xray.jit

a collection of jitter objects, absractions and javascripts

by Wesley Smith
http://www.mat.ucsb.edu/~whsmith/
wesley.hoke@gmail.com
January 2, 2006

Introduction

The xray.jit objects were created to extend the functionality of jitter for particular projects I have worked on. As such, some of the objects are quite specific to my needs at the time. There are, however, some objects that are good for everyday patching. The objects I use the most are xray.jit.timecube, xray.jit.grid2tri, and xray.jit.quicksort.

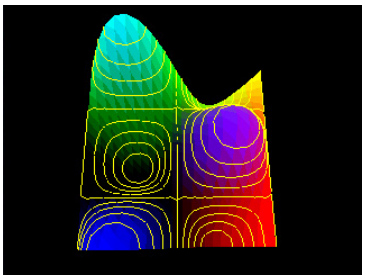
I have divided the objects into several broad categories pertaining to **Geometry**, **Data**, **Video**, and **Simulation**. The Geometry objects deal more with OpenGL situations. The Data objects operate on jitter matrices without consideration for what that data necessarily represents and are more geared toward the general matrix format. The Video objects on the other hand mostly deal with data that is in one of the common video formats such as 1 plane char or 4 plane char. Finally, the Simulation objects do physical simulation of some sort.

I hope you find these objects useful and always appreciate any comments, especially bug reports.

Installation

Jitter Objects go in *C74\jitter-externals\xray*
Jitter Help Files go in *MaxMSP\max-help\jitter\xray-help*
Javascripts go in *C74\jsextensions\xray-js*
xray.about and **xray.jit.pct** go in *C74\patches\extras\xray-extras*

1 Geometry



xray.jit.contourmap

Calculates the isolines of a surface in 3D. The output is a row of pairs of points defining line segments that make up the isolines. The lines are not output in order. That is to say you can't count on neighboring pairs to form a continuous curve. The algorithm used is based on the CONREC contouring algorithm by Paul Bourke.

attributes:

mode [0, 1]:

0: set isolevels based on levels attribute with each level equally spaced between the min and max of the surface

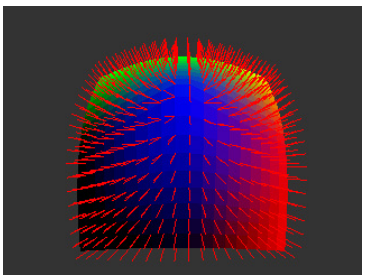
1: set isolevels based on isolevel attribute list

levels [1, n]:

the number of evenly spaced isolevels between Zmin and Zmax

isolevel list[10]:

a list of up to 10 isolevels at specific z-values

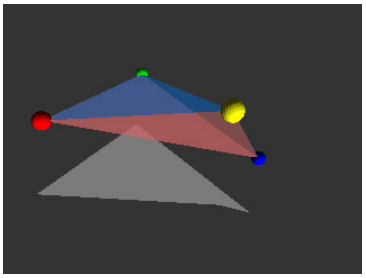


xray.jit.crossproduct

Calculates the crossproduct of two vectors. The input matrices define vectors originating at the origin, so they are only 3 planes.

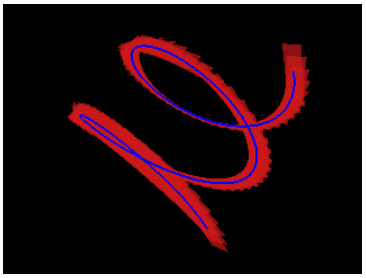
attributes:

normalize list[2] [0, 1]:
(first inlet, second inlet)
0: normalization off
1: normalization on



xray.jit.grid2tri

Transforms a set of vertices arranged in a grid fashion into explicit triangles. When rendering surfaces with draw_mode tri_grid or quad_grid, the way the vertices are connected with triangles is ambiguous and graphics card dependent. In order to explicitly set the way vertices are connected, you must draw the surface with the triangle draw_mode.



xray.jit.line2quad

Transforms a set of lines into quads. This is useful for texturing geometries described by lines.

attributes:

mode [0, 1]:
0: calculate quad in 2D
1: calculate quad in 3D

scale list[2]:
scale of width and height of quads

normalize list[2]:
normalize length before scaling

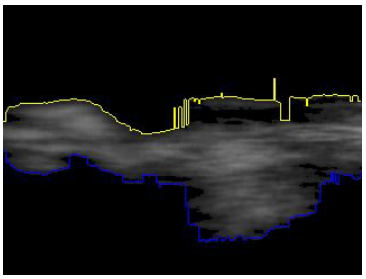
2 Data

xray.jit.cellcoords

gives the coordinates corresponding to 'logic 1. of a 2D matrix where a 'logic 1' is defined as 255 for char, 1 for long, 1.0 for float32 and float64.

attributes:

plane [0, planes-1]:
plane to look in
out_mode [0, 1]:
0: long output
1: float32 output

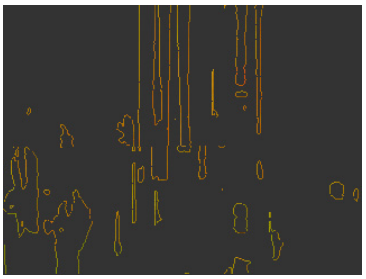


xray.jit.cellminmax

Scans a matrix bidirectionally and reports coordinates of first non-zero cell.

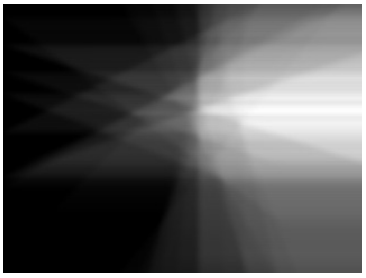
attributes:

mode [0, 1]:
0: horizontal scan
1: vertical scan



xray.jit.cellvalue

Gets the values of a specified cell based on an arbitrary list of cell coordinates. The first input is an arbitrary 2D matrix and the second input is a 2-plane, float32 matrix of xy-coordinates of cells to get.



xray.jit.cumsum

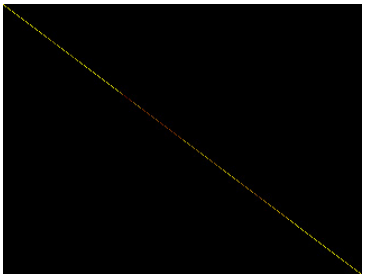
Accumulates matrix values across a dimension.

attributes:

dimmode [0, 1]:

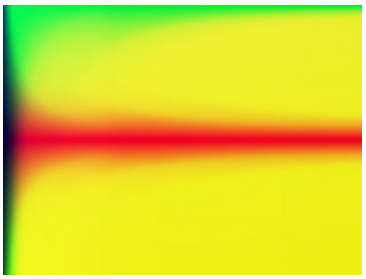
0: accumulate from left to right

1: accumulate from top to bottom



xray.diagonalize.js

Diagonalizes a matrix.



xray.dynamicexpr.js

A more intuitive way to dynamically alter an expression for jit.expr. Uses the standard max message variable format (1,2, ...) for choosing elements from a list (including symbols). The object must first be initialized with an expression specified with the 'expr' message following the exact same conventions of the jit.expr 'expr' message meaning that expressions for specific planes are seperated by quotes as demonstrated in this patch. Also, the ', ' character must be escaped (' ') as with all max messages. When floats, ints, or lists are sent to xray.dynamicexpr.js, a list of symbols is output with the appropriate substitutions made.

attributes:

expr:

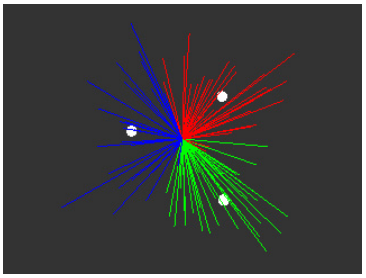
jit.expr-like message extended to accept message substitutions

list:

values to substitute in expression

xray.jit.keepcell

Keeps only those pixels specified by the second inlet matrix.



xray.jit.kmeans

Cluster vectors according to the k-means algorithm. Initially, one input vector is arbitrarily assigned to each cluster. The distance between all vectors and these cluster vectors is then calculated. Each vector is then assigned to its closest cluster and the centroid of each cluster is calculated. Then the algorithm is repeated by recalcualteng distances to centroids and resaaigning vectors to clusters. Once the centroids stop moving, the algorithm is finished.

attributes:

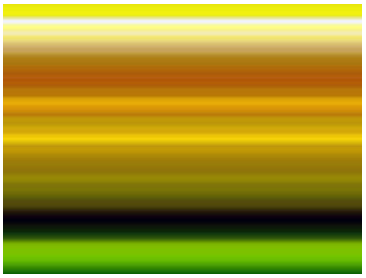
dimmode [0, 1]:

0: vertical vectors

1: horizontal vectors

clusters [1, n]:

the number of clusters to form



xray.jit.mean

Calculates the mean/sum across a 2D matrix, returning either an nx1 or 1xn matrix.

attributes:

meandim [0, 1]:

0: sum across dim[0], outputs a 1xdim[1] matrix

1: sum across dim[1], outputs a dim[0]x1 matrix

mean [0, 1]:

0: calculate the sum of a row/column

1: calculate the mean of a row/column

xray.jit.pinv

Computes the pseudo-inverse of a matrix. It takes as input the V, Sigma, and U matrices that xray.jit.svd outputs. The pseudo inverse of a matrix can be calculated from the Singular Value Decomposition of a matrix where svd(A) = U*Simga*V^T. The pseudo inverse of A is then pinv(A)=V* $Sigma^{-1}$*U^T. Once the SVD of a matrix is performed, its components are often used several times. Since the SVD is a fairly expensive operation, the decision was made to not include the calculation of the SVD in the abstraction although using this abstraction without the SVD is pointless.

xray.jit.quicksort

Sorts a matrix using the quicksort algorithm. See: http://www.iti.fh-flensburg.de/lang/algorithmen/sortieren/quick/quicken.htm

attributes:

dimmode [0, 1]:

0: horizontal sort

1: vertical sort

planemode [0, planes-1]:
specifies which plane's values to use when sorting

summode [0, 1]:
0: sort based on planemode
1: sort based on sum of planes

recursions [-1, n]:
specifies the max recursion depth of the quicksort algorithm. A value of -1 is equivalent to an infinite max.

iterations [-1, n]:
specifies the max number of iterations of the quicksort algorithm. A value of -1 is eqivalent to an infinite max.

splitpoint [0, dim-1]:
specifies the inital location to divide the data to be sorted. Negative values return to the default, values larger than the dimsize-1 are clipped.

xray.jit.rank

Calculates the rank of a matrix. Takes the Sigma matrix from xray.jit.svd as input. Sometimes the numerical solution to the SVD gives eigenvalues within the roud off error of the algorithm. In these cases, the eigenvector's value is irrelevant and should be discarded when calculating rank of a matrix.

arguments:
1:*threshold* (float):
eigenvalue threshold

xray.jit.sift

sifts out rows/columns that aren't marked by a 'logic 1' where a 'logic 1' is defined as 255 for char, 1 for long, 1.0 for float32 and float64.

attributes:
siftdim [0, 1]:
0: check top row for 'logic 1'
1: check left column for 'logic 1'

siftplane [1, planes-1]:
plane to look at for a 'logic 1'

xray.jit.svd

Computes the singular value decomposition of a matrix. Takes 1-plane 2D matrices of any type and outputs 3 1-plane 2D float32 or float64 matrices. For input A, an mxn matrix, the outputs are matrices U (nxm), Sigma (nx1), and V (nxn). The single value decomposition of a matrix breaks the matrix down into fundamental components: eigenvectors and eigenvalues. The U or left-hand matrix gives the eigenvectors spanning column space and the V or right-hand matrix gives the eigenvectors spanning row space. The vectors in both U and V are orthonormal. The Sigma matrix contains the eigenvalues of the matrix. Sigma is a diagonal matrix although xray.jit.svd outputs this diagonal matrix as a vector since every value in the matrix except values along the diagonal are zero. xray.jit.svd always outputs the matrices U, Sigma, and V with the largest eignevalue and corresponding eigenvector in the left-most position and the rest in descending order.

---

3 Video

xray.jit.3dbuffer

Multitap circular buffer for creating video delay lines.

arguments:

1:*planes*:
3D matrix planecount

2:*type*:
3D matrix type

3:*dim[0]*:
3D matrix width

4:*dim[1]*:
3D matrix height

5:*dim[2]*:
number of frames to store

6:*name*:
3D matrix name

xray.jit.colormap

Colors a grayscale matrix with the colors of another matrix. xray.jit.colormap takes a standard video matrix and generates a colormap for use with jit.charmap for coloring grayscale images. In mode 0, all of the values of the incoming

matrix a binned according to luminance accumulatively and then averaged to derive the colormap. In mode 1, only the first color value of a particular luminance is binned. In both modes, if there isn't a particular color value associated with a luminance bin, the map value is interpolated from the nearest existing values.

attributes:

mode [0, 1]:

0: average color

1: first color

xray.jit.distance

Computes the distance transform on a grayscale image. xray.jit.distance uses the L1 norm aka city block distance. See: http://people.cs.uchicago.edu/ pff/dt/

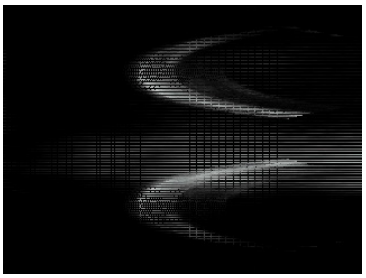
attributes:

mode [0, 1]:

0: horizontal traversal only

1: vertical traversal only

2: horizontal and vertical traversal



xray.jit.invertrepos

Inverts a repos map in absolute mode. Since most repos maps aren't 1 to 1, some information is lost, which becomes evident after the inversion process.

xray.jit.timecube

Arbitrary 3D buffer manipulator for combining pixels from multiple frames at once according to a given map.

attributes:

mode [0, 1, 2]:

0: vertical plane (aka Vasulka mode)

1: horizontal plane

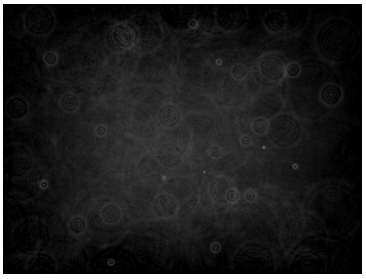
2: map mode

sync (# of frames):

current slice of the 3D buffer being written to

---

4 Simulation



xray.jit.water

Simulates a surface of water using the Finite Element Method. xray.jit.water implements a finite element simulation of a water surface. The algorithm used was modified from:

`http://www.u-aizu.ac.jp/~niki/papers/2005\%20Fast\%20water\%20animation.pdf`

. Since this is a FEM simulation, the more elements (i.e. cells), the more stable the simulation will be. Almost anything below a grid size of 100x100 is very unstable. The second inlet is similar to the second inlet of jit.op in that it can take both floats and matrices. Since the second inlet sets wavespeed, it can be used to simulate water of different depths. For instance, shallow water has a higher wavespeed than deep water. This object relies on feedback

to run the simulation, sohaving a the same named matrix at both input and output as demonstrated here is required for proper use.

attributes:

spacestep (float):
resolution of the spatial derivatives in the FEM

timestep (float):
resolution of the temporal derivative in the FEM

damping (float):
surface stiffness

2nd inlet (float) or (jit_matrix):
wavespeed