#### L12, 7 Mar, Segmentation: MAS Rotor (part 3)

1) Download Moodle/Week 8/Pgm10\_Segmentation\_MAS\_rotor\_v2.nb 2) Download Moodle/Week 8/MAS\_rotor\_cropped.h5 (177MB)



Philosophy

Our *a priori* knowledge about a sample can be used in segmentation.

Two If statements for defining boundaries of only Kel-F and only zirconia.

Human eye used to find best values of binarization threshold.

Subtract zirconia binary image from original image to get a Kel-F only image.



Mathematica commands - image processing MorphologicalBinarization - a binarization GeometricMeanFilter - a median filter TotalVariationFilter - noise reducing filter ImageMultiply -

ImageSubtract -

ImageData - convert image format to array of numbers MorphologicalComponents - makes a label field ComponentMeasurements - inspects a label field Erosion

Closing

ImageAdjust

**NOTE:** Know these commands. See Mathematica help files.



## Mathematica commands - list commands

Position Ordering Reverse First, Rest ConstantArray Length Dimensions Flatten Append

### Mathematica commands - set commands Intersection

### **NOTE:** Know these commands. See Mathematica help files.



### Mathematica commands - procedures

Manipulate Module Do

If

Map

### Mathematica commands - Input/Output NotebookDirectory Import Export FileNames

**NOTE:** Will not be explicitly tested on the exam.



### Mathematica commands - computation FindFit ReplaceAll ( /. ) Interpolation

### **NOTE:** Will not be explicitly tested on the exam.



Q2. Please explain how three orthoslices can be shown at one time.

Key: (page 41-42) By now, most Avizo users can probably do this in their sleep. The programming elements are medium quality: (1) Right mouse on the data element and "pool window" shows common options like an OrthoSlice. (2)\_ The OrthoSlice defaults to "xy" plane. (3) A bounding box nicely frames the slices.

Curiously, the user gets no visible clues about axes orientation; it's a complete surprise as to which plane will show up. I don' like the itsy-bitsy on/off button on OrthoSlice; it's used way to often to be left so small.

It would be nice to be see the connections between the main/console/pool windows. What if I moused over a slice in the main window and the icon in Pool flashed, and the console showed the data



structure essentials (dimensions, min/max). Maybe the line between data source and OrthosSlice should flash. Conversely, if we highlight an OrthoSlice icon, we might like to see the slice highlighted (perimeter?) in the main window. Your thoughts?

vi     Colonvash     Colonvash    <	Avizo – Standard Edition – Q2_Three_Orthoslices.hx							
Over Data   Colowach ContractControl CreateImage CulinderSite   Contractor Control CreateImage CulinderSite   Properties   Contentation:   Colowach Control Contr	88 ° 🗸	۵ 🛛		Β			Pool	0
Coloneach ContractControl CreateImage (vinderSites)     Coloneach Control							Open Data	
oupdate the license.							Colorwash ContrastControl CreateImage CylinderSlice	
Properties     Properties     Prioslice2     Image: Image								
Poperties   © CrhoSice?   © Data:   © Data:   © Data:   © Orientation:   > xy > xz    © Orientation:   > xy > xz    © Options:   > adjust view > bilinear view > lighting   © Mapping Type:   © Data Window:   min @ max 255   © Sice Number:   ● Lata:   • Options:   • adjust view > bilinear view > lighting   © Sice Number:   • Iza	1							1
Properties   ChtoSlice3   ChtoSlice3 <th></th> <th></th> <th></th> <th></th> <th></th> <th></th> <th></th> <th></th>								
Properties   @ Data:   @ Data:   @ Data:   @ Orientation:   x y xz 0 yz   @ Options:   adjust view bilinear view lighting   @ Mapping Type:   Inaar 255   @ Silce Number:   @ Data Window:   min Imax 255   @ Silce Number:   @ Data Window:   min Imax 255   @ Silce Number:   @ Data Window:   Imax 255   @ Silce Number:   @ Data Window:   Imax 255   @ Silce Number:   Imax 255   Imax 255 <th></th> <th></th> <th></th> <th></th> <th></th> <th></th> <th></th> <th></th>								
Properties   © Data:   © Data:   © Data:   © Orientation:   xy xz yz   © Options:   > adjust view bilinear view   lighting   © Mapping Type:   © Data Window:   min @ max 255   © Slice Number:   © Data Window:   © Inaar-efresh								"
Properties   CrthoSlice   CrthoSlice <th>X</th> <th></th> <th></th> <th></th> <th></th> <th></th> <th></th> <th></th>	X							
Properties   ChthoSlice3   © Data:   motor.am   © Data:   © Orientation:   x y x 2 O y2   © Options:   adjust view Dillnear view Ilighting   © Mapping Type:   Inear   © Data Window:   min 0   max 255   © Slice Number:   Image: State in the intervence in			オー					
Properties   © Data:   © Drientation:   xy   xz   Ø Options:   adjust view   bilinear view   Bata Window:   min   max   25:   Silice Number:   Bata Window:   min   max   25:   Silice Number:   Transparency:   None   Binary   Alpha	ſ							
Properties   Image: Control of the second s	(							
Properties   Image: CrithoSlice3   Imag	$\setminus$							
Properties   Image: Construction of the second sec								
Properties   OrthoSlice3   & Data:   motor.am   & Data:   motor.am   & Frame:   Show width:   1   0 orientation:   xy   xz   Options:   adjust view   bilinear view   lighting   Mapping Type:   Linear   Data Window:   min   max   255   Slice Number:   Image: Comparison of the state								
Properties     OrthoSlice3     Ø Data:     motor.am     Ø Data:     Mapping Type:     Interview     Interview     Ø Options:     adjust view     billnear view     Interview     Interview </th <th><math>-X_{-}</math></th> <th></th> <th></th> <th></th> <th></th> <th></th> <th></th> <th>8</th>	$-X_{-}$							8
o update the license.      o update the license.    o update the license.    o update the license.       o update the license.    o update the license.    o update the license.    o update the license.    o update the license.    o update the license.    o update the license.    o update the license.    o update the license.    o update the license.    o update the license.    o update the license.    o update the license.    o update the license.    o update the license.    o update the license.    o update the license.       o update the license.    o update the license.    o update the license.       o update the license.       o update the license.       o update the license.       o update the license.       o update the license.       o update the license.       o update the license.       o update the license.       o update the license.       o update the license.	$\sim$					í	Properties	
<ul> <li>Data:</li> <li>motor.am</li> <li>Frame:</li> <li>Show width:</li> <li>motor.am</li> <li>Frame:</li> <li>Show width:</li> <li>motor.am</li> <li>Solution:</li> <li>motor.ax</li> <li>Solution:</li> <li>motor.ax</li> <li>Solution:</li> <li>motor.ax</li> <li>Solution:</li> <li>motor.ax</li> <li>Solution:</li> <li>motor.ax</li> <li>motor.ax</li> <li>Solution:</li> <li>motor.ax</li> <li>motor.ax</li> <li>Solution:</li> <li>motor.ax</li> <li>motor.ax</li> <li>Solution:</li> <li>motor.ax</li> <li>motor</li></ul>							OrthoSlice3	?
Solution: Solution:   Solution: Solutio		V					& Data: motor.am : 🖘	
<ul> <li>Orientation:</li> <li>xy xz yz</li> </ul> <li>Options:</li> <li>adjust view   bilinear view   lighting</li> <li>Mapping Type:</li> <li>Linear</li> <li>Data Window:</li> <li>min 0 max 255</li> <li>Slice Number:</li> <li>Data Window:</li> <li>Mapping Type:</li> <li>Interspace</li> <li>Transparency:</li> <li>None</li> <li>Binary</li> <li>Alpha</li> <li>Apply Stop</li>							& Frame:	
<ul> <li>Options:</li> <li>adjust view</li> <li>bilinear view</li> <li>lighting</li> <li>Mapping Type:</li> <li>Linear</li> <li>Data Window:</li> <li>min 0</li> <li>max 255</li> <li>Slice Number:</li> <li>Data Window:</li> <li>min 0</li> <li>max 255</li> <li>Slice Number:</li> <li>None</li> <li>Binary</li> <li>Alpha</li> </ul>							& Orientation: ○ xy ○ xz ⊙ yz	
<ul> <li>Mapping Type:</li> <li>Linear</li> <li>Data Window:</li> <li>min 0</li> <li>max 255</li> <li>Slice Number:</li> <li>Slice Number:</li> <li>Transparency:</li> <li>None</li> <li>Binary</li> <li>Alpha</li> </ul> Apply Stop							8 Options: adjust view bilinear view lighting	
o update the license.							& Mapping Type: Linear 🗘	
o update the license.							🚳 Data Window: min 0 max 255	
o update the license.							§ Slice Number:	
o update the license.							💩 Transparency: 💿 None 🕞 Binary \ominus Alpha	
auto-refresh Apply Stop	o update the	e license.				ń		
auto-refresh Apply Stop								
auto-refresh Apply Stop						n		
auto-refresh Apply Stop						U		
						Ŷ	auto-refresh Apply Stop	

Philosophy

Our *a priori* knowledge about a sample can be used in segmentation.

Two If statements for defining boundaries of only Kel-F and only zirconia.

Human eye used to find best values of binarization threshold.

Subtract zirconia binary image from original image to get a Kel-F only image.



### **Philosophy** Our *a priori* knowledge about a sample can be used in segmentation.



indexRow	0	
	1	+ >
threshold	0.5	- + *
veParameters		
{three pear	eshold =, 0.5, k position=,	amp=, 301 0.503946, si
indexRow		
	300	
threshold	0.57	
veParameters		
{th pe	reshold =, 0.	57, amp=, 9 , 0.579894, s





#### Philosophy

Our *a priori* knowledge about a sample can be used in segmentation.

1) Run Part 4, Step 2 and use the human eye to get really good values for threshold, and the gaussian peak parameters for a few slices in the volume.

2) Manually expand the list of good values. In this case, there are 351 rows. Copy closest parameters into rows 1 and 351 for both components. This helps the Interpolation command.

nerpolation command.	index	threshold	amplitude	position	sigma	variance
	1	0.5	3012.71	0.503946	$5.95912 \times 10^{-6}$	172402.
Kel-F	25	0.5	3393.23	0.503751	$6.56236  imes 10^{-6}$	278923.
	50	0.5	3890.22	0.503602	$8.4914  imes 10^{-6}$	244004.
	75	0.5	5387.17	0.503891	$6.60352 \times 10^{-6}$	601834.
	100	0.5	5387.17	0.503891	$6.60352 \times 10^{-6}$	601834.
	351	0.5	5387.17	0.503891	$6.60352  imes 10^{-6}$	601834.
	index	threshold	amplitude	position	sigma	variance
	index 1	0.57	amplitude 211.127	position 0.575044	0.0000539752	385.75
	index 1 110	threshold 0.57 0.57	amplitude 211.127 211.127	position 0.575044 0.575044	sigma 0.0000539752 0.0000539752	385.75 385.75
Zirconia	index 1 110 150	threshold 0.57 0.57 0.57 0.57	amplitude 211.127 211.127 1025.38	position 0.575044 0.575044 0.577505	sigma 0.0000539752 0.0000539752 0.0000283423	variance 385.75 385.75 12251.6
Zirconia	index 1 110 150 200	threshold 0.57 0.57 0.57 0.57 0.57	amplitude 211.127 211.127 1025.38 1024.92	position 0.575044 0.575044 0.577505 0.57856	sigma 0.0000539752 0.0000539752 0.0000283423 0.0000266244	variance 385.75 385.75 12251.6 13466.5
Zirconia	index 1 110 150 200 250	threshold 0.57 0.57 0.57 0.57 0.57 0.57	amplitude 211.127 211.127 1025.38 1024.92 1171.98	position 0.575044 0.575044 0.577505 0.57856 0.57971	sigma 0.0000539752 0.0000539752 0.0000283423 0.0000266244 0.0000188064	variance 385.75 385.75 12251.6 13466.5 7759.85
Zirconia	index 1 110 150 200 250 300	threshold 0.57 0.57 0.57 0.57 0.57 0.57 0.57	amplitude 211.127 211.127 1025.38 1024.92 1171.98 982.606	position 0.575044 0.575044 0.577505 0.57856 0.57971 0.579894	sigma 0.0000539752 0.0000539752 0.0000283423 0.0000266244 0.0000188064 0.000031652	variance 385.75 385.75 12251.6 13466.5 7759.85 10497.6
Zirconia	index 1 110 150 200 250 300 351	threshold 0.57 0.57 0.57 0.57 0.57 0.57 0.57 0.57	amplitude 211.127 211.127 1025.38 1024.92 1171.98 982.606 1129.66	position 0.575044 0.575044 0.577505 0.57856 0.57971 0.579894 0.581064	sigma 0.0000539752 0.0000539752 0.0000283423 0.0000266244 0.0000188064 0.000031652 0.0000251528	variance 385.75 385.75 12251.6 13466.5 7759.85 10497.6 4898.23

3) A cubic-spline interpolation is used to get threshold values and gaussian parameters for any row in between (inclusive) of rows 1 to 351.

index	threshold	amplitude	position	sigma	variance
1	0.57	211.127	0.575044	0.0000539752	385.75
110	0.57	211.127	0.575044	0.0000539752	385.75
150	0.57	1025.38	0.577505	0.0000283423	12251.6
200	0.57	1024.92	0.57856	0.0000266244	13466.5
250	0.57	1171.98	0.57971	0.0000188064	7759.85
300	0.57	982.606	0.579894	0.000031652	10497.6
351	0.57	1129.66	0.581064	0.0000251528	4898.23

thresholdCenter = Interpolation[allParametersZirconia[[All, {1, 2}]]][indexRow]; guessA = Interpolation[allParametersZirconia[[All, {1, 3}]]][indexRow]; guessB = Interpolation[allParametersZirconia[[All, {1, 4}]]][indexRow]; guessC = Interpolation[allParametersZirconia[[All, {1, 5}]]][indexRow]; estVariance = Interpolation[allParametersZirconia[[All, {1, 6}]]][indexRow];

e >



4) Below is a function to calculate even better threshold values for any given row. It uses the interpolated values of threshold and the gaussian peak parameters as initial guesses.

As it turns out, this function is not used.

Step 3: Function to fit zirconia using interpolated starting values for the coefficients.

```
funcBestThresholdZirconia[indexRow_] := Module {},
```

```
thresholdCenter = Interpolation[allParametersZirconia[[All, {1, 2}]]][indexRow];
guessA = Interpolation[allParametersZirconia[[All, {1, 3}]]][indexRow];
guessB = Interpolation[allParametersZirconia[[All, {1, 4}]]][indexRow];
guessC = Interpolation[allParametersZirconia[[All, {1, 5}]]][indexRow];
estVariance = Interpolation[allParametersZirconia[[All, {1, 6}]]][indexRow];
allParameters = { };
slice = volMAS[[indexRow, All, All]];
image = Image[slice, "Bit16"];
imageTV = TotalVariationFilter[GeometricMeanFilter[image, 1], 0.35, Method -> "Laplacian"];
Do Module {},
  imageHighBinary = MorphologicalBinarize[imageTV, {0.0, 0.02} + threshold];
  imageHigh = ImageMultiply[imageTV, imageHighBinary];
  If Max[ImageData[imageHighBinary, "Bit"]] == 1, Module { { },
    temp = Rest[ImageLevels[imageHigh, 100, {-0.1, 0.1} + threshold]];
    (* g1=ListPlot[temp,PlotRange→{{minVolMAS,maxVolMAS},All}]; *)
    index = Flatten[Position[temp[[All, 2]], p_? (# > 0 &)]];
    tempFit = temp[[index, All]];
    If Length[tempFit] > 5, Module {};
      indexTempFit = First[Reverse[Ordering[tempFit[[All, 2]]]];
      indexTempFit = Reverse[Ordering[tempFit[[All, 2]]]];
               (\mathbf{x} - \mathbf{b})^2
```

5) The MAS rotor has some slices which have both zirconia and Kel-F. The bright zirconia is easy to segment with interpolated threshold values.

row=110



6) Then, subtract the binary version of the zirconia from the original image. Does the Kel-F show up now?



# zirconia binary

#### 7) The MAS rotor has some slices which have both zirconia and Kel-F. The bright zirconia is easy to segment with interpolated threshold row=110 values.



#### Kel-F binary

after some erosion

8) It takes a little work to find the right label field. It is usually the largest and it is centered in the middle of the image.

> kelFAreas = ComponentMeasurements [componentsKelF, "Area"]; kelFCentroids = ComponentMeasurements[componentsKelF, "Centroid", (Abs[#[[1]] - columns / 2] < 50 && Abs[#[[2]] - slices / 2] < 50 &)];</pre> indexKelFAreas = Reverse[Ordering[kelFAreas[[All, 2]]]]; indexKelFCentroids = kelFCentroids[[All, 1]]; indexBestKelFComponent = First[Intersection[indexKelFAreas, indexKelFCentroids]]; labelBestKelFComponent = kelFAreas[[indexBestKelFComponent, 1]];

### after closing

label field

#### Philosophy Our *a priori* knowledge about a sample can be used in segmentation.

ImageJ and Manipulate are used to find the rows of Kel-F and zirconia. That row information is used in the If statements.

Timing[ Do[Module[{},

funcMakeZirconiaAndKelFImages[indexRow, columns, slices];

```
If[indexRow ≥ 106, Module[{},
```

zirconiaAreas = ComponentMeasurements[componentsZirconia, "Area"]; indexBestZirconia = First[Reverse[Ordering[zirconiaAreas[[All, 2]]]]; labelBestZirconiaComponent = zirconiaAreas[[indexBestZirconia, 1]]; Do [Module [ $\{x, y\},$ 

{x, y} = coordinatesZirconiaPixels[[i, All]]; ]];

```
If[indexRow \leq 318, Module[{}],
  componentsKelF = MorphologicalComponents[imageKelFBinary];
  kelFAreas = ComponentMeasurements[componentsKelF, "Area"];
  kelFCentroids = ComponentMeasurements[componentsKelF, "Centroid",
    (Abs[#[[1]] - columns / 2] < 50 && Abs[#[[2]] - slices / 2] < 50 &)];</pre>
  indexKelFAreas = Reverse[Ordering[kelFAreas[[All, 2]]];
  indexKelFCentroids = kelFCentroids[[All, 1]];
  indexBestKelFComponent = First[Intersection[indexKelFAreas, indexKelFCentroids]];
  labelBestKelFComponent = kelFAreas[[indexBestKelFComponent, 1]];
  coordinatesKelFPixels = Position[componentsKelF, p_?(# == labelBestKelFComponent &)];
  Do[Module[{x, y}],
    {x, y} = coordinatesKelFPixels[[i, All]];
    segmentedSlice[[x, y]] = 1; ], {i, Length[coordinatesKelFPixels]}];
```

```
]];
```

```
segmentedVolume[[indexRow, All, All]] = segmentedSlice;]
, {indexRow, 1, rows}]
```

```
componentsZirconia = MorphologicalComponents[Closing[imageZirconiaBinary, 5]];
coordinatesZirconiaPixels = Position[componentsZirconia, p_?(# == labelBestZirconiaComponent &)]
```

```
segmentedSlice[[x, y]] = 2; ], {i, Length[coordinatesZirconiaPixels]}];
```