Matrix Processing

PyoObjects to perform operations on PyoMatrixObjects.

PyoMatrixObjects are 2 dimensions table containers. They can be used to store audio samples or algorithmic sequences. Writing and reading are done by giving row and column positions.

MatrixMorph

class **MatrixMorph**(*input*, *matrix*, *sources*) [source]

Morphs between multiple PyoMatrixObjects.

Uses an index into a list of PyoMatrixObjects to morph between adjacent matrices in the list. The resulting morphed function is written into the *matrix* object at the beginning of each buffer size. The matrices in the list and the resulting matrix must be equal in size.

Parent: PyoObject

Args: input: PyoObject

Morphing index between 0 and 1. 0 is the first matrix in the list and 1 is the last.

matrix: NewMatrix

The matrix where to write morphed function.

sources: list of PyoMatrixObject

List of matrices to interpolate from.

Note: The out() method is bypassed. MatrixMorph returns no signal.
MatrixMorph has no *mul* and *add* attributes.

```
>>> s = Server().boot()
>>> s.start()
>>> m1 = NewMatrix(256, 256)
>>> m1.genSineTerrain(1, 4)
>>> m2 = NewMatrix(256, 256)
>>> m2.genSineTerrain(2, 8)
>>> mm = NewMatrix(256, 256)
>>> inter = Sine(.2, 0, .5, .5)
>>> morph = MatrixMorph(inter, mm, [m1,m2])
>>> x = Sine([49,50], 0, .45, .5)
>>> y = Sine([49.49,50.5], 0, .45, .5)
>>> c = MatrixPointer(mm, x, y, .2).out()
```

setInput(*x*, *fadetime=0.05*) [source]

Replace the *input* attribute.

Args: x: PyoObject

New signal to process.

fadetime: float, optional

Crossfade time between old and new input. Defaults to 0.05.

setMatrix(*x*) [source]

Replace the *matrix* attribute.

Args: x: NewMatrix

new *matrix* attribute.

setSources(*x*) [source]

Replace the *sources* attribute.

Args: x: list of PyoMatrixObject

new *sources* attribute.

input

PyoObject. Morphing index between 0 and 1.

matrix

NewMatrix. The matrix where to write samples.

sources

list of PyoMatrixObject. List of matrices to interpolate from.

MatrixPointer

class **MatrixPointer**(*matrix*, *x*, *y*, *mul=1*, *add=0*) [source]

Matrix reader with control on the 2D pointer position.

Parent: PyoObject

Args: matrix: PyoMatrixObject

Matrix containing the waveform samples.

x: PyoObject

Normalized X position in the matrix between 0 and 1.

y: PyoObject

Normalized Y position in the matrix between 0 and 1.

```
>>> s = Server().boot()
>>> s.start()
>>> SIZE = 256
```

```
>>> mm = NewMatrix(SIZE, SIZE)
>>> fmind = Sine(.2, 0, 2, 2.5)
>>> fmrat = Sine(.33, 0, .05, .5)
>>> aa = FM(carrier=10, ratio=fmrat, index=fmind)
>>> rec = MatrixRec(aa, mm, 0).play()
>>> lfx = Sine(.1, 0, .24, .25)
>>> lfy = Sine(.15, 0, .124, .25)
>>> x = Sine([500,501], 0, lfx, .5)
>>> y = Sine([10.5,10], 0, lfy, .5)
>>> c = MatrixPointer(mm, x, y, .2).out()
```

setMatrix(*x*) [source]

Replace the *matrix* attribute.

Args: x: PyoTableObject
new *matrix* attribute.

setX(*x*) [source]

Replace the *x* attribute.

Args: x: PyoObject
new *x* attribute.

setY(*x*) [source]

Replace the *y* attribute.

Args: y: PyoObject
new *y* attribute.

matrix

PyoMatrixObject. Matrix containing the samples.

x

PyoObject. Normalized X position in the matrix.

y

PyoObject. Normalized Y position in the matrix.

MatrixRec

class **MatrixRec**(*input*, *matrix*, *fadetime=0*, *delay=0*) [source]

MatrixRec records samples into a previously created NewMatrix.

See **NewMatrix** to create an empty matrix.

The play method is not called at the object creation time. It starts the recording into the matrix, row after row, until the matrix is full. Calling the play method again restarts the recording and overwrites

previously recorded samples. The stop method stops the recording. Otherwise, the default behaviour is to record through the end of the matrix.

Parent: `PyoObject`

Args: input: PyoObject

Audio signal to write in the matrix.

matrix: PyoMatrixObject

The matrix where to write samples.

fadetime: float, optional

Fade time at the beginning and the end of the recording in seconds. Defaults to 0.

delay: int, optional

Delay time, in samples, before the recording begins. Available at initialization time only. Defaults to 0.

Note: The out() method is bypassed. MatrixRec returns no signal.
MatrixRec has no *mul* and *add* attributes.

MatrixRec will send a trigger signal at the end of the recording. User can retreive the trigger streams by calling obj['trig']. See *TableRec* documentation for an example.

See also: `NewMatrix`

```
>>> s = Server().boot()
>>> s.start()
>>> SIZE = 256
>>> mm = NewMatrix(SIZE, SIZE)
>>> fmind = Sine(.2, 0, 2, 2.5)
>>> fmrat = Sine(.33, 0, .05, .5)
>>> aa = FM(carrier=10, ratio=fmrat, index=fmind)
>>> rec = MatrixRec(aa, mm, 0).play()
>>> lfx = Sine(.1, 0, .24, .25)
>>> lfy = Sine(.15, 0, .124, .25)
>>> x = Sine([500,501], 0, lfx, .5)
>>> y = Sine([10.5,10], 0, lfy, .5)
>>> c = MatrixPointer(mm, x, y, .2).out()
```

setInput(*x*, *fadetime=0.05*) [source]

Replace the *input* attribute.

Args: x: PyoObject

New signal to process.

fadetime: float, optional

Crossfade time between old and new input. Defaults to 0.05.

setMatrix(*x*) [source]

Replace the *matrix* attribute.

Args: x: NewMatrix
new *matrix* attribute.

input

PyoObject. Audio signal to record in the matrix.

matrix

PyoMatrixObject. The matrix where to write samples.

MatrixRecLoop

class **MatrixRecLoop**(*input*, *matrix*) [source]

MatrixRecLoop records samples in loop into a previously created NewMatrix.

See `NewMatrix` to create an empty matrix.

MatrixRecLoop records samples into the matrix, row after row, until the matrix is full and then loop back to the beginning.

Parent: `PyoObject`

Args: input: PyoObject
Audio signal to write in the matrix.

matrix: PyoMatrixObject
The matrix where to write samples.

Note: The out() method is bypassed. MatrixRecLoop returns no signal.
MatrixRecLoop has no *mul* and *add* attributes.

MatrixRecLoop will send a trigger signal when reaching the end of the matrix. User can retreive the trigger streams by calling obj['trig']. See *TableRec* documentation for an example.

See also: `NewMatrix`

```
>>> s = Server().boot()
>>> s.start()
>>> env = CosTable([(0,0), (300,1), (1000,.4), (8191,0)])
>>> matrix = NewMatrix(8192, 8)
>>> src = SfPlayer(SNDS_PATH+'/transparent.aif', loop=True, mul=.3)
>>> m_rec = MatrixRecLoop(src, matrix)
>>> period = 8192 / s.getSamplingRate()
>>> metro = Metro(time=period/2, poly=2).play()
>>> x = TrigLinseg(metro, [(0,0), (period,1)])
```

```
>>> y = TrigRandInt(metro, max=2, mul=0.125)
>>> amp = TrigEnv(metro, table=env, dur=period)
>>> out = MatrixPointer(matrix, x, y, amp).out()
```

setInput(*x*, *fadetime=0.05*) [source]

Replace the *input* attribute.

Args: x: PyoObject

New signal to process.

fadetime: float, optional

Crossfade time between old and new input. Defaults to 0.05.

setMatrix(*x*) [source]

Replace the *matrix* attribute.

Args: x: NewMatrix

new *matrix* attribute.

input

PyoObject. Audio signal to record in the matrix.

matrix

PyoMatrixObject. The matrix where to write samples.