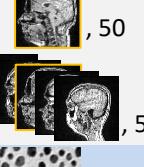
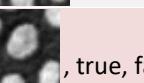
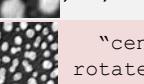
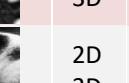




CLIJ2 cheat sheet: ImageJ macro I

GPU-accelerated image processing in Fiji

Operation	Parameters	Result	Dim	Examples
Basics / Wrangling	Initialize CLIJ	[] , HD, GFX or CPU		run("CLIJ Macro Extensions", "cl_device=[]");
	Push		2D 3D	// send current image to GPU input = getTitle(); Ext.CLIJ2_push(input);
	Pull		2D 3D	// get result image from GPU back Ext.CLIJ2_pull(output);
	Create	1024, 1024, 8	2D 3D	Ext.CLIJ2_create2D("new2D", w, h, bitDepth); Ext.CLIJ2_create3D("new3D", w, h, depth, bitDepth);
	Convert		2D 3D	Ext.CLIJ2_convertFloat(input, "result_float"); Ext.CLIJ2_convertUInt8(input, "result_uint8"); Ext.CLIJ2_convertUInt16(input, "result_uint16");
	Copy			// duplicate Ext.CLIJ2_copy(source, result);
	Copy slice	 	2D 3D	// put a slice into a stack Ext.CLIJ2_copySlice(stack, slice, sliceIndex); // copy a slice out of a stack Ext.CLIJ2_copySlice(slice, stack, sliceIndex);
	Crop	 , 20, 20	2D 3D	// crop image Ext.CLIJ2_crop2D("original", "cropped", x, y, width, height);
	Paste	 "clij", 9, 9	2D 3D	// paste image Ext.CLIJ2_paste2D("cropped", "target", x, y);
	Release			// free / release memory occupied by an image Ext.CLIJ2_release("image name");
	Clear			Ext.CLIJ2_clear(); // empty GPU memory
Spatial transforms	Rotate by 90 degrees		2D 3D	Ext.CLIJ2_rotateClockwise(input, result);
	Rotate	 , 45, true	2D 3D	Ext.CLIJ2_rotate2D(input, result, angle, rotateAroundCenter);
	Flip	 , true, false	2D 3D	Ext.CLIJ2_flip2D(input, result, flipX, flipY); Ext.CLIJ2_flip3D(input, result, flipX, flipY, flipZ);
	Translate	 , 20, 20	2D 3D	Ext.CLIJ2_translate2D(input, result, shiftX, shiftY);
	Affine transform	 "center scale=2 rotate=45 -center"	2D 3D	transf = "center scale=2 rotate=45 -center"; Ext.CLIJ2_affineTransform2D(source, result, transf);
	Deform / warp	  	2D 3D	// warp image Ext.CLIJ2_applyVectorField2D(source, vectorFieldX, vectorFieldY, result);
	Projections		3D -> 2D	Ext.CLIJ2_argMaximumZProjection(in, result, arg_z); Ext.CLIJ2_standardDeviationZProjection(in, result);



<https://clij.github.io/clij2-docs>



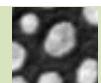
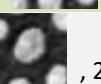
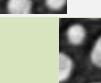
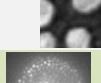
@haesleinhuepf #clij

cheat sheet 2020-04-20



CLIJ2 cheat sheet: ImageJ macro II

GPU-accelerated image processing in Fiji

Operation	Parameters	Result	Dim	Examples
Filters	Gaussian blur	 , 10, 10	2D 3D	Ext.CLIJ2_gaussianBlur2D(input, result, sigmaX, sigmaY);
	Difference of Gaussian	 , 2, 2, 20, 20	2D 3D	Ext.CLIJ2_differenceOfGaussian2D(input, result, sigmaX, sigmaY, sigmaZ);
	Invert		2D 3D	Ext.CLIJ2_invert(input, result);
	Laplace		2D 3D	Ext.CLIJ2_laplaceBox(input, result);
	Mean	 , 5, 5	2D 3D	Ext.CLIJ2_mean2DBox(input, result, radiusX, radiusY);
	Median	 , 5, 5	2D 3D	Ext.CLIJ2_medianSliceBySliceBox(input, result, radiusX, radiusY);
	Minimum	 , 5, 5	2D 3D	Ext.CLIJ2_minimum2DBox(input, result, radiusX, radiusY);
	Maximum	 , 5, 5	2D 3D	Ext.CLIJ2_maximum3DBox(input, result, radiusX, radiusY, radiusZ);
	Top-hat	 , 25, 25, 0	2D 3D	Ext.CLIJ2_topHatBox(input, result, radiusX, radiusY, radiusZ);
Segmentation / labeling	Logarithm / Exponential		2D 3D	Ext.CLIJ2_logarithm(input, result); Ext.CLIJ2_exponential(input, result);
	Threshold	 "Otsu", 127 or	2D 3D	Ext.CLIJ2_threshold(input, binary_result, 127); Ext.CLIJ2_thresholdOtsu(input, binary_result); Ext.CLIJ2_localThreshold(input, threshold_image, binary_result);
	Mask		2D 3D	// mask an image Ext.CLIJ2_mask(input, mask, result);
	Connected components		2D 3D	Ext.CLIJ2_connectedComponentsLabelingBox(binary_in, labelmap_out);
	Label to mask	 , 4	2D 3D	Ext.CLIJ2_labelToMask(labelmap_input, mask_result, label_index);
	Mask labelled	 , 4	2D 3D	Ext.CLIJ2_maskLabel(input, labelmap, result, label_index);
	Exclude on edges		2D 3D	Ext.CLIJ2_
	Label spots		2D 3D	Ext.CLIJ2_labelSpots();
	Label Voronoi		2D 3D	Ext.CLIJ2_labelVoronoiOctagon(labelled_spots, label_voronoi)



<https://clij.github.io/clij2-docs>



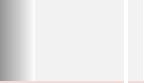
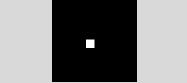
@haesleinhuepf #clij

cheat sheet 2020-04-20



CLIJ2 cheat sheet: ImageJ macro III

GPU-accelerated image processing in Fiji

Operation	Parameters	Result	Dim	Examples	
Math	Set  , 100		2D 3D	Ext.CLIJ2_set(result, pixel_value); Ext.CLIJ2_setRampX(result); Ext.CLIJ2_setColumn(result, column_index, value);	
	Absolute $ x $		2D 3D	Ext.CLIJ2_absolute(input, result);	
	Add / Subtract  or 50		2D 3D	Ext.CLIJ2_addImages(summand1, summand2, result); Ext.CLIJ2_addImageAndScalar(input, result, scalar); Ext.CLIJ2_addImagesWeighted(in1, in2, result, a,b);	
	Multiply / Divide  , 2		2D 3D	Ext.CLIJ2_multiplyImages(input1, input2, result); Ext.CLIJ2_multiplyImageAndScalar(input, result, n); Ext.CLIJ2_divideImages(divident, divisor, result);	
	Multiply Matrix 		2D 3D	Ext.CLIJ2_multiplyMatrix(matrix1, matrix2, matrix_out);	
	Equal = Not Equal !=		2D 3D	Ext.CLIJ2_equal(source1, source2, result); Ext.CLIJ2_notEqual(source1, source2, result);	
	Greater / Smaller		2D 3D	Ext.CLIJ2_greater(source1, source2, result); Ext.CLIJ2_smaller(source1, source2, result); Ext.CLIJ2_smallerOrEqual(source1, source2, result);	
	Equal = Not Equal !=		2D 3D	Ext.CLIJ2_equal(source1, source2, result); Ext.CLIJ2_notEqual(source1, source2, result);	
Binary Images	PullBinary			2D 3D	Ext.CLIJ2_pullBinary(String image);
	Draw line / box / sphere	10, 10, 50, 50		2D 3D	Ext.CLIJ2_drawLine(img, x1, y1, z1, x2, y2, z2, thickness, value); Ext.CLIJ2_drawBox(img, x, y, z, width, height, depth, value); Ext.CLIJ2_drawSphere(img, x, y, z, r_x,r_y,r_z, value); // Pixels apart from the line/box/sphere are untouched!
	Pull regions of interest			2D 3D	Ext.CLIJ2_pullAsROI(binary_image); Ext.CLIJ2_pullToROIManager(binary_image);
	Fill holes			2D 3D	Ext.CLIJ2_binaryFillHoles(source, result)
	Not			2D 3D	Ext.CLIJ2_binaryNot(source, result);
	And / Intersection			2D 3D	Ext.CLIJ2_binaryAnd(operand1, operand2, result); Ext.CLIJ2_binaryIntersection(op1, op2, result);
	Or / Union			2D 3D	Ext.CLIJ2_binaryOr(operand1, operand2, result); Ext.CLIJ2_binaryUnion(operand1, operand2, result);
	XOr			2D 3D	Ext.CLIJ2_binaryXOr(operand1, operand2, result);
	Dilate/ Erode			2D 3D	Ext.CLIJ2_dilateSphere(source, result); Ext.CLIJ2_dilateBox(source, result); Ext.CLIJ2_erodeSphereSliceBySlice(input, result);





CLIJ2 cheat sheet: ImageJ macro IV

Operation	Parameters	Result	Dim	Examples	
Vectors, arrays, matrices, graphs & meshes	Spots to point lists			2D 3D	Ext.CLIJ2_spotsToPointlist(binary_spots, pointlist); Ext.CLIJ2_labelledSpotsToPointlist(labelled_spots, pointlist);
	Generate distance matrix			2D 3D	Ext.CLIJ2_generateDistanceMatrix(pointlist1, pointlist2, distance_matrix);
	Generate touch matrix			2D 3D	Ext.CLIJ2_generateTouchMatrix(label_map, touch_matrix);
	Touch matrix to mesh			2D 3D	Ext.CLIJ2_touchMatrixToMesh(pointlist, touch_matrix, mesh);
	Distance matrix to mesh			2D 3D	Ext.CLIJ2_distanceMatrixToMesh(pointlist, distance_matrix, mesh, max_distance);
	Mean of touching neighbors			2D 3D	Ext.CLIJ2_meanOfTouchingNeighbors(values, touch_matrix, mean_values);
	Count touching neighbors			2D 3D	Ext.countTouchingNeighbors(touch_matrix, count_vector)
Working with arrays and tables	Statistics	 		2D 3D	Ext.CLIJ2_statisticsOfBackgroundAndLabelledPixels(image, labelmap); Ext.CLIJ2_statisticsOfLabelledPixels(input, labelmap);
	Push Results Table			2D 3D	Ext.CLIJ2_pushResultsTable(image_name);
	Push Results table column			2D 3D	Ext.CLIJ2_pushResultsTableColumn(image_name, column_name)
	Pull to Results table			2D 3D	Ext.CLIJ2_pullToResultsTable(image_name);
	Push Array	[1,4,0,0,0,2], 3, 2, 1		2D 3D	CLIJ2_pushArray(image_name, array, width, height, depth);

Detailed documentation

CLIJ documentation can be found

- in CLIJs dialogs under the menu Plugins > ImageJ on GPU (CLIJ2)
 - Embedded in Fijis script editor – just start typinh
 - and online: <https://clij.github.io/clij2-docs>

The screenshot shows a Windows-style dialog box titled "CLII2_gaussianBlur2D". It contains the following fields:

- CL Device: A dropdown menu set to "gt900".
- source: A dropdown menu set to "blobs.gif".
- sigmaX: An input field containing "2.00".
- sigmaY: An input field containing "2.00".

Below the dialog box, the text reads: "Computes the Gaussian blurred image of an image given two sigma values in X and Y. Thus, the filterkernel can have non-isotropic shape. The implementation is done separable. In case a sigma equals zero, the direction is not blurred. Parameters: Image source, ByRef Image destination, Number sigmaX, Number sigmaY Available for: 2D".

Installation instructions

- Install CLIJ2 by activating the “clij” and “clij2” update sites in Fiji.
 - Commands listed as “CLIJx” are experimental should be handled with care. They may change or disappear at any point. To build reliable, reproducible workflows use CLIJ or CLIJ2 commands only.



The screenshot shows a browser window displaying the `absolute(inplace)` function from the `CLIJ2 (alpha release)` documentation. The page includes the function signature, parameters, and examples. Below the documentation, there is a small image of a logo consisting of the letters 'c' and 'ij' in a stylized font.

`[A,B],[C,D][E,F][G,H][I,J][K,L][M,N][O,P],[Q,R][S,T][U,V][W]`

`X,Y,Z`

A

`absolute(inplace)` (Experimental)

Computes the absolute value of every individual pixel *x* in a given image.

`absolute`

Computes the absolute value of every individual pixel *x* in a given image.

`addImage` and `Scalar`

Adds a scalar value *s* to all pixels *x* of a given image *X*.

`addImageWeighted`

Calculates the sum of pairs of pixels *x* and *y* from images *X* and *Y* weighted with factors *a* and *b*.

`addImages`

Calculates the sum of pairs of pixels *x* and *y* of two images *X* and *Y*.



<https://clii.github.io/clii2-docs>



@haesleinphuenf #clii

sheet sheet 2020-04-20