Assignment 10: Image Processing and Morphing

```
<< PollyMorphometrics`
<< PollyPhylogenetics`
PollyMorphometrics 12.0
(c) P. David Polly, 26 January 2016

Phylogenetics for Mathematica 3.0
(c) P. David Polly, 16 October 2014
```

Collecting Landmark Coordinates in Mathematica

Create a list of file names

```
filelist = FileNames["/Users/pdavidpolly/Desktop/Lectures/G562
Geometric Morphometrics/Data files/Morphing/*.JPG"];
```

Import the file names

```
images = Import[#] & /@ filelist;
```

Display the image (it is convenient to show the image name at the same time)
Open the Drawing Tools from the Graphics Menu
Select the GetCoordinates[] tool (cross hairs)
Click on landmark locations and then copy coordinates to the clipboard
Paste coordinates into Mathematica or a text file
Here are pasted landmarks from several images. I entered the image name by hand. Join the landmarks into a single data list using curly brackets and commas.

lands = {
    "BlarinaBrevicauda", {1092, 823}, {1190, 817.4}, {1220, 664.3},
    {1165, 544.7}, {1229, 444.5}, {1059, 475.1}, {563.5, 511.3}},
    "SorexAntinorii", {1329, 836.9}, {983.7, 795.1}, {627.5, 764.5},
    {229.6, 611.5}, {813.9, 951}, {455, 923.1}, {360.3, 859.1}, {1134, 1051},
    {1287, 861.9}, {911.3, 661.6}, {839, 992.7}, {580.2, 953.7}}
}
Morphing a photo

\texttt{landmarks = tpsImport[}
    "/Users/pdavidpolly/Desktop/Lectures/G562 Geometric Morphometrics/Data
    files/Morphing/landmarks.tps", "IMAGE"];

We need to resize the image, and also resize the landmarks. Note we are changing size by 1/8.

\texttt{Dimensions[ImageData[images[[2]]]]}
\{1200, 1600, 3\}

\texttt{images = ImageResize[#, 200] & /@ images;}
\texttt{Dimensions[ImageData[images[[2]]]]}
\{150, 200, 3\}

Here we partition the landmarks for the second image and multiply them by 1/8 to match the resizing of the photo.

\texttt{pts = Partition[landmarks[[2, 2 ;;]] / 8, 2];}

Now we put the data into two matrices. The first matrix (im) contains the RGB values for each pixel in the image. The second matrix (imcoords) contains the x-y coordinates for each pixel. Because pixels are standardly numbered from the top left corner of the photograph, we have to create the table of x,y values in opposite order so the image ends up in the right orientation.

\texttt{im = ImageData[images[[2]]];}
\texttt{imcoords = Table[{x, y}, {y, Dimensions[im][[1]], 1, -1}, {x, Dimensions[im][[2]]}];}
\texttt{Dimensions[imcoords]}
\{150, 200, 2\}

Now we can plot the whole thing as a bivariate scatter plot using the Graphics function. We put each coordinate from imcoords in the Point primitive and set its color with the corresponding RGB value in the im matrix. We then plot the landmarks on top of it. Adding axis
Now we perform Procrustes on the landmarks and translate, rescale, and rotate the image to match. First the Procrustes...

\[ \text{proc} = \text{Procrustes}[\text{landmarks}[[1;;, 2;;]], \text{Length}[\text{landmarks}[[1, 2;;]]]/2, 2]; \]

Now figure out where the original centroid of the landmarks for shape 1 were, and subtract them from the coordinates in the imcoords matrix. We'll put the coords in a new variable just in case we mess up. And we'll plot the image again to see if we got it right. Plotting the Procrustes coords for the first image helps us tell if we are getting close. Note the landmarks appear to be tiny because they have been rescaled but we haven't rescaled the image yet.

\[ \text{centroid} = \text{Mean}[\text{pts}] \]
\[ \text{pts} = \text{imcoords} - \text{centroid} \]
\[ \{91.5221, 71.8235\} \]

\[ \text{newcoords} = \text{Table}[\text{imcoords}[[x, y]] - \text{centroid}, \{x, \text{Length}[\text{imcoords}]\}, \{y, \text{Length}[\text{imcoords}[[1]]]\}]\]
Now calculate the centroid size of the original landmarks. We divide each coordinate of the image by that number to bring it to the same scale. We are getting close...

```math
\text{centroidsize} = \text{Sqrt}[\text{Plus} @ (\text{Flatten}[\text{pts}]^2)]
```

```
pts = pts / centroidsize;
98.5245
```

```math
\text{newcoords} = \text{Table}[\text{newcoords}[[x, y]] / \text{centroidsize},
  \{x, \text{Length}[\text{newcoords}]\}, \{y, \text{Length}[\text{newcoords}[[1]]]\}];
```

But we still need to rotate. To do this we have to find the rotation matrix that will rotate the original landmarks to the orientation of their Procrustes equivalent. The first step is to translate and scale the
originals, which we do by subtracting the centroid and dividing by centroid size.

Then, following the fine print in the Rohlf and Slice paper on Procrustes, we calculate the rotation matrix using singular value decomposition on the rescaled original points and the Procrustes superimposed points. I’ve converted the variable names to the ones used by Rohlf in case you want to compare this code to his paper. In the code, \( I \) contains the original points, \( y \) contains the target points (Procrustes superimposed ones), and \( hh \) is the rotation matrix from original to target.

Once we find \( hh \) then we multiply each coordinate in \( \text{newcoords} \) to rotate the entire image to its Procrustes superimposed landmark orientation.

\[
l = \text{pts} \\quad \{\begin{array}{l}
(0.294119, 0.0195532), (0.325837, 0.070302), \\
(0.297925, 0.0804518), (0.24337, 0.0601523), (0.129186, 0.0347779), \\
(0.0327628, -0.00201503), (-0.0382855, -0.0223146), (-0.181651, 0.160381), \\
(-0.264118, 0.157844), (-0.276805, -0.0197771), (-0.351659, -0.0261207), \\
(-0.253968, -0.111125), (-0.218444, -0.123812), (-0.148664, -0.142843), \\
(0.0378377, -0.109856), (0.165978, -0.0426141), (0.206578, 0.0170158)
\end{array} \}
\]

\[
y = \text{Partition[proc[[2]], 2]} \quad \{\begin{array}{l}
(0.285782, -0.0722278), (0.331622, -0.0337539), \\
(0.308208, -0.0154809), (0.250051, -0.0179415), (0.133611, -0.00681497), \\
(0.0305393, -0.0120339), (-0.0433052, -0.00940121), (-0.123246, 0.208637), \\
(-0.202466, 0.23169), (-0.269383, 0.0666681), (-0.342538, 0.0837499), \\
(-0.275871, -0.0272674), (-0.246001, -0.0503045), (-0.185509, -0.0899534), \\
(0.00206432, -0.116172), (0.144707, -0.0917862), (0.201736, -0.0476079)
\end{array} \}
\]

\[
\{u, w, v\} = \text{SingularValueDecomposition[Transpose[l].y];} \\
\quad hh = u.(w*\text{Inverse[Abs[w]]}).\text{Transpose[v]}; \\
\text{newcoords} = \text{newcoords.hh};
\]
Graphics[
  Table[{RGBColor[im[[y, x]]], Point[newcoords[[y, x]]]}, {y, Dimensions[im][[1]]}, 
  {x, Dimensions[im][[2]]}], {Red, Point[Partition[proc[[2]], 2]]}], Frame -> True]

tpSpline[proc[[2]], proc[[1]]]

tpSplineMorph[source_, target_, imgcoords_] := Module[{T, st, SigmaH, i, j, InvBigG, W, c, A, LastTerm, T=Partition[Flatten[source], 2]; 
  Y=Partition[Flatten[target], 2]; 
  SigmaH[vec1_, vec2_] := Module[{h},If[(h=Norm[vec1-vec2])>0,Return[(h^2)*Log[h]],Return[0]]; 
  st=Table[SigmaH[T[[i]], T[[j]]], {i,Length[T]}, {j,Length[T]}]; 
  InvBigG=Inverse[ArrayFlatten[{{(st, T, (1, 0, 0), (Transpose[T], 0, 0))}]]]; 
  W=InvBigG[[1]];Length[T], 1]; Length[T]].Y; 
  {c,A}={InvBigG[[Length[T]+1];Length[T]+3, 1];Length[T]].Y[[1,1;;2]], Transpose[(InvBigG LastTerm[pt_]:=Transpose[W].Table[SigmaH[pt, T[[i]]], {i,Length[T]}]); 
  yyyy[pt_]:=Flatten[c]+A, pt+LastTerm[pt]; 
  gridpts=Table[yyyy[imgcoords[[y,x]]], {y, Dimensions[imgcoords][[1]]}, {x, Dimensions[imgcoords][[2]]}];
}
morphedcoords = tpSplineMorph[proc[[2]], proc[[1]], newcoords];

Graphics[{Table[{RGBColor[im[[y, x]]], Point[morphedcoords[[y, x]]]},
{y, Dimensions[im][[1]]}, {x, Dimensions[im][[2]]}], {Blue, Thick,
Line[im] & @ Transpose[{Partition[proc[[1]], 2], Partition[proc[[2]], 2]()]},
{Red, Point[Partition[proc[[1]], 2]]}], Frame -> True}
images[[2]]

images[[1]]