IpAnMove(0, x(1)-szcir, y(1)-szcir)

IpAnMove(5, x(1)+szcir, y(1)+szcir)

'Place point 2

getapoint(x(2),y(2))

IpMeasShow(1)

IpMeasTag(MEAS\_ALL, 1)

IpMeasDelete(MEAS\_TAG)

'Draw point 2

obj2 = IpAnCreateObj(GO\_OBJ\_ELLIPSE)

IpAnMove(0, x(2)-szcir, y(2)-szcir)

IpAnMove(5, x(2)+szcir, y(2)+szcir)

goodpt = IpMacroStop("Are the points entered correctly for this image?",MS\_MODAL+MS\_YESNO+MS\_QUEST)

If goodpt=0 Then

IpAnActivateObjID(obj1)

IpAnDeleteObj()

IpAnActivateObjID(obj2)

IpAnDeleteObj()

End If

'End coord placing loop

Loop

'Shrink picture and XY values of points

'Remember, shrink factor describes the difference in size between the original pictures and the 640x480 version.

IpWsScale(640,480,1)

x(1) = x(1)/shrinkFactor

x(2) = x(2)/shrinkFactor

y(1) = y(1)/shrinkFactor

y(2) = y(2)/shrinkFactor

'Calculate today's date and time from system info 07/04/09 SJS

dateandtime = Replace(Format(Now,"m/d/yy h:nnAM/PM"),"/","-")

'Define the custom dialog box for collecting user-entered information

'Programming note: In future, create new dialog boxes by pressing the "edit user dialog" button on the macro toolbar.

Begin Dialog UserDialog 400,381,"Image information" ' %GRID:10,7,1,1

TextBox 180,21,180,21,.TextBox1

TextBox 180,49,180,21,.TextBox9

TextBox 180,77,180,21,.TextBox2

TextBox 180,105,180,21,.TextBox3

TextBox 180,133,180,21,.TextBox6

TextBox 180,161,180,21,.TextBox4

TextBox 180,189,180,21,.TextBox5

TextBox 180,217,180,21,.TextBox7

TextBox 180,245,180,21,.TextBox8

TextBox 180,283,180,21,.TextBox10

Text 30,28,140,14,"Your name",.Text1

Text 30,84,140,14,"Today's date and time",.Text2

Text 30,112,140,14,"Name of first image",.Text3

Text 30,140,130,14,"ID number of image",.Text6

Text 30,168,140,14,"Path for images",.Text4

Text 30,196,140,14,"Name of ascii file",.Text5

Text 30,224,130,14,"Genotype of image",.Text7

Text 30,252,130,14,"Comments",.Text8

Text 30,56,130,14,"Calibration scale",.Text9

Text 30,290,130,14,"Sex",.Text10

OKButton 70,321,100,21

CancelButton 190,321,100,21

End Dialog

Dim dlg As UserDialog

'Assign default values to the textboxes of the custom dialog box

dlg.textbox1 = author

dlg.textbox9 = pixpermm

dlg.textbox2 = dateandtime

dlg.textbox3 = imagename

dlg.textbox4 = imagepath

dlg.textbox5 = asciiname

dlg.textbox6 = imagenumber

dlg.textbox8 = comments

dlg.textBox10 = flySex

'SJS 10-23-10: The following do loop encloses the dialog showing and reading part.

'It is designed to check some entered data and force reentry if incorrect.

Do While(Not(okayToWrite))

'Show the custom dialog box

Dialog dlg

'read in user values entered into the textboxes

author = dlg.textbox1

dateandtime = dlg.textbox2

imagename = dlg.textbox3

imagepath = dlg.textbox4

asciiname = dlg.textbox5

imagenumber = dlg.textbox6

genotypename = dlg.textbox7

comments = dlg.textbox8

pixpermm = dlg.textbox9

flySex = dlg.textbox10

'trim the trailing spaces on the string variables

author = IpTrim(author)

dateandtime = IpTrim(dateandtime)

imagename = IpTrim(imagename)

imagepath = IpTrim(imagepath)

asciiname = IpTrim(asciiname)

genotypename = IpTrim(genotypename)

imagenum = CInt(imagenumber)

comments = IpTrim(comments)

pixpermm = IpTrim(pixpermm)

flySex = IpTrim(flySex)

'DEBUG: Ipmacrostop(genotypename, MS\_MODAL+MS\_OKCAN) 'testing genotype

If comments = "" Then

nocomment = True

End If

If comments = "x" Then

dontWrite = True

End If

'Create new image name

'10-23-10 SJS: The starting length of iname is what determines the final length of the output image name.

'That is really, really dumb that I didn't figure this out sooner.

'There must be a more elegant solution than the one I have used. But at least it works.

iname = "ThisIsAGenericFileNameThatIsReallyLongSoTheActualFilenamesCanHaveSuperLength"

IpStAutoName(imagename,imagenum,iname)

iname = IpTrim(iname)

'NOTE: to concatenate properly (linking imagepath and iname to make namepath), the variables must be trimmed first (see above).

namepath = imagepath + iname

'Check if the file exists, and if the numbers are within the min/max range for this machine.

'Note: Dir$() returns the filename if a file exists, returns blank string "" if it does not.

'If so, it is okay to go on.

If Dir$(namepath)="" Then

okayToWrite=True

'DEBUG: MsgBox("File found is named: " + Dir$(namepath))

Else

IpMacroStop(("Warning! The file " + iname + " already exists in that directory. Please check your prefix, directory, and image number."),MS\_MODAL+MS\_STOP)

'DEBUG: MsgBox("File found is named: " + Dir$(namepath))

End If

If imagenum<minVal Then

IpMacroStop(("Warning! Image number must be between " + Str(minVal) + " and " + Str(maxVal) + ". Please choose another number."),MS\_MODAL+MS\_STOP)

okayToWrite=False

ElseIf imagenum>maxVal Then

IpMacroStop(("Warning! Image number must be between " + Str(minVal) + " and " + Str(maxVal) + ". Please choose another number."),MS\_MODAL+MS\_STOP)

okayToWrite=False

End If

If genotypename = "" Then

IpMacroStop(("Warning! You must include a genotype name."),MS\_MODAL+MS\_STOP)

okayToWrite=False

End If

'End 'entering info and checking if file exists loop'

'Check if directory exists too.

Loop

'Once the loop has exited, it means that the directory exists and the file doesn't, and we can save now.

If dontWrite = False Then

'Define and open the output file

'NOTE: to concatenate properly (linking imagepath and asciiname to make outfile), the variables must be trimmed first (see above).

'10-23-10 SJS: Adding a prefix to ascii file that specifies which wingmachine it came from.

outfile = imagepath+asciiPrefix+asciiname

'DEBUG: ret=MsgBox(outfile)

Open outfile For Append As #1

IpWsSaveAs(namepath, "TIF")

'Also, reset the condition of okayToWrite for the next image name.

okayToWrite=False

'Print all info to the output file

Print #1, iname + Str$(x(1)) + Str$(y(1)) + Str$(x(2)) + Str$(y(2)) + " " + author + " " + dateandtime + " "+ genotypename + " " + flySex + " " + Str$(mmperpix) + " " + Str$(1) + " " + Str$(1)

'Print comment line if necessary

If nocomment = False Then

Print #1, comments

End If

'Increment id# so that the next image will have a different id name associated with it

imagenum = imagenum + 1

imagenumber = Str$(imagenum)

'Ask the user if there is another image to be acquired

'Note: this has been diabled. It's faster to just assume the answer is yes, because it nearly always is.

'The user can cancel after they take an image and it asks about a retake. The cancel button there still works.

'Collect = IpmacroStop("Do you want to acquire another image?",MS\_MODAL+MS\_YESNO+MS\_QUEST)

'Close current image and output file

Close #1

End If

End If

IpAppCloseAll()

'End loop that lets you take multiple pictures in one session

Loop

End Sub

'Made by Gabriel Calvin, June 2016

'Modified version of FlexWingZoom (which is a modified BigWingZoom) to take images of wings and legs, and attach landmarks.

'Image size produced is unspecific.

'Uses basic dialog

'Contains four locations to change the exposure value, based on the camera being used; two in the calibrate module, and two in the collect module

'The QCam will need these values to be operational, while the Lumenera camera has them disabled.

Sub WingLeg()

Dim comments$, author$, namepath$, dateandtime$, flySex$, imagename$, imagepath$, asciiname$, genotypename$, imagenumber$

Dim block$, transfer$, replicate$, mife$, gene$, driver$, direction$, message$

Dim imagenum%, outfile$, collect%, image%, scale%, goodptwing%, goodptlegs%, ampm%, iname$

Dim calname$, calpath$

Dim points%, obj1%, obj2%, obj3%, obj4%, obj5%

Dim x(5) As Single, y(5) As Single, mmperpix As Single

Dim nocomment As Boolean

Dim dontWrite As Boolean

Dim okayToWrite As Boolean

Dim debuggingmode As Boolean

Dim pathacquire%, okToMakeDir As Boolean

Dim pixpermm$

Dim unitPerPix As String \*20

Dim picDate As Date

Dim exposure As Double 'Needs to be declared as a double. GC

'Temp enabled. Closes any open windows from any previously-running macro. GC

IpAppCloseAll()

IpSCalShow(0)

IpLocZoomShow(0)

IpAcqShow(ACQ\_LIVE, 0)

IpAcqShow(ACQ\_SNAP, 0)

'Set starting values for the information to be collected.

imagenum = minVal

imagenumber = Str(minVal)

author = "AnonymousHouligan"

dateandtime = "22\_Feb\_05 Mon\_PM"

imagename = "example\_.tif"

asciiname = "example.asc"

flySex = "X"

nocomment = False

dontWrite = False

'oktomakedir is a boolean which determines if a directory exists or not. GC 2016.6.22

'This allows users to create new directories at the workspace, instead of manually.

'It begins as False until checked through the pathacquire loop.

okToMakeDir = False

'SJS 10-23-10: The following boolean is used to determine whether an image file exists before writing.

'This should prevent file overwrites.

'It starts out false until we check and make sure it's okay to write to a given filename

okayToWrite=False

' This set of commands changes the appearance of the annotated points.

IpAnSet(GO\_ATTR\_PENWIDTH, 2)

IpAnSet(GO\_ATTR\_USEASDEFAULT, 1)

'IpAnSet(GO\_ATTR\_PENCOLOR, 100)

IpAnSet(GO\_ATTR\_USEASDEFAULT, 1)

'Debugging mode. Set debuggingmode = True to initiate If statement.

debuggingmode = False

If debuggingmode = True Then

scale = 1

mmperpix = 1

End If

'The code for making new directories. GC 2016.6.22

'Input a string variable.

'Surround in if-statement that checks if the directory exists, and if not, makes the new one.

'If the directory does exist, the user is prompted whether they would like to use that directory.

'Still doesn't contain robust code to account for nested directory creation.

Do

'imagepath = "Z:\DGRP\Wings\_Legs"

imagepath = Space(255) 'imagepath must have a value prior to IpStGetStr, otherwise it returns a null string. GC

pathacquire = IpStGetString("Enter the filepath for images.", imagepath, 75)

imagepath = IpTrim(imagepath)

'Debug.Print(imagepath + "&&&&Filepath should go here.")

If pathacquire = 1 Then

If imagepath = "" Or IsNull(imagepath) Then 'This is if the user didn't enter a value at the prompt, and simply clicked OK. GC

IpMacroStop("No filepath was given. Enter a filepath in the prompt.", 0)

GoTo loopToMakeDir

End If

If Dir(imagepath, vbDirectory) = "" Then 'This evaluates if the string imagepath is a directory. GC

okToMakeDir = True

MkDir(imagepath)

MkDir(imagepath + "\scale")

IpMacroStop("Created new directory: " + imagepath, 0)

Else

If IpMacroStop("Directory already exists. Use this directory?", MS\_MODAL+MS\_YESNO) = 1 Then

okToMakeDir = True 'Doesn't make a new directory. Overrides the loop to continue. GC

Else

GoTo loopToMakeDir

End If

End If

Else

GoTo exitmacro

End If

loopToMakeDir:

'End filepath acquisition loop.

Loop Until okToMakeDir = True

'Opens a live window so user can set the appropriate zoom level

exposure = 50.000000

'IpAcqControl(84, 0, exposure) 'Sets exposure time for preview to 50ms. Disabled on Lumenera device. GC 2016.11.7

IpAcqShow(ACQ\_LIVE, 1)

IpMacroStop("Center the fly in the live window. Adjust scope zoom to the appropriate level, such that the tips of the wing and legs are fully in the frame, slightly away from the edges. Click CONTINUE when ready to proceed to calibration routine.", 0)

Do While scale=0

'When scale = 0 the calibration has not been set

'These commands 1) open the live preview; and 2) adjust exposure. Then calibration pathway initiates. GC

exposure = 50.000000

'IpAcqControl(84, 0, exposure) 'Sets exposure time for preview to 50ms

IpAcqShow(ACQ\_LIVE, 1)

'IpAcqControl(88, 1, IPNULL) Temp disabled. GC

'ACQCMD\_AUTOEXPOSURE

'Taking calibration picture:

IpMacroStop("Place a calibration scale in the active video image, then click CONTINUE when ready to snap.",0)

exposure = 500.000000

'IpAcqControl(84, 1, exposure) 'Sets exposure time for acquisition to 500ms. Disabled on Lumenera device. GC 2016.11.7

IpAcqSnap(ACQ\_CURRENT)

IpAcqShow(ACQ\_LIVE,0)

IpWsZoom(50) 'Set image window to 50% zoom

IpDocMaximize() 'Maximize image window

IpDocSize(1053, 775) 'Image window is this large

Dim szcir As Single

szcir = 5 'This determines the size of the annotated circle on the wings.

'Open calibration dialog

IpSCalShow(1)

IpSCalMove(1057, 450) 'Moves calibration window to bottom-right of screen, next to image window

IpSCalDestroy()

IpSCalCreate()

IpSCalSetUnitName("millimeter")

IpLocZoomShow(1)

IpLocZoomSet(IP\_LZ\_ZOOM, 200)

IpLocZoomSet(IP\_LZ\_CROSS, 1)

IpLocZoomMove(1057, 75) 'Moves local zoom window to top right of screen, next to image window

IpMacroStop("1. Click on the IMAGE button (pixels/unit) in the SPATIAL CALIBRATION window, 2. Position the line to measure 2 mm in the IMAGE window, 3. Enter a value of 2 in SCALING window, 4) Click CONTINUE",0)

IpCalGet("sYUnitPerPix", unitPerPix)

scale = IpMacroStop("Did you successfully get the scale?",MS\_MODAL+MS\_YESNOCAN+MS\_QUEST)

'Reset the calibration so that futher measurements are taken in pixels, not millimeters

'The WINGS programs locates the coordinates in pixels.

IpSCalSetUnit(1,1)

'Save the calibration image. GC

If scale = 1 Then

dateandtime = Replace(Format(Now,"m.d.yy/h.nn.AM/PM"),"/","\_")

calname = "scale"+dateandtime+".tif"

calname = IpTrim(calname)

calpath = imagepath + "\scale\" + calname

'Debug.Print(calname)

'Debug.Print(calpath)

IpWsSaveAs(calpath,"TIF")

End If

IpSCalShow(0)

IpAppCloseAll()

IpLocZoomShow(0)

'End calibration collection loop

Loop

If scale=2 Then

'If scale = 2 the user has pressed the cancel button during calibration and the macro calls exitmacro.

GoTo exitmacro

End If

'If scale = 1 the calibration has been set and the program will go on.

'Convert pixels per mm to mms per pix 4/09/08 SJS

If debuggingmode = False Then

mmperpix = Val(unitPerPix)

pixpermm = 1/CVar(mmperpix)

End If

'Collect is the main looper for taking multiple images.

'While Collect=1, another image will be taken.

collect = 1

Do While collect = 1

comments = " "

'If image = 0, an image will be snapped. Image will be set to 1 as soon as a good shot is taken.

Do While image = 0

exposure = 50.000000

'IpAcqControl(84, 0, exposure) 'Sets exposure time for preview. Disabled on Lumenera device. GC 2016.11.7

IpAcqShow(ACQ\_LIVE, 1)

'IpAcqControl(88, 1, IPNULL) ' ACQCMD\_AUTOEXPOSURE Temp disabled. GC

x(1) = 0

x(2) = 0

x(3) = 0

x(4) = 0

x(5) = 0

y(1) = 0

y(2) = 0

y(3) = 0

y(4) = 0

y(5) = 0

nocomment = False

dontWrite = False

'Prompts user to continue when ready to snap

IpMacroStop("Arrange the WING near one edge in the live preview, with the tips of LEGS near the other edge, then click CONTINUE when ready to snap.",0)

exposure = 500.000000

'IpAcqControl(84, 1, exposure) 'Sets exposure time for acquisition. Disabled on Lumenera device. GC 2016.11.7

IpAcqSnap(ACQ\_CURRENT)

IpAcqShow(ACQ\_LIVE,0)

'Resizes snap window to 50% size

IpWsZoom(50) 'Set image window to 50% zoom

IpDocMaximize() 'Maximize image window

IpDocSize(1053, 775) 'Image window is this large

image = IpMacroStop("Did you get a good image?",MS\_MODAL+MS\_YESNOCAN+MS\_QUEST)

If image = 0 Then

IpDocClose()

End If

Loop

'If image = 2 the user has pressed the cancel button in the previous prompt, and the macro is stopped.

If image = 2 Then

GoTo exitmacro

End If

If collect = 1 Then

'We must reset image to 0 for the next time around, since from the previous prompt, image = 1

image = 0

'Gets coordinates for wings

goodptwing = 0

Do While goodptwing = 0

'Place point 1

getapoint(x(1),y(1))

'Draw point 1

obj1 = IpAnCreateObj(GO\_OBJ\_ELLIPSE)

IpAnMove(0, x(1)-szcir, y(1)-szcir)

IpAnMove(5, x(1)+szcir, y(1)+szcir)

'Place point 2

getapoint(x(2),y(2))

IpMeasShow(1)

IpMeasTag(MEAS\_ALL, 1)

IpMeasDelete(MEAS\_TAG)

'Draw point 2

obj2 = IpAnCreateObj(GO\_OBJ\_ELLIPSE)

IpAnMove(0, x(2)-szcir, y(2)-szcir)

IpAnMove(5, x(2)+szcir, y(2)+szcir)

'Hides local zoom window

IpLocZoomShow(0)

'User input whether picture is good or not. Clears points if no.

goodptwing = IpMacroStop("Are the wing points entered correctly for this image?",MS\_MODAL+MS\_YESNO+MS\_QUEST)

If goodptwing=0 Then

IpAnActivateObjID(obj1)

IpAnDeleteObj()

IpAnActivateObjID(obj2)

IpAnDeleteObj()

End If

'End coord placing loop

Loop

'Gets coordinates for legs

goodptlegs = 0

Do While goodptlegs = 0

'Place point 3

getapoint(x(3),y(3))

'Draw point 3

obj3 = IpAnCreateObj(GO\_OBJ\_ELLIPSE)

IpAnMove(0, x(3)-szcir, y(3)-szcir)

IpAnMove(5, x(3)+szcir, y(3)+szcir)

'Place point 4

getapoint(x(4),y(4))

IpMeasShow(1)

IpMeasTag(MEAS\_ALL, 1)

IpMeasDelete(MEAS\_TAG)

'Draw point 4

obj4 = IpAnCreateObj(GO\_OBJ\_ELLIPSE)

IpAnMove(0, x(4)-szcir, y(4)-szcir)

IpAnMove(5, x(4)+szcir, y(4)+szcir)

'Place point 5

getapoint(x(5),y(5))

IpMeasShow(1)

IpMeasTag(MEAS\_ALL, 1)

IpMeasDelete(MEAS\_TAG)

'Draw point 5

obj5 = IpAnCreateObj(GO\_OBJ\_ELLIPSE)

IpAnMove(0, x(5)-szcir, y(5)-szcir)

IpAnMove(5, x(5)+szcir, y(5)+szcir)

'Hide local zoom window after placing landmarks

IpLocZoomShow(0)

'User input whether picture is good or not. Clears points if no.

goodptlegs = IpMacroStop("Are the leg points entered correctly for this image?",MS\_MODAL+MS\_YESNO+MS\_QUEST)

If goodptlegs=0 Then

IpAnActivateObjID(obj3)

IpAnDeleteObj()

IpAnActivateObjID(obj4)

IpAnDeleteObj()

IpAnActivateObjID(obj5)

IpAnDeleteObj()

End If

'End coord placing loop

Loop

dialogbox:

'Calculate today's date and time from system info 07/04/09 SJS

dateandtime = Replace(Format(Now,"m/d/yy h:nnAM/PM"),"/","-")

'Define the custom dialog box for collecting user-entered information

'Programming note: In future, create new dialog boxes by pressing the "edit user dialog" button on the macro toolbar.

Begin Dialog UserDialog 400,381,"Image information" ' %GRID:10,7,1,1

TextBox 180,21,180,21,.TextBox1

TextBox 180,49,180,21,.TextBox9

TextBox 180,77,180,21,.TextBox2

TextBox 180,105,180,21,.TextBox3

TextBox 180,133,180,21,.TextBox6

TextBox 180,161,180,21,.TextBox4

TextBox 180,189,180,21,.TextBox5

TextBox 180,217,180,21,.TextBox7

TextBox 180,245,180,21,.TextBox8

TextBox 180,283,180,21,.TextBox10

Text 30,28,140,14,"Your name",.Text1

Text 30,84,140,14,"Today's date and time",.Text2

Text 30,112,140,14,"Name of first image",.Text3

Text 30,140,130,14,"Image Number",.Text6

Text 30,168,140,14,"Image Filepath",.Text4

Text 30,196,140,14,"Name of ascii File",.Text5

Text 30,224,130,14,"Genotype",.Text7

Text 30,252,130,14,"Comments",.Text8

Text 30,56,130,14,"Calibration Scale",.Text9

Text 30,290,130,14,"Sex",.Text10

OKButton 70,321,100,21

'CancelButton 190,321,100,21

End Dialog

'On Cancel GoTo exitmacro

Dim dlg As UserDialog

'Assign default values to the textboxes of the custom dialog box

dlg.textbox1 = author

dlg.textbox9 = pixpermm

dlg.textbox2 = dateandtime

dlg.textbox3 = imagename

dlg.textbox4 = imagepath

dlg.textbox5 = asciiname

dlg.textbox6 = imagenumber

dlg.textbox8 = comments

dlg.textBox10 = flySex

'SJS 10-23-10: The following do loop encloses the dialog showing and reading part.

'It is designed to check some entered data and force reentry if incorrect.

Do While(Not(okayToWrite))

'Show the custom dialog box

Dialog dlg

'read in user values entered into the textboxes

author = dlg.textbox1

dateandtime = dlg.textbox2

imagename = dlg.textbox3

imagepath = dlg.textbox4

asciiname = dlg.textbox5

imagenumber = dlg.textbox6

genotypename = dlg.textbox7

comments = dlg.textbox8

pixpermm = dlg.textbox9

flySex = dlg.textbox10

'trim the trailing spaces on the string variables

author = IpTrim(author)

dateandtime = IpTrim(dateandtime)

imagename = IpTrim(imagename)

imagepath = IpTrim(imagepath)

asciiname = IpTrim(asciiname)

genotypename = IpTrim(genotypename)

imagenum = CInt(imagenumber)

comments = IpTrim(comments)

pixpermm = IpTrim(pixpermm)

flySex = IpTrim(flySex)

'DEBUG: Ipmacrostop(genotypename, MS\_MODAL+MS\_OKCAN) 'testing genotype

If comments = "" Then

nocomment = True

End If

If comments = "x" Then

dontwrite = True

End If

'Create new image name

'iname is originally given a value of 255 spaces, so that it isn't an empty variable.

'255 spaces allows it to accomodate large names (many characters). GC

iname = Space(255)

IpStAutoName(imagename,imagenum,iname)

'To concatenate properly (linking imagepath and iname to make namepath), the variables must be trimmed first.

iname = IpTrim(iname)

namepath = imagepath + "\" + iname

'Check if the file exists, and if the numbers are within the min/max range for this machine.

'Note: Dir$() returns the filename if a file exists, returns blank string "" if it does not.

'If so, it is okay to go on.

If Dir$(namepath)="" Then

okayToWrite=True

'DEBUG: MsgBox("File found is named: " + Dir$(namepath))

Else

If IpMacroStop(("Warning! The file " + iname + " already exists in that directory. Overwrite Y/N?"),MS\_MODAL+MS\_YESNO+MS\_EXCLAM)=1 Then

'If user presses yes, return value will be 1

okayToWrite=True

'DEBUG: MsgBox("File found is named: " + Dir$(namepath))

Else

imagenumber = CStr(imagenum + 1)

GoTo dialogbox

End If

End If

Loop

'Once the loop has exited, it means that the directory exists and the file doesn't, and we can save now.

If dontWrite = False Then

'Define and open the output file

'NOTE: to concatenate properly (linking imagepath and asciiname to make outfile), the variables must be trimmed first (see above).

'10-23-10 SJS: Adding a prefix to ascii file that specifies which wingmachine it came from.

outfile = imagepath + "\" + asciiname

'DEBUG: ret=MsgBox(outfile)

Open outfile For Append As #1

IpWsSaveAs(namepath, "TIF")

'Also, reset the condition of okayToWrite for the next image name.

okayToWrite=False

'Print all info to the output file. Tabs are used to delimit information when importing into Excel.

'The last two variables to be printed, "Str$(1) + Str$(1)" are superfluous. What data do they convey? GC

Print #1, iname + " " + Str$(x(1)) + " " + Str$(y(1)) + " " + Str$(x(2)) + " " + Str$(y(2)) + " " + Str$(x(3)) + " " + Str$(y(3)) + " " + Str$(x(4)) + " " + Str$(y(4)) + " " + Str$(x(5)) + " " + Str$(y(5)) + " " + author + " " + dateandtime + " "+ genotypename + " " + flySex + " " + Str$(mmperpix)

'Print comment line if necessary

'On Error Next GoTo dialogbox

If nocomment = False Then

Print #1, comments

End If

'Increment id# so that the next image will have a different id name associated with it

imagenum = imagenum + 1

imagenumber = CStr(imagenum)

'Close current image and output file

Close #1

'Ask the user if there is another image to be acquired

collect = IpMacroStop("Do you want to acquire another image?",MS\_MODAL+MS\_YESNO+MS\_QUEST)

End If

End If

'End collect

IpAppCloseAll()

'End loop that lets you take multiple pictures in one session

Loop

If collect = 0 Then

GoTo exitmacro

End If

Exit Sub

'exitmacro: place where goto calls. Allows resetting of sub upon cancel clicks. all cancel options call exitmacro. GC 2016.6.15

'IpMacroRun("WingLeg", "") Recalls the macro

exitmacro:

Dim userinput As Single

userinput = IpMacroStop("Exit macro?", MS\_MODAL+MS\_YESNO)

If userinput = 1 Then

IpAppCloseAll()

Exit Sub

Else

IpAppCloseAll()

IpMacroRun("WingLeg", "")

End If

Exit Sub

End Sub

'Made by Luke Jones 10/2/2017

'Modified version of FlexWingZoom (which is a modified BigWingZoom) to take images of wings and legs, and attach 6 landmarks, calculates distance between point 3 and 4 and places it in excel file.

'Image size produced is unspecific.

'Uses basic dialog

'Contains four locations to change the exposure value, based on the camera being used; two in the calibrate module, and two in the collect module

'The QCam will need these values to be operational, while the Lumenera camera has them disabled.

Sub PleioSelect()

Dim comments$, author$, namepath$, dateandtime$, flySex$, imagename$, imagepath$, asciiname$, genotypename$, imagenumber$

Dim block$, transfer$, replicate$, mife$, gene$, driver$, direction$, message$

Dim imagenum%, outfile$, collect%, image%, scale%, goodptwing%, goodptlegs%, ampm%, iname$

Dim calname$, calpath$

Dim points%, obj1%, obj2%, obj3%, obj4%, obj5%, obj6%

Dim x(6) As Single, y(6) As Single, mmperpix As Single

Dim nocomment As Boolean

Dim dontWrite As Boolean

Dim okayToWrite As Boolean

Dim debuggingmode As Boolean

Dim pathacquire%, okToMakeDir As Boolean

Dim pixpermm$

Dim unitPerPix As String \*20

Dim picDate As Date

Dim exposure As Double 'Needs to be declared as a double. GC

'Temp enabled. Closes any open windows from any previously-running macro. GC

IpAppCloseAll()

IpSCalShow(0)

IpLocZoomShow(0)

IpAcqShow(ACQ\_LIVE, 0)

IpAcqShow(ACQ\_SNAP, 0)

'Set starting values for the information to be collected.

imagenum = minVal

imagenumber = Str(minVal)

author = "AnonymousHouligan"

dateandtime = "22\_Feb\_05 Mon\_PM"

imagename = "example\_.tif"

asciiname = "example.asc"

flySex = "X"

nocomment = False

dontWrite = False

'oktomakedir is a boolean which determines if a directory exists or not. GC 2016.6.22

'This allows users to create new directories at the workspace, instead of manually.

'It begins as False until checked through the pathacquire loop.

okToMakeDir = False

'SJS 10-23-10: The following boolean is used to determine whether an image file exists before writing.

'This should prevent file overwrites.

'It starts out false until we check and make sure it's okay to write to a given filename

okayToWrite=False

' This set of commands changes the appearance of the annotated points.

IpAnSet(GO\_ATTR\_PENWIDTH, 2)

IpAnSet(GO\_ATTR\_USEASDEFAULT, 1)

'IpAnSet(GO\_ATTR\_PENCOLOR, 100)

IpAnSet(GO\_ATTR\_USEASDEFAULT, 1)

'Debugging mode. Set debuggingmode = True to initiate If statement.

debuggingmode = False

If debuggingmode = True Then

scale = 1

mmperpix = 1

End If

'The code for making new directories. GC 2016.6.22

'Input a string variable.

'Surround in if-statement that checks if the directory exists, and if not, makes the new one.

'If the directory does exist, the user is prompted whether they would like to use that directory.

'Still doesn't contain robust code to account for nested directory creation.

Do

'imagepath = "Z:\DGRP\Wings\_Legs"

imagepath = Space(255) 'imagepath must have a value prior to IpStGetStr, otherwise it returns a null string. GC

pathacquire = IpStGetString("Enter the filepath for images.", imagepath, 75)

imagepath = IpTrim(imagepath)

'Debug.Print(imagepath + "&&&&Filepath should go here.")

If pathacquire = 1 Then

If imagepath = "" Or IsNull(imagepath) Then 'This is if the user didn't enter a value at the prompt, and simply clicked OK. GC

IpMacroStop("No filepath was given. Enter a filepath in the prompt.", 0)

GoTo loopToMakeDir

End If

If Dir(imagepath, vbDirectory) = "" Then 'This evaluates if the string imagepath is a directory. GC

okToMakeDir = True

MkDir(imagepath)

MkDir(imagepath + "\scale")

IpMacroStop("Created new directory: " + imagepath, 0)

Else

If IpMacroStop("Directory already exists. Use this directory?", MS\_MODAL+MS\_YESNO) = 1 Then

okToMakeDir = True 'Doesn't make a new directory. Overrides the loop to continue. GC

Else

GoTo loopToMakeDir

End If

End If

Else

GoTo exitmacro

End If

loopToMakeDir:

'End filepath acquisition loop.

Loop Until okToMakeDir = True

'Opens a live window so user can set the appropriate zoom level

exposure = 50.000000

'IpAcqControl(84, 0, exposure) 'Sets exposure time for preview to 50ms. Disabled on Lumenera device. GC 2016.11.7

IpAcqShow(ACQ\_LIVE, 1)

IpMacroStop("Center the fly in the live window. Adjust scope zoom to the appropriate level, such that the tips of the wing and legs are fully in the frame, slightly away from the edges. Click CONTINUE when ready to proceed to calibration routine.", 0)

Do While scale=0

'When scale = 0 the calibration has not been set

'These commands 1) open the live preview; and 2) adjust exposure. Then calibration pathway initiates. GC

exposure = 50.000000

'IpAcqControl(84, 0, exposure) 'Sets exposure time for preview to 50ms

IpAcqShow(ACQ\_LIVE, 1)

'IpAcqControl(88, 1, IPNULL) Temp disabled. GC

'ACQCMD\_AUTOEXPOSURE

'Taking calibration picture:

IpMacroStop("Place a calibration scale in the active video image, then click CONTINUE when ready to snap.",0)

exposure = 500.000000

'IpAcqControl(84, 1, exposure) 'Sets exposure time for acquisition to 500ms. Disabled on Lumenera device. GC 2016.11.7

IpAcqSnap(ACQ\_CURRENT)

IpAcqShow(ACQ\_LIVE,0)

IpWsZoom(50) 'Set image window to 50% zoom

IpDocMaximize() 'Maximize image window

IpDocSize(1053, 775) 'Image window is this large

Dim szcir As Single

szcir = 5 'This determines the size of the annotated circle on the wings.

'Open calibration dialog

IpSCalShow(1)

IpSCalMove(1057, 450) 'Moves calibration window to bottom-right of screen, next to image window

IpSCalDestroy()

IpSCalCreate()

IpSCalSetUnitName("millimeter")

IpLocZoomShow(1)

IpLocZoomSet(IP\_LZ\_ZOOM, 200)

IpLocZoomSet(IP\_LZ\_CROSS, 1)

IpLocZoomMove(1057, 75) 'Moves local zoom window to top right of screen, next to image window

IpMacroStop("1. Click on the IMAGE button (pixels/unit) in the SPATIAL CALIBRATION window, 2. Position the line to measure 2 mm in the IMAGE window, 3. Enter a value of 2 in SCALING window, 4) Click CONTINUE",0)

IpCalGet("sYUnitPerPix", unitPerPix)

scale = IpMacroStop("Did you successfully get the scale?",MS\_MODAL+MS\_YESNOCAN+MS\_QUEST)

'Reset the calibration so that futher measurements are taken in pixels, not millimeters

'The WINGS programs locates the coordinates in pixels.

IpSCalSetUnit(1,1)

'Save the calibration image. GC

If scale = 1 Then

dateandtime = Replace(Format(Now,"m.d.yy/h.nn.AM/PM"),"/","\_")

calname = "scale"+dateandtime+".tif"

calname = IpTrim(calname)

calpath = imagepath + "\scale\" + calname

'Debug.Print(calname)

'Debug.Print(calpath)

IpWsSaveAs(calpath,"TIF")

End If

IpSCalShow(0)

IpAppCloseAll()

IpLocZoomShow(0)

'End calibration collection loop

Loop

If scale=2 Then

'If scale = 2 the user has pressed the cancel button during calibration and the macro calls exitmacro.

GoTo exitmacro

End If

'If scale = 1 the calibration has been set and the program will go on.

'Convert pixels per mm to mms per pix 4/09/08 SJS

If debuggingmode = False Then

mmperpix = Val(unitPerPix)

pixpermm = 1/CVar(mmperpix)

End If

'Collect is the main looper for taking multiple images.

'While Collect=1, another image will be taken.

collect = 1

Do While collect = 1

comments = " "

'If image = 0, an image will be snapped. Image will be set to 1 as soon as a good shot is taken.

Do While image = 0

exposure = 50.000000

'IpAcqControl(84, 0, exposure) 'Sets exposure time for preview. Disabled on Lumenera device. GC 2016.11.7

IpAcqShow(ACQ\_LIVE, 1)

'IpAcqControl(88, 1, IPNULL) ' ACQCMD\_AUTOEXPOSURE Temp disabled. GC

x(1) = 0

x(2) = 0

x(3) = 0

x(4) = 0

x(5) = 0

x(6) = 0

y(1) = 0

y(2) = 0

y(3) = 0

y(4) = 0

y(5) = 0

y(6) = 0

nocomment = False

dontWrite = False

'Prompts user to continue when ready to snap

IpMacroStop("Arrange the WING near one edge in the live preview, with the tips of LEGS near the other edge, then click CONTINUE when ready to snap.",0)

exposure = 500.000000

'IpAcqControl(84, 1, exposure) 'Sets exposure time for acquisition. Disabled on Lumenera device. GC 2016.11.7

IpAcqSnap(ACQ\_CURRENT)

IpAcqShow(ACQ\_LIVE,0)

'Resizes snap window to 50% size

IpWsZoom(50) 'Set image window to 50% zoom

IpDocMaximize() 'Maximize image window

IpDocSize(1053, 775) 'Image window is this large

image = IpMacroStop("Did you get a good image?",MS\_MODAL+MS\_YESNOCAN+MS\_QUEST)

If image = 0 Then

IpDocClose()

End If

Loop

'If image = 2 the user has pressed the cancel button in the previous prompt, and the macro is stopped.

If image = 2 Then

GoTo exitmacro

End If

If collect = 1 Then

'We must reset image to 0 for the next time around, since from the previous prompt, image = 1

image = 0

'Gets coordinates for wings

goodptwing = 0

Do While goodptwing = 0

'Place point 1

getapoint(x(1),y(1))

'Draw point 1

obj1 = IpAnCreateObj(GO\_OBJ\_ELLIPSE)

IpAnMove(0, x(1)-szcir, y(1)-szcir)

IpAnMove(5, x(1)+szcir, y(1)+szcir)

'Place point 2

getapoint(x(2),y(2))

IpMeasShow(1)

IpMeasTag(MEAS\_ALL, 1)

IpMeasDelete(MEAS\_TAG)

'Draw point 2

obj2 = IpAnCreateObj(GO\_OBJ\_ELLIPSE)

IpAnMove(0, x(2)-szcir, y(2)-szcir)

IpAnMove(5, x(2)+szcir, y(2)+szcir)

'Hides local zoom window

IpLocZoomShow(0)

'User input whether picture is good or not. Clears points if no.

goodptwing = IpMacroStop("Are the wing points entered correctly for this image?",MS\_MODAL+MS\_YESNO+MS\_QUEST)

If goodptwing=0 Then

IpAnActivateObjID(obj1)

IpAnDeleteObj()

IpAnActivateObjID(obj2)

IpAnDeleteObj()

End If

'End coord placing loop

Loop

'Gets coordinates for legs

goodptlegs = 0

Do While goodptlegs = 0

'Place point 3

getapoint(x(3),y(3))

'Draw point 3

obj3 = IpAnCreateObj(GO\_OBJ\_ELLIPSE)

IpAnMove(0, x(3)-szcir, y(3)-szcir)

IpAnMove(5, x(3)+szcir, y(3)+szcir)

'Place point 4

getapoint(x(4),y(4))

IpMeasShow(1)

IpMeasTag(MEAS\_ALL, 1)

IpMeasDelete(MEAS\_TAG)

'Draw point 4

obj4 = IpAnCreateObj(GO\_OBJ\_ELLIPSE)

IpAnMove(0, x(4)-szcir, y(4)-szcir)

IpAnMove(5, x(4)+szcir, y(4)+szcir)

'Place point 5

getapoint(x(5),y(5))

IpMeasShow(1)

IpMeasTag(MEAS\_ALL, 1)

IpMeasDelete(MEAS\_TAG)

'Draw point 5

obj5 = IpAnCreateObj(GO\_OBJ\_ELLIPSE)

IpAnMove(0, x(5)-szcir, y(5)-szcir)

IpAnMove(5, x(5)+szcir, y(5)+szcir)

'Place point 6

getapoint(x(6),y(6))

IpMeasShow(1)

IpMeasTag(MEAS\_ALL, 1)

IpMeasDelete(MEAS\_TAG)

'Draw Point 6

obj6 = IpAnCreateObj(GO\_OBJ\_ELLIPSE)

IpAnMove(0, x(6)-szcir, y(6)-szcir)

IpAnMove(5, x(6)+szcir, y(6)+szcir)

'Hide local zoom window after placing landmarks

IpLocZoomShow(0)

'User input whether picture is good or not. Clears points if no.

goodptlegs = IpMacroStop("Are the leg points entered correctly for this image?",MS\_MODAL+MS\_YESNO+MS\_QUEST)

If goodptlegs=0 Then

IpAnActivateObjID(obj3)

IpAnDeleteObj()

IpAnActivateObjID(obj4)

IpAnDeleteObj()

IpAnActivateObjID(obj5)

IpAnDeleteObj()

IpAnActivateObjID(obj6)

IpAnDeleteObj()

End If

'End coord placing loop

Loop

dialogbox:

'Calculate today's date and time from system info 07/04/09 SJS

dateandtime = Replace(Format(Now,"m/d/yy h:nnAM/PM"),"/","-")

'Define the custom dialog box for collecting user-entered information

'Programming note: In future, create new dialog boxes by pressing the "edit user dialog" button on the macro toolbar.

Begin Dialog UserDialog 400,381,"Image information" ' %GRID:10,7,1,1

TextBox 180,21,180,21,.TextBox1

TextBox 180,49,180,21,.TextBox9

TextBox 180,77,180,21,.TextBox2

TextBox 180,105,180,21,.TextBox3

TextBox 180,133,180,21,.TextBox6

TextBox 180,161,180,21,.TextBox4

TextBox 180,189,180,21,.TextBox5

TextBox 180,217,180,21,.TextBox7

TextBox 180,245,180,21,.TextBox8

TextBox 180,283,180,21,.TextBox10

Text 30,28,140,14,"Your name",.Text1

Text 30,84,140,14,"Today's date and time",.Text2

Text 30,112,140,14,"Name of first image",.Text3

Text 30,140,130,14,"Image Number",.Text6

Text 30,168,140,14,"Image Filepath",.Text4

Text 30,196,140,14,"Name of ascii File",.Text5

Text 30,224,130,14,"Genotype",.Text7

Text 30,252,130,14,"Comments",.Text8

Text 30,56,130,14,"Calibration Scale",.Text9

Text 30,290,130,14,"Sex",.Text10

OKButton 70,321,100,21

'CancelButton 190,321,100,21

End Dialog

'On Cancel GoTo exitmacro

Dim dlg As UserDialog

'Assign default values to the textboxes of the custom dialog box

dlg.textbox1 = author

dlg.textbox9 = pixpermm

dlg.textbox2 = dateandtime

dlg.textbox3 = imagename

dlg.textbox4 = imagepath

dlg.textbox5 = asciiname

dlg.textbox6 = imagenumber

dlg.textbox8 = comments

dlg.textBox10 = flySex

'SJS 10-23-10: The following do loop encloses the dialog showing and reading part.

'It is designed to check some entered data and force reentry if incorrect.

Do While(Not(okayToWrite))

'Show the custom dialog box

Dialog dlg

'read in user values entered into the textboxes

author = dlg.textbox1

dateandtime = dlg.textbox2

imagename = dlg.textbox3

imagepath = dlg.textbox4

asciiname = dlg.textbox5

imagenumber = dlg.textbox6

genotypename = dlg.textbox7

comments = dlg.textbox8

pixpermm = dlg.textbox9

flySex = dlg.textbox10

'trim the trailing spaces on the string variables

author = IpTrim(author)

dateandtime = IpTrim(dateandtime)

imagename = IpTrim(imagename)

imagepath = IpTrim(imagepath)

asciiname = IpTrim(asciiname)

genotypename = IpTrim(genotypename)

imagenum = CInt(imagenumber)

comments = IpTrim(comments)

pixpermm = IpTrim(pixpermm)

flySex = IpTrim(flySex)

'DEBUG: Ipmacrostop(genotypename, MS\_MODAL+MS\_OKCAN) 'testing genotype

If comments = "" Then

nocomment = True

End If

If comments = "x" Then

dontWrite = True

End If

'Create new image name

'iname is originally given a value of 255 spaces, so that it isn't an empty variable.

'255 spaces allows it to accomodate large names (many characters). GC

iname = Space(255)

IpStAutoName(imagename,imagenum,iname)

'To concatenate properly (linking imagepath and iname to make namepath), the variables must be trimmed first.

iname = IpTrim(iname)

namepath = imagepath + "\" + iname

'Check if the file exists, and if the numbers are within the min/max range for this machine.

'Note: Dir$() returns the filename if a file exists, returns blank string "" if it does not.

'If so, it is okay to go on.

If Dir$(namepath)="" Then

okayToWrite=True

'DEBUG: MsgBox("File found is named: " + Dir$(namepath))

Else

If IpMacroStop(("Warning! The file " + iname + " already exists in that directory. Overwrite Y/N?"),MS\_MODAL+MS\_YESNO+MS\_EXCLAM)=1 Then

'If user presses yes, return value will be 1

okayToWrite=True

'DEBUG: MsgBox("File found is named: " + Dir$(namepath))

Else

imagenumber = CStr(imagenum + 1)

GoTo dialogbox

End If

End If

Loop

'Once the loop has exited, it means that the directory exists and the file doesn't, and we can save now.

If dontWrite = False Then

'Define and open the output file

'NOTE: to concatenate properly (linking imagepath and asciiname to make outfile), the variables must be trimmed first (see above).

'10-23-10 SJS: Adding a prefix to ascii file that specifies which wingmachine it came from.

outfile = imagepath + "\" + asciiname

'DEBUG: ret=MsgBox(outfile)

Open outfile For Append As #1

IpWsSaveAs(namepath, "TIF")

'Also, reset the condition of okayToWrite for the next image name.

okayToWrite=False

'Print all info to the output file. Tabs are used to delimit information when importing into Excel.

'The last two variables to be printed, "Str$(1) + Str$(1)" are superfluous. What data do they convey? GC

Print #1, iname + " " + Str$(x(1)) + " " + Str$(y(1)) + " " + Str$(x(2)) + " " + Str$(y(2)) + " " + Str$(x(3)) + " " + Str$(y(3)) + " " + Str$(x(4)) + " " + Str$(y(4)) + " " + Str$(x(5)) + " " + Str$(y(5)) + " " + author + " " + dateandtime + " "+ genotypename + " " + flySex + " " + Str$(mmperpix) + " " + Str$(x(6))+ " " + Str$(y(6))

'Print comment line if necessary

'On Error Next GoTo dialogbox

If nocomment = False Then

Print #1, comments

End If

'Increment id# so that the next image will have a different id name associated with it

imagenum = imagenum + 1

imagenumber = CStr(imagenum)

'Close current image and output file

Close #1

'Ask the user if there is another image to be acquired

collect = IpMacroStop("Do you want to acquire another image?",MS\_MODAL+MS\_YESNO+MS\_QUEST)

End If

End If

'End collect

IpAppCloseAll()

'End loop that lets you take multiple pictures in one session

Loop

If collect = 0 Then

GoTo exitmacro

End If

Exit Sub

'exitmacro: place where goto calls. Allows resetting of sub upon cancel clicks. all cancel options call exitmacro. GC 2016.6.15

'IpMacroRun("WingLeg", "") Recalls the macro

exitmacro:

Dim userinput As Single

userinput = IpMacroStop("Exit macro?", MS\_MODAL+MS\_YESNO)

If userinput = 1 Then

IpAppCloseAll()

Exit Sub

Else

IpAppCloseAll()

IpMacroRun("WingLeg", "")

End If

Exit Sub

End Sub

Sub Trident()

'Trident modified by David Houle 3/2/2023

'Modified version of PleioSelect, based On FlexWingZoom (which Is a modified BigWingZoom) To take images

'Image size produced is unspecific.

'Uses basic dialog

'Contains four locations to change the exposure value, based on the camera being used; two in the calibrate module, and two in the collect module

'The QCam will need these values to be operational, while the Lumenera camera has them disabled.

Dim comments$, author$, namepath$, dateandtime$, flySex$, imagename$, imagepath$, asciiname$, genotypename$, imagenumber$

Dim block$, transfer$, replicate$, mife$, gene$, driver$, direction$, message$

Dim imagenum%, outfile$, collect%, image%, scale%, goodptwing%, goodptlegs%, ampm%, iname$

Dim calname$, calpath$

'Dim points%, obj1%, obj2%, obj3%, obj4%, obj5%, obj6%

Dim x(6) As Single, y(6) As Single, mmperpix As Single

Dim nocomment As Boolean

Dim dontWrite As Boolean

Dim okayToWrite As Boolean

Dim debuggingmode As Boolean

Dim pathacquire%, okToMakeDir As Boolean

Dim pixpermm$

Dim unitPerPix As String \*20

Dim picDate As Date

Dim exposure As Double 'Needs to be declared as a double. GC

'Temp enabled. Closes any open windows from any previously-running macro. GC

IpAppCloseAll()

IpSCalShow(0)

IpLocZoomShow(0)

IpAcqShow(ACQ\_LIVE, 0)

IpAcqShow(ACQ\_SNAP, 0)

'Set starting values for the information to be collected.

imagenum = minVal

imagenumber = Str(minVal)

author = "AnonymousRuckmanite"

dateandtime = "2\_Mar\_2023"

imagename = "example\_.tif"

asciiname = "example.asc"

flySex = "X"

nocomment = False

dontWrite = False

'oktomakedir is a boolean which determines if a directory exists or not. GC 2016.6.22

'This allows users to create new directories at the workspace, instead of manually.

'It begins as False until checked through the pathacquire loop.

okToMakeDir = False

'SJS 10-23-10: The following boolean is used to determine whether an image file exists before writing.

'This should prevent file overwrites.

'It starts out false until we check and make sure it's okay to write to a given filename

okayToWrite=False

' This set of commands changes the appearance of the annotated points.

'IpAnSet(GO\_ATTR\_PENWIDTH, 2)

'IpAnSet(GO\_ATTR\_USEASDEFAULT, 1)

'IpAnSet(GO\_ATTR\_PENCOLOR, 100)

'IpAnSet(GO\_ATTR\_USEASDEFAULT, 1)

'Debugging mode. Set debuggingmode = True to initiate If statement.

debuggingmode = False

If debuggingmode = True Then

scale = 1

mmperpix = 1

End If

'The code for making new directories. GC 2016.6.22

'Input a string variable.

'Surround in if-statement that checks if the directory exists, and if not, makes the new one.

'If the directory does exist, the user is prompted whether they would like to use that directory.

'Still doesn't contain robust code to account for nested directory creation.

Do

'imagepath = "Z:\DGRP\Wings\_Legs"

imagepath = Space(255) 'imagepath must have a value prior to IpStGetStr, otherwise it returns a null string. GC

pathacquire = IpStGetString("Enter the filepath for images.", imagepath, 75)

imagepath = IpTrim(imagepath)

'Debug.Print(imagepath + "&&&&Filepath should go here.")

If pathacquire = 1 Then

If imagepath = "" Or IsNull(imagepath) Then 'This is if the user didn't enter a value at the prompt, and simply clicked OK. GC

IpMacroStop("No filepath was given. Enter a filepath in the prompt.", 0)

GoTo loopToMakeDir

End If

If Dir(imagepath, vbDirectory) = "" Then 'This evaluates if the string imagepath is a directory. GC

okToMakeDir = True

MkDir(imagepath)

MkDir(imagepath + "\scale")

IpMacroStop("Created new directory: " + imagepath, 0)

Else

If IpMacroStop("Directory already exists. Use this directory?", MS\_MODAL+MS\_YESNO) = 1 Then

okToMakeDir = True 'Doesn't make a new directory. Overrides the loop to continue. GC

Else

GoTo loopToMakeDir

End If

End If

Else

GoTo exitmacro

End If

loopToMakeDir:

'End filepath acquisition loop.

Loop Until okToMakeDir = True

'Opens a live window so user can set the appropriate zoom level

exposure = 50.000000

'IpAcqControl(84, 0, exposure) 'Sets exposure time for preview to 50ms. Disabled on Lumenera device. GC 2016.11.7

IpAcqShow(ACQ\_LIVE, 1)

IpMacroStop("Center the fly in the live window. Adjust scope zoom to the appropriate level, such that the tips of the wing and legs are fully in the frame, slightly away from the edges. Click CONTINUE when ready to proceed to calibration routine.", 0)

Do While scale=0

'When scale = 0 the calibration has not been set

'These commands 1) open the live preview; and 2) adjust exposure. Then calibration pathway initiates. GC

exposure = 50.000000

'IpAcqControl(84, 0, exposure) 'Sets exposure time for preview to 50ms

IpAcqShow(ACQ\_LIVE, 1)

'IpAcqControl(88, 1, IPNULL) Temp disabled. GC

'ACQCMD\_AUTOEXPOSURE

'Taking calibration picture:

IpMacroStop("Place a calibration scale in the active video image, then click CONTINUE when ready to snap.",0)

exposure = 500.000000

'IpAcqControl(84, 1, exposure) 'Sets exposure time for acquisition to 500ms. Disabled on Lumenera device. GC 2016.11.7

IpAcqSnap(ACQ\_CURRENT)

IpAcqShow(ACQ\_LIVE,0)

IpWsZoom(50) 'Set image window to 50% zoom

IpDocMaximize() 'Maximize image window

IpDocSize(1053, 775) 'Image window is this large

Dim szcir As Single

szcir = 5 'This determines the size of the annotated circle on the wings.

'Open calibration dialog

IpSCalShow(1)

IpSCalMove(1057, 450) 'Moves calibration window to bottom-right of screen, next to image window

IpSCalDestroy()

IpSCalCreate()

IpSCalSetUnitName("millimeter")

IpLocZoomShow(1)

IpLocZoomSet(IP\_LZ\_ZOOM, 200)

IpLocZoomSet(IP\_LZ\_CROSS, 1)

IpLocZoomMove(1057, 75) 'Moves local zoom window to top right of screen, next to image window

IpMacroStop("1. Click on the IMAGE button (pixels/unit) in the SPATIAL CALIBRATION window, 2. Position the line to measure 2 mm in the IMAGE window, 3. Enter a value of 2 in SCALING window, 4) Click CONTINUE",0)

IpCalGet("sYUnitPerPix", unitPerPix)

scale = IpMacroStop("Did you successfully get the scale?",MS\_MODAL+MS\_YESNOCAN+MS\_QUEST)

'Reset the calibration so that futher measurements are taken in pixels, not millimeters

'The WINGS programs locates the coordinates in pixels.

IpSCalSetUnit(1,1)

'Save the calibration image. GC

If scale = 1 Then

dateandtime = Replace(Format(Now,"m.d.yy/h.nn.AM/PM"),"/","\_")

calname = "scale"+dateandtime+".tif"

calname = IpTrim(calname)

calpath = imagepath + "\scale\" + calname

'Debug.Print(calname)

'Debug.Print(calpath)

IpWsSaveAs(calpath,"TIF")

End If

IpSCalShow(0)

IpAppCloseAll()

IpLocZoomShow(0)

'End calibration collection loop

Loop

If scale=2 Then

'If scale = 2 the user has pressed the cancel button during calibration and the macro calls exitmacro.

GoTo exitmacro

End If

'If scale = 1 the calibration has been set and the program will go on.

'Convert pixels per mm to mms per pix 4/09/08 SJS

If debuggingmode = False Then

mmperpix = Val(unitPerPix)

pixpermm = 1/CVar(mmperpix)

End If

'Collect is the main looper for taking multiple images.

'While Collect=1, another image will be taken.

collect = 1

Do While collect = 1

comments = " "

'If image = 0, an image will be snapped. Image will be set to 1 as soon as a good shot is taken.

Do While image = 0

exposure = 50.000000

'IpAcqControl(84, 0, exposure) 'Sets exposure time for preview. Disabled on Lumenera device. GC 2016.11.7

IpAcqShow(ACQ\_LIVE, 1)

'IpAcqControl(88, 1, IPNULL) ' ACQCMD\_AUTOEXPOSURE Temp disabled. GC

nocomment = False

dontWrite = False

'Prompts user to continue when ready to snap

IpMacroStop("Arrange the fly so that the thorax is clearly visible and well list, then click CONTINUE when ready to snap.",0)

exposure = 500.000000

'IpAcqControl(84, 1, exposure) 'Sets exposure time for acquisition. Disabled on Lumenera device. GC 2016.11.7

IpAcqSnap(ACQ\_CURRENT)

IpAcqShow(ACQ\_LIVE,0)

'Resizes snap window to 50% size

IpWsZoom(50) 'Set image window to 50% zoom

IpDocMaximize() 'Maximize image window

IpDocSize(1053, 775) 'Image window is this large

image = IpMacroStop("Did you get a good image?",MS\_MODAL+MS\_YESNOCAN+MS\_QUEST)

If image = 0 Then

IpDocClose()

End If

Loop

'If image = 2 the user has pressed the cancel button in the previous prompt, and the macro is stopped.

If image = 2 Then

GoTo exitmacro

End If

If collect = 1 Then

'We must reset image to 0 for the next time around, since from the previous prompt, image = 1

image = 0

dialogbox:

'Calculate today's date and time from system info 07/04/09 SJS

dateandtime = Replace(Format(Now,"m/d/yy h:nnAM/PM"),"/","-")

'Define the custom dialog box for collecting user-entered information

'Programming note: In future, create new dialog boxes by pressing the "edit user dialog" button on the macro toolbar.

Begin Dialog UserDialog 400,381,"Image information" ' %GRID:10,7,1,1

TextBox 180,21,180,21,.TextBox1

TextBox 180,49,180,21,.TextBox9

TextBox 180,77,180,21,.TextBox2

TextBox 180,105,180,21,.TextBox3

TextBox 180,133,180,21,.TextBox6

TextBox 180,161,180,21,.TextBox4

TextBox 180,189,180,21,.TextBox5

TextBox 180,217,180,21,.TextBox7

TextBox 180,245,180,21,.TextBox8

TextBox 180,283,180,21,.TextBox10

Text 30,28,140,14,"Your name",.Text1

Text 30,84,140,14,"Today's date and time",.Text2

Text 30,112,140,14,"Name of first image",.Text3

Text 30,140,130,14,"Image Number",.Text6

Text 30,168,140,14,"Image Filepath",.Text4

Text 30,196,140,14,"Name of ascii File",.Text5

Text 30,224,130,14,"Genotype",.Text7

Text 30,252,130,14,"Comments",.Text8

Text 30,56,130,14,"Calibration Scale",.Text9

Text 30,290,130,14,"Sex",.Text10

OKButton 70,321,100,21

'CancelButton 190,321,100,21

End Dialog

'On Cancel GoTo exitmacro

Dim dlg As UserDialog

'Assign default values to the textboxes of the custom dialog box

dlg.textbox1 = author

dlg.textbox9 = pixpermm

dlg.textbox2 = dateandtime

dlg.textbox3 = imagename

dlg.textbox4 = imagepath

dlg.textbox5 = asciiname

dlg.textbox6 = imagenumber

dlg.textbox8 = comments

dlg.textBox10 = flySex

'SJS 10-23-10: The following do loop encloses the dialog showing and reading part.

'It is designed to check some entered data and force reentry if incorrect.

Do While(Not(okayToWrite))

'Show the custom dialog box

Dialog dlg

'read in user values entered into the textboxes

author = dlg.textbox1

dateandtime = dlg.textbox2

imagename = dlg.textbox3

imagepath = dlg.textbox4

asciiname = dlg.textbox5

imagenumber = dlg.textbox6

genotypename = dlg.textbox7

comments = dlg.textbox8

pixpermm = dlg.textbox9

flySex = dlg.textbox10

'trim the trailing spaces on the string variables

author = IpTrim(author)

dateandtime = IpTrim(dateandtime)

imagename = IpTrim(imagename)

imagepath = IpTrim(imagepath)

asciiname = IpTrim(asciiname)

genotypename = IpTrim(genotypename)

imagenum = CInt(imagenumber)

comments = IpTrim(comments)

pixpermm = IpTrim(pixpermm)

flySex = IpTrim(flySex)

'DEBUG: Ipmacrostop(genotypename, MS\_MODAL+MS\_OKCAN) 'testing genotype

If comments = "" Then

nocomment = True

End If

If comments = "x" Then

dontWrite = True

End If

'Create new image name

'iname is originally given a value of 255 spaces, so that it isn't an empty variable.

'255 spaces allows it to accomodate large names (many characters). GC

iname = Space(255)

IpStAutoName(imagename,imagenum,iname)

'To concatenate properly (linking imagepath and iname to make namepath), the variables must be trimmed first.

iname = IpTrim(iname)

namepath = imagepath + "\" + iname

'Check if the file exists, and if the numbers are within the min/max range for this machine.

'Note: Dir$() returns the filename if a file exists, returns blank string "" if it does not.

'If so, it is okay to go on.

If Dir$(namepath)="" Then

okayToWrite=True

'DEBUG: MsgBox("File found is named: " + Dir$(namepath))

Else

If IpMacroStop(("Warning! The file " + iname + " already exists in that directory. Overwrite Y/N?"),MS\_MODAL+MS\_YESNO+MS\_EXCLAM)=1 Then

'If user presses yes, return value will be 1

okayToWrite=True

'DEBUG: MsgBox("File found is named: " + Dir$(namepath))

Else

imagenumber = CStr(imagenum + 1)

GoTo dialogbox

End If

End If

Loop

'Once the loop has exited, it means that the directory exists and the file doesn't, and we can save now.

If dontWrite = False Then

'Define and open the output file

'NOTE: to concatenate properly (linking imagepath and asciiname to make outfile), the variables must be trimmed first (see above).

'10-23-10 SJS: Adding a prefix to ascii file that specifies which wingmachine it came from.

outfile = imagepath + "\" + asciiname

'DEBUG: ret=MsgBox(outfile)

Open outfile For Append As #1

IpWsSaveAs(namepath, "TIF")

'Also, reset the condition of okayToWrite for the next image name.

okayToWrite=False

'Print all info to the output file. Tabs are used to delimit information when importing into Excel.

'The last two variables to be printed, "Str$(1) + Str$(1)" are superfluous. What data do they convey? GC

Print #1, iname + " " + author + " " + dateandtime + " "+ genotypename + " " + flySex + " " + Str$(mmperpix)

'Print comment line if necessary

'On Error Next GoTo dialogbox

If nocomment = False Then

Print #1, comments

End If

'Increment id# so that the next image will have a different id name associated with it

imagenum = imagenum + 1

imagenumber = CStr(imagenum)

'Close current image and output file

Close #1

'Ask the user if there is another image to be acquired

collect = IpMacroStop("Do you want to acquire another image?",MS\_MODAL+MS\_YESNO+MS\_QUEST)

End If

End If

'End collect

IpAppCloseAll()

'End loop that lets you take multiple pictures in one session

Loop

If collect = 0 Then

GoTo exitmacro

End If

Exit Sub

'exitmacro: place where goto calls. Allows resetting of sub upon cancel clicks. all cancel options call exitmacro. GC 2016.6.15

'IpMacroRun("WingLeg", "") Recalls the macro

exitmacro:

Dim userinput As Single

userinput = IpMacroStop("Exit macro?", MS\_MODAL+MS\_YESNO)

If userinput = 1 Then

IpAppCloseAll()

Exit Sub

Else

IpAppCloseAll()

IpMacroRun("WingLeg", "")

End If

Exit Sub

End Sub 'Trident