

Mathematica Module for ARAP Interpolation

MARAP User's Manual

Version 0.24

22 November 2016

Tomomi Hirano, Kyushu University.

Yoshihiro Mizoguchi, Kyushu University.

Ver. 0.1 13 September 2016

Table of Contents

| | | |
|----------|-----------------------------------|-----------|
| 1 | Introduction | 1 |
| 2 | Basic Functions | 2 |
| 2.1 | MakePolygon | 2 |
| 2.2 | HorizontalSnake | 2 |
| 2.3 | VerticalSnake | 3 |
| 2.4 | SnakeTriangle | 3 |
| 2.5 | AnimationRange | 3 |
| 2.6 | AnimationRange2 | 4 |
| 2.7 | Polar | 4 |
| 2.8 | Cartesian | 4 |
| 2.9 | LinearInterpolate | 4 |
| 2.10 | LinearInterpolation | 5 |
| 2.11 | PolarInterpolate | 5 |
| 2.12 | PolarInterpolation | 6 |
| 2.13 | LinearInterpolateSnake2D | 6 |
| 2.14 | PolarInterpolateSnake2D | 6 |
| 2.15 | PolarDecomposition | 7 |
| 2.16 | PolarDecompositionPlus | 7 |
| 2.17 | RotateAngle | 7 |
| 2.18 | CogTrans | 8 |
| 2.19 | PolygonToTriangles | 8 |
| 2.20 | TrianglesToTriangles | 10 |
| 2.21 | TrianglesToPolygon | 10 |
| 2.22 | ValNames | 10 |
| 2.23 | NormF | 11 |
| 3 | Matrix Functions | 12 |
| 3.1 | QuadraticFormVariableMatrix | 12 |
| 3.2 | Div2if | 12 |
| 3.3 | Div2Matrix | 12 |
| 3.4 | QuadraticFormMatrix | 12 |
| 3.5 | LinearFormVector | 13 |
| 3.6 | VtoTriangle | 13 |
| 3.7 | VtoTriangles | 13 |
| 3.8 | Cog | 13 |
| 3.9 | Trans | 14 |
| 3.10 | FindMatrix | 14 |
| 3.11 | FindMatrix1 | 14 |
| 3.12 | FindMatrices | 14 |
| 3.13 | FindAffineMatrix | 15 |
| 3.14 | FindAffineMatrices | 15 |

| | | |
|----------|------------------------------------|-----------|
| 3.15 | F1a | 15 |
| 3.16 | F2a | 16 |
| 3.17 | EmbedMatrix | 16 |
| 3.18 | EmbedMatrix | 16 |
| 3.19 | EmbedMatrix2 | 17 |
| 3.20 | F1v | 17 |
| 3.21 | EmbedVector | 18 |
| 4 | Local Interpolations | 19 |
| 4.1 | RotateAngle | 19 |
| 4.2 | NewFindMatrices | 19 |
| 4.3 | LocalLinear | 19 |
| 4.4 | LocalPolar | 19 |
| 4.5 | LocalAlexa | 19 |
| 4.6 | LocalLogExp | 20 |
| 4.7 | LocalInterpolations | 20 |
| 5 | Constraint Functions | 21 |
| 5.1 | ConstMatrix | 21 |
| 5.2 | ConstMatrixM | 21 |
| 5.3 | ConstVectorM | 21 |
| 5.4 | ConstfixMatrix | 22 |
| 5.5 | ConstfixVector | 22 |
| 5.6 | ConstPair | 22 |
| 5.7 | ConstPair | 23 |
| 5.8 | DoubleMatrix | 23 |
| 6 | Grobal Interpolations | 24 |
| 6.1 | QuadraticFormAlexa | 24 |
| 6.2 | QuadraticFormSim | 24 |
| 6.3 | ARAP | 24 |
| 7 | Draw Animation | 25 |
| 7.1 | ShowStatus | 25 |
| 7.2 | DrawAnimation | 25 |
| 7.3 | ListAnimation | 26 |
| | Index | 28 |

1 Introduction

This is a Mathematica Module for investigating an interpolation in Computer Graphics, called the ARAP (as rigid as possible) interpolation. Our module contains about 100 functions of elementary matrix operations, matrix and polygon interpolations and drawing polygons.

To use this package "ARAPlibv024" and sample data "ARAPdata021" users should set a directory where the modules are stored.

[Example]

```
SetDirectory[FileNameJoin[$HomeDirectory, "--- Some Folder ---"]];
<< ARAPlibv024';
<< ARAPdata021';
```

This module was used and introduced in the followings:

- [1] S.Kaji, S.Hirose, S.Sakata, Y.Mizoguchi, Mathematical analysis on affine maps for 2D shape interpolation (<http://dl.acm.org/citation.cfm?id=2422368>), SCA '12 Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp.71-76.
- [2] Y. Mizoguchi, Mathematical Aspects of Interpolation Technique for Computer Graphics, Forum "Math-for-Industry" 2012, Information Recovery and Discovery, 22 October 2012. <http://fmi2012.imi.kyushu-u.ac.jp/>
- [3] T.Hirano, A.hirakawa, N.Miyaki, C.Matsufuji and Y.Mizoguchi. Mathematica Module for ARAP Interpolation
<https://github.com/KyushuUniversityMathematics/MathematicaARAP>

2 Basic Functions

2.1 MakePolygon

`MakePolygon[V, tindex]`
 :: Make polygon

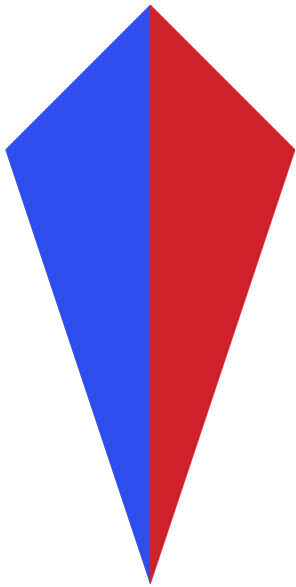
V vertex set

tindex constitution of triangle

return polygon

[Example]

`Graphics[MakePolygon[dataS2, dataT2]]`



2.2 HorizontalSnake

`HorizontalSnake[n]`
 :: Make vertexes of horizontally long rectangle

n rectangle size

return vertex set of rectangle

[Example]

`HorizontalSnake[5]`

`={{0, 0}, {0, 1}, {1, 0}, {1, 1}, {2, 0}, {2, 1}, {3, 0}, {3, 1}, {4, 0}, {4, 1}}`

2.3 VerticalSnake

```
VerticalSnake[n]
    :: Make vertexes of vertically long rectangle
n
    rectangle size
return
    vertex set
[Example]
VerticalSnake[5]
= {{1, 0}, {0, 0}, {1, 1}, {0, 1}, {1, 2}, {0, 2}, {1, 3}, {0, 3}, {1, 4}, {0, 4}}
```

2.4 SnakeTriangle

```
SnakeTriangle[n]
    :: Triangulate Horizontal/Vertex Snake
n
    rectangle size
return
    constitution of triangle
[Example]

Graphics[MakePolygon[HorizontalSnake[10], SnakeTriangle[10]]]
```



```
Graphics[MakePolygon[VerticalSnake[10], SnakeTriangle[10]]]
```



2.5 AnimationRange

```
AnimationRange[conf]
    :: size of Display conforming with polygon
conf
    { start vertex set , end vertex set }
return
    coordinates of display
```

[Example]

```
AnimationRange[Configuration2]
={{-2, 4}, {-3, 5}}
```

2.6 AnimationRange2

```
AnimationRange2[range1, range2]
:: size of Display
```

range1

range2

return coordinates of display

[Example]

```
AnimationRange[Configuration2]
={{-2, 4}, {-3, 5}}
```

2.7 Polar

```
Polar[a] :: Convert cartesian coordinates to polar coordinates
```

a cartesian coordinates

return polar coordinates

[Example]

```
Polar[{0, 1}]
={1, 2 $\pi$ }
```

2.8 Cartesian

```
Cartesian[a]
:: Convert polar coordinates to cartesian coordinates
```

a polar coordinates

return cartesian coordinate

[Example]

```
Cartesian[{1, 2 $\pi$ }]
={0, 1}
```

2.9 LinearInterpolate

```
LinearInterpolate[a, b, t]
:: Perform linear interpolation of point
```

a a start point

b an end point

t time

return an interpolated point

[Example]

```
LinearInterpolate[{1, 1}, {2, 2}, 0.5]
={1.5, 1.5}
```

2.10 LinearInterpolation

`LinearInterpolation[p,q,t]`
 :: Perform linear interpolation of vertex set

p start vertex set

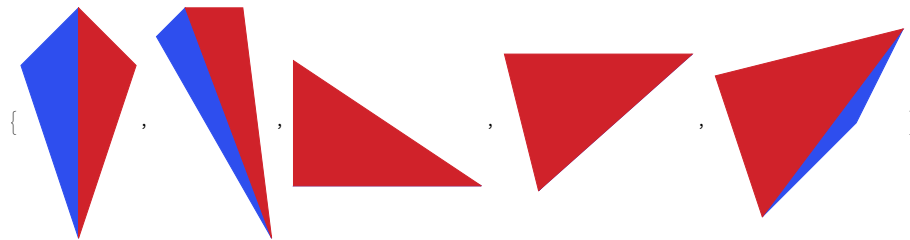
q end vertex set

t time

return interpolated vertex set

[Example]

```
Graphics[MakePolygon[LinearInterpolation[dataS2, dataE2, #], dataT2]] & /
{0, 0.25, 0.5, 0.75, 1}
```



2.11 PolarInterpolate

`PolarInterpolate[a,b,t]`
 :: Perform Polar interpolation of point

a a start point

b an end point

t time

return an interpolated point

[Example]

```
PolarInterpolate[{1, 1}, {2, 2}, 0.5]
={1.5, 1.5}
```

2.12 PolarInterpolation

`PolarInterpolation[p,q,t]`
 :: Perform Polar interpolation of vertex set

p start vertex set

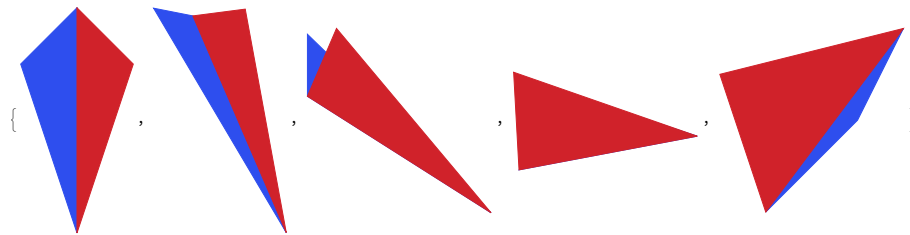
q end vertex set

t time

return interpolated vertex set

[Example]

```
Graphics[MakePolygon[PolarInterpolation[dataS2, dataE2, #], dataT2]] & /
{0, 0.25, 0.5, 0.75, 1}
```



2.13 LinearInterpolateSnake2D

`LinearInterpolateSnake2D[n,t]`
 :: aiueo

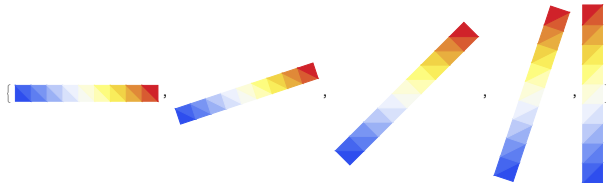
n Snake size

t time (0~1)

return interpolated snake rectangle

[Example]

```
Graphics[LinearInterpolateSnake2D[10, #]] & / {0, 0.25, 0.5, 0.75, 1}
```



2.14 PolarInterpolateSnake2D

`PolarInterpolateSnake2D[n,t]`
 :: aiueo

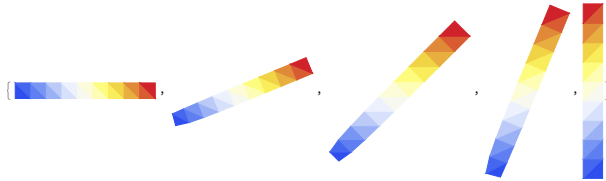
n Snake size

t time (0~1)

return interpolated snake rectangle

[Example]

```
Graphics[PolarInterpolateSnake2D[10, #]] & / {0, 0.25, 0.5, 0.75, 1}
```



2.15 PolarDecomposition

`PolarDecomposition[m]`

:: Perform polar decomposition

m matrix

return orthogonal matrix and positive-semidefinite matrix

[Example]

```
PolarDecomposition[{{1, 1}, {-1, -1}}]
```

```
={{0., 1.}, {-1., 0.}}, {{1., 1.}, {1., 1.}}
```

2.16 PolarDecompositionPlus

`PolarDecompositionPlus[m]`

:: aiueo

m aiueo

return aiueo

[Example]

```
PolarDecompositionPlus[m_]
```

2.17 RotateAngle

`RotateAngle[m]`

:: Compute angle of Rotation matrix that is performed polar decomposition

m matrix

return angle

[Example]

```
RotateAngle[{{1, 2}, {3, 4}}]
```

```
=ArcTan[3]
```

2.18 CogTrans

`CogTrans[pl]`
 :: move to triangles (trigangles center og gravity(0,0))

pl list of a coordinate triangle

return list of a coordinate triangle

[Example]
`CogTrans[{{0,3},{3,3},{0,-3}}]`
`={{-1, 2}, {2, 2}, {-1, -4}}`

2.19 PolygonToTriangles

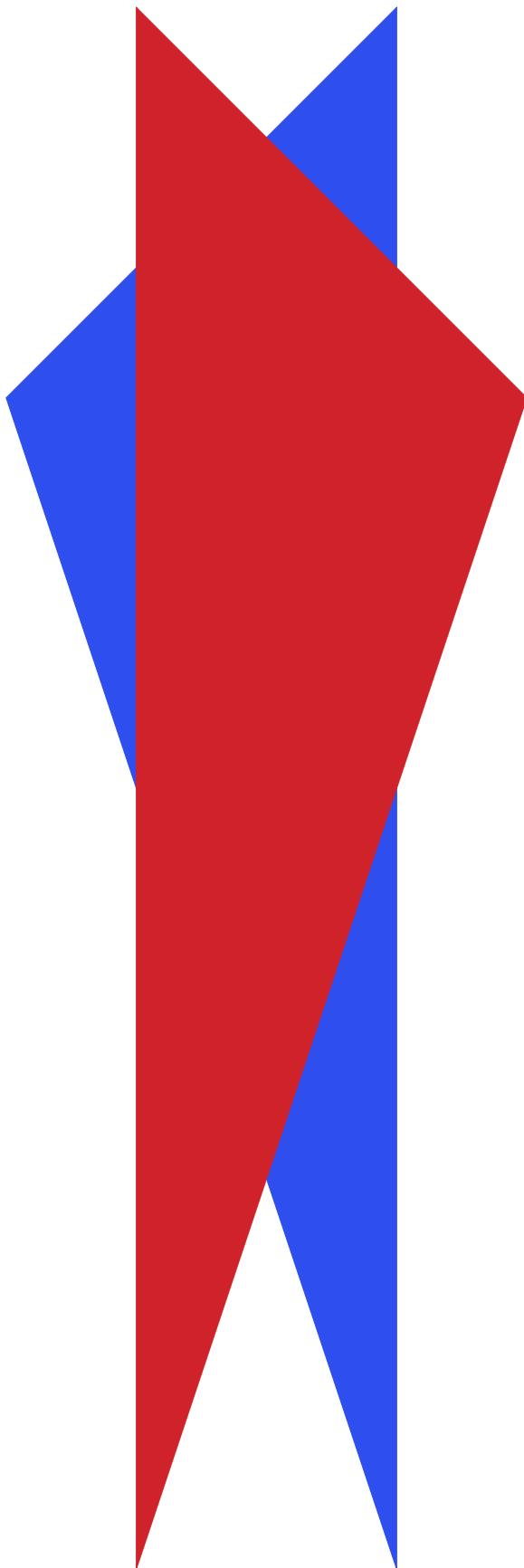
`PolygonToTriangles[l]`
 :: Set polygon to vertex set of triangle(Center of gravity(0,0))

l Polygon

return vertex of triangles

[Example]

```
Graphics[TrianglesToPolygon[PolygonToTriangles[MakePolygon[dataS2, dataT2]]]]
```



2.20 TrianglesToTriangles

`TrianglesToTriangles[offset,cog]`
 :: *cog* run vertexes in a parallel direction

offset vertex set

cog coordinate

return vertex set

[Example]

```
TrianglesToTriangles[{{{ -1, 1}, {0, -2}, {0, 2}, {1, 1}}, {{1, 1}}, {{1, -1}, {0, 1}}]
= {{{0, 0}, {1, -3}, {1, 1}, {2, 0}}, {{1, 2}}}
```

2.21 TrianglesToPolygon

`TrianglesToPolygon[t1]`
 :: Set vertex set of triangle to polygon

t1 list of triangle vertex set

return polygon

[Example]

```
Graphics[TrianglesToPolygon[1, 1, 1, 2, -1, 3, -1, 1, -1, 2, 1, 3]]
```

Out[102]=



2.22 ValNames

`ValNames[n]`
 :: Name variable

n number of variable

return neme of variable

[Example]

ValNames[3]

={{v1x, v1y}, {v2x, v2y}, {v3x, v3y}}

2.23 NormF

NormF[*m*] :: Calculate Frobenius norm

m matrix

return resultant value of Frobenius norm

[Example]

NormF[1,2,3,4]

=30

3 Matrix Functions

3.1 QuadraticFormVariableMatrix

QuadraticFormVariableMatrix[vl]
 :: QuadraticFormVariableMatrix
 vl list of values
 return matrix
 [Example]
 QuadraticFormVariableMatrix[{1,2,3}]
 {{1, 2, 3}, {2, 4, 6}, {3, 6, 9}}

3.2 Div2if

Div2if[n,l]
 :: divide all elements except n-th element of l by 2
 n,l index,list of values
 return vector divided all elements except n-th element of l by 2
 [Example]
 Div2if[3, {3, 6, 7, 5, 1}]
 {3/2, 3, 7, 5/2, 1/2}

3.3 Div2Matrix

Div2Matrix[m]
 :: divide all elements except diagonal ones of m by 2
 m matrix
 return matrix divided all elements except diagonal ones of m by 2
 [Example]
 Div2Matrix[{{1, 3}, {7, 4}}]
 {{1, 3/2}, {7/2, 4}}

3.4 QuadraticFormMatrix

QuadraticFormMatrix[poly, vl]
 :: QuadraticFormMatrix
 poly, vl polynomial,list of values
 return QuadraticFormMatrix
 [Example]
 QuadraticFormMatrix[(a*x^2 + b*x + c-y)^2, {a, b, c}]
 {{x^4, x^3, x^2}, {x^3, x^2, x}, {x^2, x, 1}}

3.5 LinearFormVector

```
LinearFormVector[poly,vl]
    :: LinearFormVector

poly,vl    polynominal,list of values

return     LinearFormVector

[Example]
LinearFormVector[(a*x^2 + b*x + c - y)^2, {a, b, c}]
{-2 x^2 y, -2 x y, -2 y}
```

3.6 VtoTriangle

```
VtoTriangle[V,t]
    :: return coordinates of triangle

V,t        list of vertices,list of triangulation

return     list of coordinates of vertices of triangle

[Example]
VtoTriangle[{{-1, 1}, {0, -2}, {1, 1}, {0, 2}}, {1, 2, 4}]
{{-1, 1}, {0, -2}, {0, 2}}
```

3.7 VtoTriangles

```
VtoTriangles[V,T]
    :: return coordinates of triangles

V,T        list of vertices,list of triangulation

return     list of coordinates of vertices of triangles

[Example]
VtoTriangles[{{-1, 1}, {0, -2}, {1, 1}, {0, 2}}, {{1, 2, 3}, {1, 2, 4}, {2, 3, 4}}]
{{{ -1, 1}, {0, -2}, {1, 1}}, {{-1, 1}, {0, -2}, {0, 2}}, {{0, -2}, {1,1}, {0, 2}}}
```

3.8 Cog

```
Cog[P]     :: return triangle center

P          triangle

return     triangle center of P

[Example]
Cog[{{-1, 1}, {0, -2}, {0, 2}}]
{-(1/3), 1/3}
```

3.9 Trans

```
Trans[P,l]
    :: parallel shift by l
P,l      triangle,vector
return    triangle
[Example]
Trans[{1, 2}, {5, -3}, {-4, 1}], {5, 6}]
{{-4, -4}, {0, -9}, {-9, -5}}
```

3.10 FindMatrix

```
FindMatrix[P1,P2]
    :: find matrix converts P1 to P2
P1,P2    triangles whose center is origin
return    matrix convert P1 to P2
[Example]
P11 = {{0, 2}, {-3, -1}, {3, -1}};
P21 = {{-4, 3}, {1, -2}, {3, -1}};
FindMatrix[P11, P21]
{{1/3, -2}, {1/6, 3/2}}
```

3.11 FindMatrix1

```
FindMatrix1[P1,P2]
    :: find matrix converts P1 to P2
P1,P2    triangles whose center is origin
return    matrix convert P1 to P2
[Example]
P11 = {{0, 2}, {-3, -1}, {3, -1}};
P21 = {{-4, 3}, {1, -2}, {3, -1}};
FindMatrix1[P11, P21]
{{1/3, -2}, {1/6, 3/2}}
```

3.12 FindMatrices

```
FindMatrices[V1,V2,T]
    :: find matrices converts each triangles represented by V1 and T to ones represented by V2 and T
V1,V2,T   V1,V2:list of vertices T:list of triangulation
return    matrices
[Example]
V1 = {{-1, 1}, {0, 2}, {1, -3}, {4, -5}};
```

```

V2 = {{-3, 3}, {-2, 5}, {2, 1}, {3, 1}};
T = {{1, 2, 3}, {1, 2, 4}, {2, 3, 4}};
FindMatrices[V1, V2, T]
{{{3/2, -(1/2)}, {1, 1}},
 {{12/11, -(1/11)}, {10/11, 12/11}},
 {{-(3/13), -(11/13)}, {8/13, 12/13}}}

```

3.13 FindAffineMatrix

```

FindAffineMatrix[P1,P2]
    :: find affine matrix converts P1 to P2

P1,P2    triangles

return    matrix converts P1 to P2

[Example]
FindAffineMatrix[{{-1, 1}, {0, -2}, {0, 2}}, {{-4, 3}, {1, -2}, {3, 0}}]

```

3.14 FindAffineMatrices

```

FindAffineMatrices[V1,V2,T]
    :: find affine matrix convert each triangles

V1,V2,T    V1,V2:list of vertices T:list of triangulation

return    matrices

[Example]
V1 = {{-1, 1}, {0, 2}, {1, -3}, {4, -5}};
V2 = {{-3, 3}, {-2, 5}, {2, 1}, {3, 1}};
T = {{1, 2, 3}, {1, 2, 4}, {2, 3, 4}};
FindAffineMatrices[V1, V2, T]
{{{3/2, -(1/2), -1}, {1, 1, 3}, {0, 0, 1}},
 {{12/11, -(1/11), -(20/11)}, {10/11, 12/11, 31/11}, {0, 0, 1}},
 {{-(3/13), -(11/13), -(4/13)}, {8/13, 12/13, 41/13}, {0, 0, 1}}}

```

3.15 F1a

```

F1a[{{a1x,a1y},{b1x,b1y},{c1x,c1y}},{{m11,m12},{m21,m22}}]
    :: compute quadratic form matrix

{{a1x,a1y},{b1x,b1y},{c1x,c1y}},{{m11,m12},{m21,m22}}
    triangle,matrix

return    quadratic form matrix

[Example]
F1a[{{-1, 0}, {1, 1}, {2, -3}}, {{1, 3}, {4, 2}}]
{{17/81, -(5/27), -(2/81)},
 {-(5/27), 2/9, -(1/27)},
 {-(2/81), -(1/27), 5/81}}

```

3.16 F2a

```
F1a[{a1x,a1y},{b1x,b1y},{c1x,c1y}},{m11,m12},{m21,m22}]
:: compute quadratic form matrix

{{a1x,a1y},{b1x,b1y},{c1x,c1y}},{m11,m12},{m21,m22}}
triangle,matrix

return quadratic form matrix
```

[Example]

```
F2a[{-1, 0}, {1, 1}, {2, -3}], {{1, 3}, {4, 2}}]
{{137/2430, 0, -(7/405), -(1/27), -(19/486), 1/27},
{0, 137/2430, 1/27, -(7/405), -(1/27), -(19/486)},
{-(7/405), 1/27, 4/135, 0, -(1/81), -(1/27)},
{-(1/27), -(7/405), 0, 4/135, 1/27, -(1/81)},
{-(19/486), -(1/27), -(1/81), 1/27, 25/486, 0},
{1/27, -(19/486), -(1/27), -(1/81), 0, 25/486}}
```

3.17 EmbedMatrix

```
EmbedMatrix[n,i,j,M]
:: embed 2-degree matrix M in n-degree 0 matrix

n,i,j,M degree,index,index,matrix

return matrix
```

[Example]

```
EmbedMatrix[6, 2, 4, {{1, 2}, {3, 4}}]
{{0, 0, 0, 0, 0, 0},
{0, 1, 0, 2, 0, 0},
{0, 0, 0, 0, 0, 0},
{0, 3, 0, 4, 0, 0},
{0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0}}
```

3.18 EmbedMatrix

```
EmbedMatrix[n,i,j,k,M]
:: embed 2-degree matrix M in n-degree 0 matrix

n,i,j,k,M degree,index,index,index,matrix

return matrix
```

[Example]

```
EmbedMatrix[8, 2, 4, 7, {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}]
{{0, 0, 0, 0, 0, 0, 0, 0},
{0, 1, 0, 2, 0, 0, 3, 0},
{0, 0, 0, 0, 0, 0, 0, 0},
{0, 4, 0, 5, 0, 0, 6, 0},
{0, 0, 0, 0, 0, 0, 0, 0},
```

```
{0, 0, 0, 0, 0, 0, 0, 0},
{0, 7, 0, 8, 0, 0, 9, 0},
{0, 0, 0, 0, 0, 0, 0, 0}}
```

3.19 EmbedMatrix2

```
EmbedMatrix2[n,i,j,k,M]
  :: embed 6-degree matrix M in 2n-degree 0 matrix
```

```
n,i,j,k,M    n:size, i,j,k:index M:matrix
```

```
return      matrix
```

[Example]

```
A = EmbedMatrix[6, 1, 3, 5, {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}] +
  EmbedMatrix[6, 2, 4, 6, {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}]
```

```
{{1, 0, 2, 0, 3, 0},
 {0, 1, 0, 2, 0, 3},
 {4, 0, 5, 0, 6, 0},
 {0, 4, 0, 5, 0, 6},
 {7, 0, 8, 0, 9, 0},
 {0, 7, 0, 8, 0, 9}}
```

```
EmbedMatrix2[4, 1, 3, 4, A]
```

```
{{1, 0, 0, 0, 2, 0, 3, 0},
 {0, 1, 0, 0, 0, 2, 0, 3},
 {0, 0, 0, 0, 0, 0, 0, 0},
 {0, 0, 0, 0, 0, 0, 0, 0},
 {4, 0, 0, 0, 5, 0, 6, 0},
 {0, 4, 0, 0, 0, 5, 0, 6},
 {7, 0, 0, 0, 8, 0, 9, 0},
 {0, 7, 0, 0, 0, 8, 0, 9}}
```

3.20 F1v

```
F1v[{{a1x,a1y},{b1x,b1y},{c1x,c1y}},{{m11,m12},{m21,m22}}]
  :: compute linear form vector
```

```
{{a1x,a1y},{b1x,b1y},{c1x,c1y}},{{m11,m12},{m21,m22}}
  triangle,matrix
```

```
return      linear form vector
```

[Example]

```
F1v[{{-1, 0}, {1, 1}, {2, -3}}, {{1, 3}, {4, 2}}]
{14/9, 4, -(8/3), -4, 10/9, 0}
```

3.21 EmbedVector

`EmbedVector[n,i,j,k,V]`

:: embed vector V in 2n-degree 0 vector

`n,i,j,k,V` `n`:size `i,j,k`:index `V`:vector

return vector

[Example]

`EmbedVector[5, 1, 3, 4, {-6, 7, 28, 19, -4, 31}]`
`{-6, 7, 0, 0, 28, 19, -4, 31, 0, 0}`

4 Local Interpolations

4.1 RotateAngle

`RotateAngle[m_, flag_]`
 ::controlled rotate angle of m by flag
m matrix
flag 1 or 0 or -1, control the rotate angle
return angle

4.2 NewFindMatrices

`NewFindMatrices[conf_]`
 ::find matrices converted each triangles
conf configuration
return matrices
 [Example]
`NewFindMatrices[Configuration2]=`
`{{{-(5/4), -(3/4)}, {-1, -1}}, {{{-(7/4), -(3/4)}, {2, -1}}}}`

4.3 LocalLinear

`LocalLinear[m_]`
 ::a local linear interpolations depend on time
m matrices computed by `NewFindMatrices[conf_]`
return matrices of local linear interpolation
 [Example]
`LocalLinear[NewFindMatrices[Configuration2]][0.5]=`
`{{{-0.125, 0.125}, {-0.5, -0.5}}, {{-0.875, -0.375}, {1.5, 0.}}}`

4.4 LocalPolar

`LocalPolar[m_]`
 ::a local polar interpolations depend on time
m matrices computed by `NewFindMatrices[conf_]`
return matrices of local polar interpolation

4.5 LocalAlexa

`LocalAlexa[m_]`
 ::a local ARAP interpolations depend on time
m matrices computed by `NewFindMatrices[conf_]`
return matrices of local alexa interpolation

4.6 LocalLogExp

`LocalLogExp[m_]`
 ::a local log-exp interpolations depend on time
m matrices computed by `NewFindMatrices[conf_]`
return matrices of local log-exp interpolation

4.7 LocalInterpolations

`LocalInterpolations[local_, conf_]`
 ::local interpolations that you choice
local `LocalLinear/LocalPolar/LocalAlexa/LocalLogExp`
conf configuration
return

[Example]

```
LocalInterpolations[LocalPolar, Configuration2][t]=
{{{(1 + 0.352786 t) Cos[3.03094 t] + 0.855844 t Sin[3.03094 t],
0.855844 t Cos[3.03094 t] + (1 - 0.0889399 t) Sin[3.03094 t]},
{0.855844 t Cos[3.03094 t] - 1. (1 + 0.352786 t) Sin[3.03094 t],
(1 - 0.0889399 t) Cos[3.03094 t] - 0.855844 t Sin[3.03094 t]}},
{{(1 + 1.65165 t) Cos[2.35619 t] + 0.176777 t Sin[2.35619 t],
-0.176777 t Cos[2.35619 t] - 1. (1 + 0.237437 t) Sin[2.35619 t]},
{-0.176777 t Cos[2.35619 t] + (1 + 1.65165 t) Sin[2.35619 t],
(1 + 0.237437 t) Cos[2.35619 t] - 0.176777 t Sin[2.35619 t]}}}
```


5 Constraint Functions

5.1 ConstMatrix

`ConstMatrix[m_,st_]`
 ::a matrix part of a constraint function of a specific vertex you choice

m vertex

st list of start coordinates

return matrix

5.2 ConstMatrixM

`ConstMatrixM[st_,en_,tri_,t_]`
 ::a matrix part of a constraint function of barycenter

st list of start coordinates

en list of end coordinates

tri list of triangulation

t parameter of time(0...1)

return matrix

[Example]

```
ConstMatrixM[dataS2,dataE2,dataT2,0.5]=
{{1/16, 0, 1/16, 0, 1/16, 0, 1/16, 0},
{0, 1/16, 0, 1/16, 0, 1/16, 0, 1/16},
{1/16, 0, 1/16, 0, 1/16, 0, 1/16, 0},
{0, 1/16, 0, 1/16, 0, 1/16, 0, 1/16},
{1/16, 0, 1/16, 0, 1/16, 0, 1/16, 0},
{0, 1/16, 0, 1/16, 0, 1/16, 0, 1/16},
{1/16, 0, 1/16, 0, 1/16, 0, 1/16, 0},
{0, 1/16, 0, 1/16, 0, 1/16, 0, 1/16}}
```

5.3 ConstVectorM

`ConstVectorM[st_,en_,tri_,t_]`
 ::a vector part of a constraint function of barycenter

st list of start coordinates

en list of end coordinates

tri list of triangulation

t parameter of time(0...1)

return vector

[Example]

```
ConstVectorM[dataS2,dataE2,dataT2,0.5]=
{-0.0625, -0.171875, -0.0625, -0.171875, -0.0625, -0.171875, -0.0625, -0.171875}
```

5.4 ConstfixMatrix

`ConstfixMatrix[n_,k_,l_,st_]`
 ::a matrix part of a constraint function fixing user - specified vector

n weight of constraint function

k,l Choice two numbers you want to fix.

st list of start coordinates

return matrix

[Example]

```
ConstfixMatrix[2,1,2,dataS2]=
{{2, 0, -2, 0, 0, 0, 0, 0},
{0, 2, 0, -2, 0, 0, 0, 0},
{-2, 0, 2, 0, 0, 0, 0, 0},
{0, -2, 0, 2, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0}}
```

5.5 ConstfixVector

`ConstfixVector[n_,k_,l_,st_]`
 ::a vector part of a constraint function fixing user - specified vector

n weight of constraint function

k,l Choice two numbers you want to fix.

st list of start coordinates

return vector

[Example]

```
ConstfixVector[2,1,2,dataS2]={2, -6, -2, 6, 0, 0, 0, 0}
```

5.6 ConstPair

`ConstPair[m_]`
 ::a pair of matrix and vector of a constraint function of a specific vertex you choice

m choice of vertex

return {matrix,vector}

[Example]

```
ConstPair[1][Configuration,s]=
{{{1, 0, 0, 0, 0, 0, 0, 0},
{0, 1, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0},
```

```

{0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0}},
{-2 (-1 + 3 s), -2 (1 + s), 0, 0, 0, 0, 0, 0}}

```

5.7 ConstPair

ConstPair[*m*_,*n*_]
 ::a pair of matrix and vector of a constraint function of a specific two vertices
 you choice

m,n choice of vertex

return {matrix,vector}

[Example]

```

ConstPair[1,2][Configuration2,s]=
{{{1, 0, 0, 0, 0, 0, 0, 0},
{0, 1, 0, 0, 0, 0, 0, 0},
{0, 0, 1, 0, 0, 0, 0, 0},
{0, 0, 0, 1, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0}},
{-2 (-1 + 3 s), -2 (1 + s), -6 s, -2 (-2 (1 - s) + 4 s), 0, 0, 0, 0}}

```

5.8 DoubleMatrix

DoubleMatrix[*m*_]
 ::Return matrix appearing elements and 0 in turn.

m matrix

return matrix

[Example]

```

DoubleMatrix[{{1, 2}, {3, 4}}]=
{{1, 0, 2, 0}, {0, 1, 0, 2}, {3, 0, 4, 0}, {0, 3, 0, 4}}

```

6 Grobal Interpolations

6.1 QuadraticFormAlexa

QuadraticFormAlexa[local_,conf_]
 ::a error function using Alexa's method

local choice of local interpolation

conf configuration

return {matrix,vector}

6.2 QuadraticFormSim

QuadraticFormSim[local_,conf_]
 ::a error function using similarity preserving method

local choice of local interpolation

conf configuration

return {matrix,vector}

6.3 ARAP

ARAP[local_,energy_,const_,conf_]
 ::a global error function in matrix form

local choice of local interpolation

energy choice of energy function

const choice of constraint function

conf configuration

return {matrix,vector}

[Example]

```
ARAP[LocalPolar,QuadraticFormAlexa,ConstPair[1],Configuration2][1]=
{{2, 0, -(1/4), 0, 0, 0, -(3/4), 0},
{0, 2, 0, -(1/4), 0, 0, 0, -(3/4)},
{-(1/4), 0, 1/4, 0, -(1/4), 0, 1/4, 0},
{0, -(1/4), 0, 1/4, 0, -(1/4), 0, 1/4},
{0, 0, -(1/4), 0, 1, 0, -(3/4), 0},
{0, 0, 0, -(1/4), 0, 1, 0, -(3/4)},
{-(3/4), 0, 1/4, 0, -(3/4), 0, 5/4, 0},
{0, -(3/4), 0, 1/4, 0, -(3/4), 0, 5/4}},
{-6.5, -6., -1., 0.5, 3.5, -4., 8.88178*10^-16, 5.5}}
```

7 Draw Animation

7.1 ShowStatus

`ShowStatus[st,en,tri,plotrange]`
 :: Draw start and end of Graphic

st start state of vertex set

en end state of vertex set

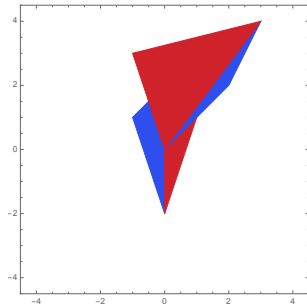
tri constitution of triangle

plotrange Display range

return graph of start and end status

[Example]

`ShowStatus[dataS2, dataE2, dataT2, 4.5]`



7.2 DrawAnimation

`DrawAnimation[local,energy,const,conf]`
 :: aiueo

local Choice of local interpolation

energy Choice of energy interpolation

const Choice of constraint interpolation

conf {start vertex set,end vertex set,constitution of triangle}

return animation

[Example]

`DrawAnimation[LocalAlexa, FrobeniusEnergy, Const2, Configuration2]`

7.3 ListAnimation

ListAnimation[*k*,*local*,*energy*,*const*,*conf*]
 :: aiueo

k number of frame division

local Choice of local interpolation

energy Choice of energy interpolation

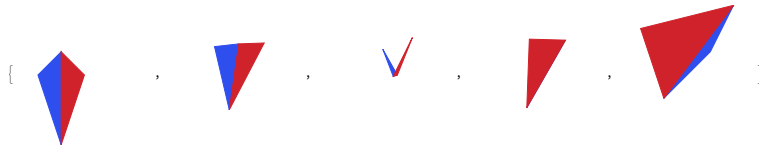
const Choice of constraint interpolation

conf {start vertex set,end vertex set,constitution of triangle}

return list of graph

[Example]

ListAnimation[*k*,*n*_,*c*_,*e*_,*st*_,*en*_,*tri*_,plotrange]



Index

A

| | |
|-----------------------|----|
| AnimationRange | 3 |
| AnimationRange2 | 4 |
| ARAP | 24 |

C

| | |
|----------------------|--------|
| Cartesian | 4 |
| Cog | 13 |
| CogTrans | 8 |
| ConstfixMatrix | 22 |
| ConstfixVector | 22 |
| ConstMatrix | 21 |
| ConstMatrixM | 21 |
| ConstPair | 22, 23 |
| ConstVectorM | 21 |

D

| | |
|---------------------|----|
| Div2if | 12 |
| Div2Matrix | 12 |
| DoubleMatrix | 23 |
| DrawAnimation | 25 |

E

| | |
|--------------------|----|
| EmbedMatrix | 16 |
| EmbedMatrix2 | 17 |
| EmbedVector | 18 |

F

| | |
|--------------------------|----|
| F1a | 15 |
| F1v | 17 |
| F2a | 16 |
| FindAffineMatrices | 15 |
| FindAffineMatrix | 15 |
| FindMatrices | 14 |
| FindMatrix | 14 |
| FindMatrix1 | 14 |

H

| | |
|-----------------------|---|
| HorizontalSnake | 2 |
|-----------------------|---|

L

| | |
|---------------------------|----|
| LinearFormVector | 13 |
| LinearInterpolate | 4 |
| LinearInterpolation | 5 |
| ListAnimation | 26 |
| LocalAlexa | 19 |
| LocalInterpolations | 20 |
| LocalLinear | 19 |
| LocalLogExp | 20 |
| LocalPolar | 19 |

M

| | |
|-------------------|---|
| MakePolygon | 2 |
|-------------------|---|

N

| | |
|-----------------------|----|
| NewFindMatrices | 19 |
| NormF | 11 |

P

| | |
|-------------------------------|---|
| Polar | 4 |
| PolarDecomposition | 7 |
| PolarDecompositionPlus | 7 |
| PolarInterpolate | 5 |
| PolarInterpolateSnake2D | 6 |
| PolarInterpolation | 6 |
| PolygonToTriangles | 8 |

Q

| | |
|-----------------------------------|----|
| QuadraticFormAlexa | 24 |
| QuadraticFormMatrix | 12 |
| QuadraticFormSim | 24 |
| QuadraticFormVariableMatrix | 12 |

R

| | |
|-------------------|-------|
| RotateAngle | 7, 19 |
|-------------------|-------|

S

| | |
|---------------------|----|
| ShowStatus | 25 |
| SnakeTriangle | 3 |

T

| | |
|----------------------------|----|
| Trans | 14 |
| TrianglesToPolygon | 10 |
| TrianglesToTriangles | 10 |

V

| | |
|---------------------|----|
| ValNames | 10 |
| VerticalSnale | 3 |
| VtoTriangle | 13 |
| VtoTriangles | 13 |