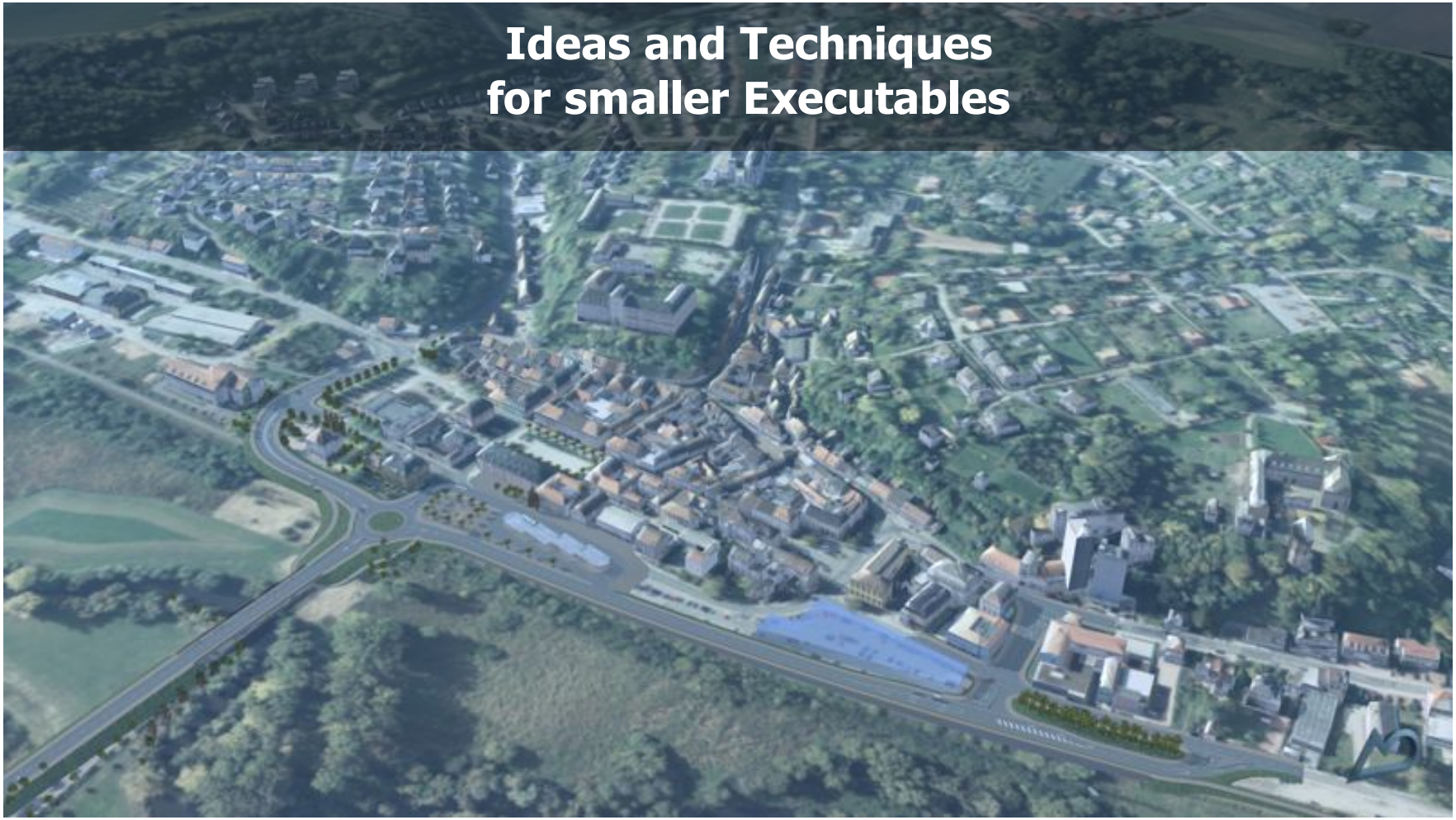


Marcus Winter aka muhmac/freestyle
http://20to4.net/

Agenda

- Compression Basics
- The 20to4 Executable Compressor
- The SuShI Introsystem

Compression Basics

- **Entropy**
  - Describes disorder in a file
  - Calculated based on probability of symbols
  - Smaller entropy means less symbols
  - Compression increases entropy
- **See the following pages**
  - http://datacompression.info/
  - http://www.maximumcompression.com/

Compression Algorithms
Pattern Matching

- **Simple Example**

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Input	D	R	D	O	B	B	S	_	D	R	D	O	B	B	S	_	D	R	_	D	O
Literal	D	R	[]	O	B	[]	S	_	[	-	-	-	-	-	-	-	-	]	[]	[	]
Match			-2 1			-1 1			-8 10										-3 1	-9 2	

Encoding: Prefixes

0 – literal

10 – 1 byte match, 4 bit for offset

11 – longer match, 4 bit offset, 4 bit length

Original: 21*8 = 168 bits

Encoded: 6*9+3*6+2*10 = 92 bits

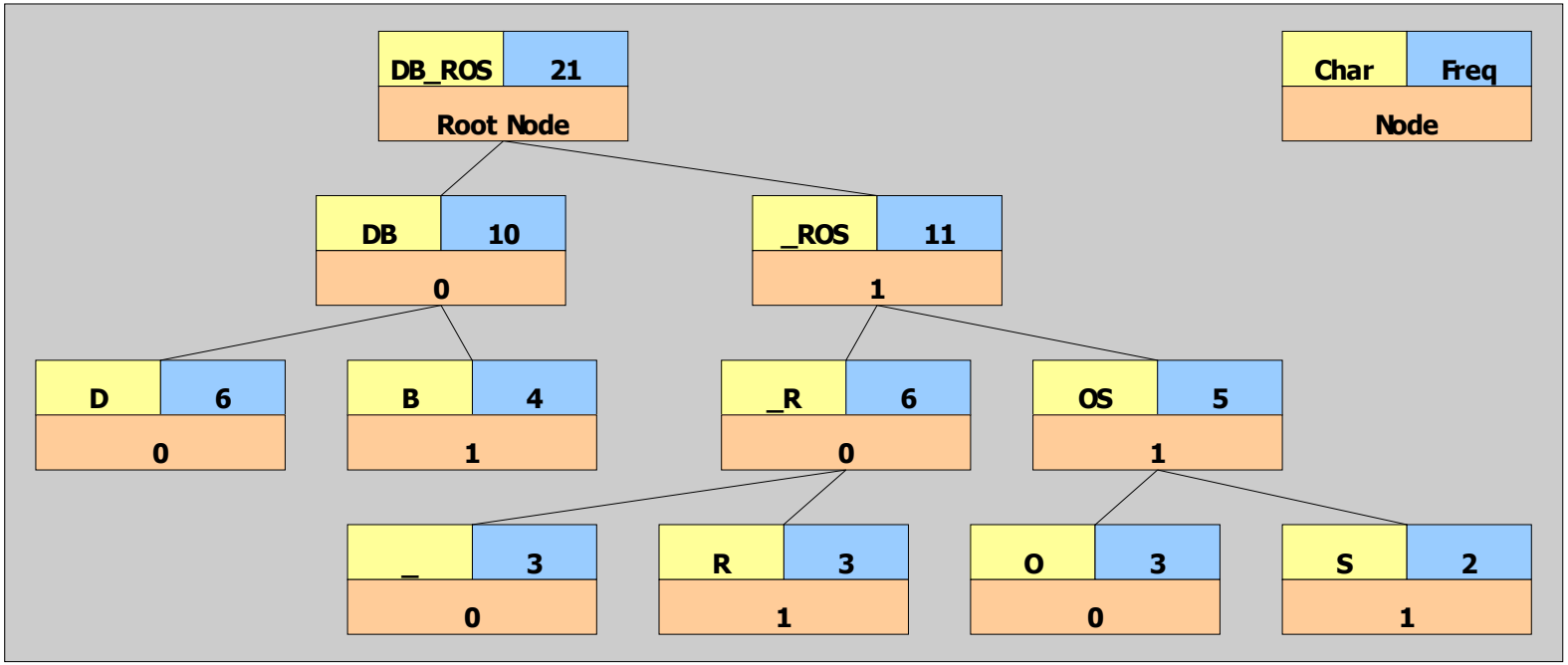
- **Low memory requirements**
- **Decompressor can be kept small**
- **Normally, compression is slower than decompression**

Compression Algorithms
Huffman Encoding

- **Example**

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
D	R	D	O	B	B	S	_	D	R	D	O	B	B	S	_	D	R	_	D	O
Input Code	00	101	00	110	01	01	111	100	00	101	00	110	01	01	111	100	00	101	00	110

- **Tree**



Original: 21*8 = 168 bits

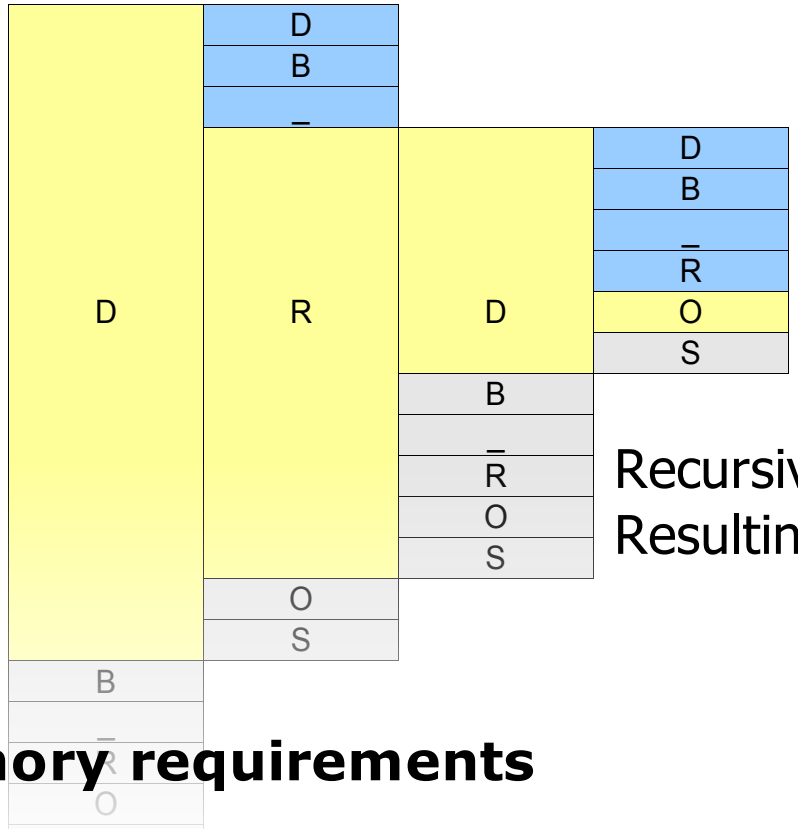
Encoded: 10*2+11*3 = 53 bits

Compression Algorithms
Range Coding

- **Example**

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Input	D	R	D	O	B	B	S	_	D	R	D	O	B	B	S	_	D	R	_	D	O
Code	0	0,18	0,18	0,19																	

Symbol	Probability	Cumulative
D	0,29	0
B	0,19	0,29
_	0,14	0,48
R	0,14	0,62
O	0,14	0,76
S	0,1	0,9



Recursive embedding of probability intervals
Resulting value identifies input stream

- **Low memory requirements**
- **Fast decompression**
- **Limited accuracy is a problem**

Simplicity and Reuse

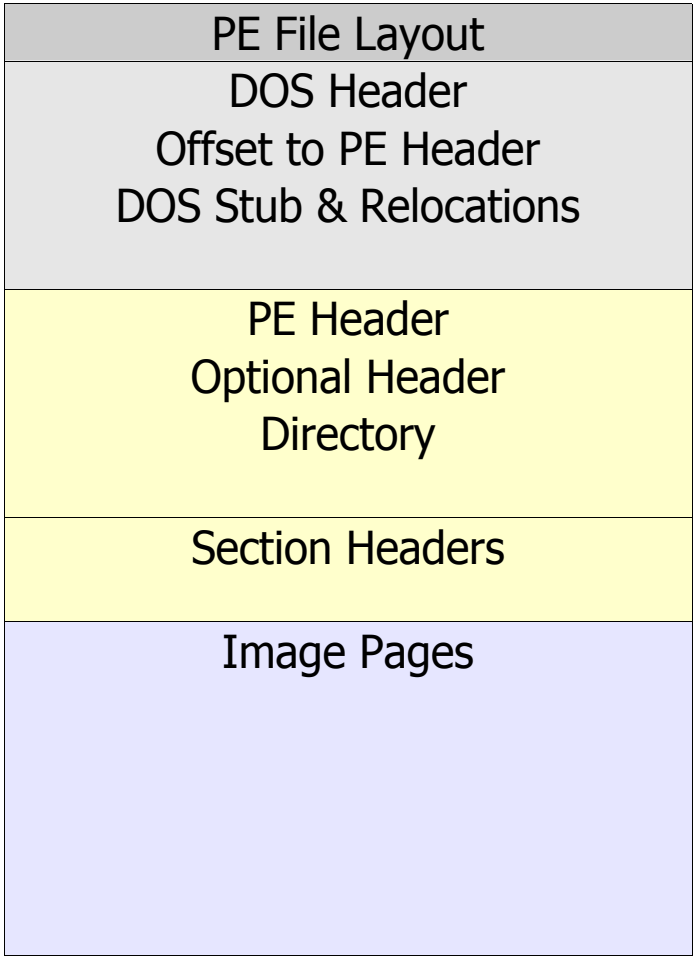
- **Simple code**
  - Code compresses only 2:1 (usually even less)
  - Data can be layed out for optimal compression performance
  - Data driven architectures usually perform better
  - Virtual machine is a good choice
- **Reuse of code is critical**
  - Less code in the executable
  - Good side effect: less errors
  - Lots of the subroutines for sound and graphics are quite similar
  - Example: Interpolators can be shared easily

20to4 Concept

- **Improvement of compression of 4k intros**
  - Decompressor size is critical
  - Target OS is Windows
- **Optimisation of PE File structure**
  - Headers
  - Sections
  - Imports
- **Microsoft CAB compression**

20to4
PE Files

- **Concept**
  - Image of memory block
  - Unused parts are left out
- **Directory**
  - Offsets and sizes of imports and exports
- **Section headers**
  - Relative location and size in memory
  - Location and size in file
  - Protection information
- **Image pages**
  - Contain actual data
  - Must be aligned



20to4
PE File Optimisiation

- **Header cleanup**
  - Most of the values in the headers are not checked
  - Headers can be interleaved
- **Section realignment**
  - Removal of trailing zeroes
  - Sections are initialised to zero up to their virtual size
- **Merging sections**
  - Sections can be combined into a single section
  - Protection suffers, but file usually compresses better
- **Reorganisation of imports**
  - Import by name
  - Import by number
  - Hashed import

20to4
CAB Files

- **Compression**
  - MSZIP – bad performance on small files
  - LZX – better than apack or upx
    - Pattern Matching, Lazy Matching
    - Matches are encoded using Huffman Encoding
- **Decompression**
  - Batch File
    - Output file will be a batch
    - Decompression code resides in the file name field
  - Exe file
    - Functions from cabinet.dll
    - Interleaved headers
    - Decompression code resides between headers and first section

20to4 Decompressor

- **Batch file**

```
set t=%temp%\x.exe
del %t%
extrac32 %0 %t%
%t%
del %t%
exit
```

- **PE exe decompressor**

```
fdintCOPY_FILE
    mov eax,dword [ra(_destfile)]
openfile
    push 384
    push 33057
    push eax
    call [ra(_open)]
    add esp,12
    ret

cabinet_dll
    db 'cabinet.dll',0
crtdll_dll
    db 'crtdll.dll',0

        align 512

entrypoint
;   jmp endofheaders-4096+512

;        align 4

cabinet_thunks
FDICreate
    dd 80000000h|20
FDICopy
    dd 80000000h|22
FDIDestroy
    dd 80000000h|23
    dd 0

crtdll_thunks
_malloc
    dd 80000000h|427 ;va(crtdll_malloc)-2
_free
    dd 80000000h|378 ;va(crtdll_free)-2
```

SuShI Introsystem
Basic Ideas

- **Virtual machine**
- **Supershape object generator**
- **OpenGL texture generator**
- **Modular softsynth**
- **Scripting engine**
- **Optimising script compiler**
- **Tools for ease of scripting**

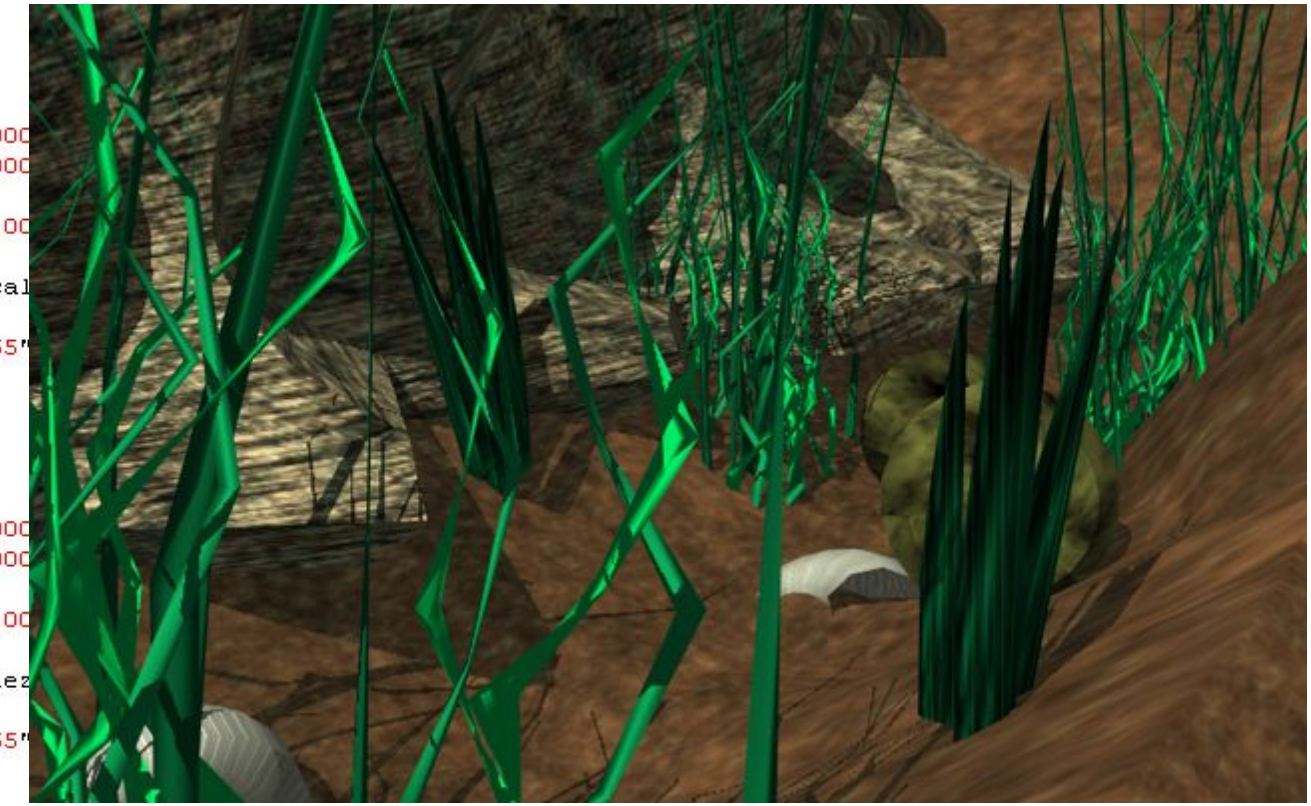
SuShI Introsystem Virtual Machine

- **Organisation**
  - 4096 float/int registers
  - Separate code and data streams
  - Instructions work on register ranges, only base register is specified
  - Some instructions require buffers (eg. reverb effect)
  - Instruction selected from code stream
  - Instruction reads from the data stream and advances the data pointer
- **Instructions**
  - Specification of geometry, materials, lighting and camera setup
  - Generating of geometry and textures
  - Interpolators, Oscillators
  - Data movement

SuShI Introsystem
Scripting Engine

- **XML format**
  - Easy and reliable parsing
  - Flexible and easily extendable
  - Versioning is no problem
- **Scripting**
  - Scripts are generated by tools
  - Scripting by hand should be possible
    (easy reordering and optimising)
  - Interpolation instructions are main means of scripting
  - Oscillators from soundcode can be used too

SuShI Introsystem
Example Script



```
<!-- s_blatt_2 -->
    <supershape name="s_blatt_2"
        baseshape="1"
        r0="1.000000" rphi="-1.000000"
        s1_m="4.000000" s1_a="1.000000" s1_b="1.000
        rfunc="0" rfunca="1.000000" rfuncb="1.00000
        vfunc="1" vfunca="40.000000"
        s2_m="20.000000" s2_a="1.000000" s2_b="1.00
        phires="7" thetares="3"
        scalex="-20.000000" scaley="20.000000" scal
        phimin="-180.005310" phimax="700.020691"
        thetamin="-270.007965" thetamax="90.002655"
<!-- end s_blatt_2 -->
<!-- s_troeten_blatt -->
    <supershape name="s_troeten_blatt"
        baseshape="1"
        r0="1.000000" rphi="0.000000"
        s1_m="4.000000" s1_a="1.000000" s1_b="1.000
        rfunc="2" rfunca="1.000000" rfuncb="7.00000
        vfunc="1" vfunca="40.000000"
        s2_m="20.000000" s2_a="1.000000" s2_b="1.00
        phires="1" thetares="11"
        scalex="-6.000000" scaley="1.000000" scalez
        phimin="120.003540" phimax="220.006485"
        thetamin="-90.002655" thetamax="270.007965"
<!-- end s_troeten_blatt -->
<!-- s_schnecke -->
    <supershape name="s_schnecke"
        baseshape="1"
        r0="-3.000000" rphi="0.000000"
        s1_m="20.000000" s1_a="1.000000" s1_b="1.000000" s1_n1="100.000000" s1_n2="100.000000" s1_n3="100.000000"
        rfunc="1" rfunca="4.000000" rfuncb="1.000000"
        vfunc="1" vfunca="3.000000"
        s2_m="30.000000" s2_a="1.000000" s2_b="1.000000" s2_n1="100.000000" s2_n2="30.000000" s2_n3="100.000000"
        phires="11" thetares="9"
        scalex="-100.000000" scaley="100.000000" scalez="100.000000"
        phimin="-0.000012" phimax="220.006485"
        thetamin="-90.002655" thetamax="270.007965" />
<!-- end s_schnecke -->
<!-- s_saeule -->
    <supershape name="s_saeule"
```

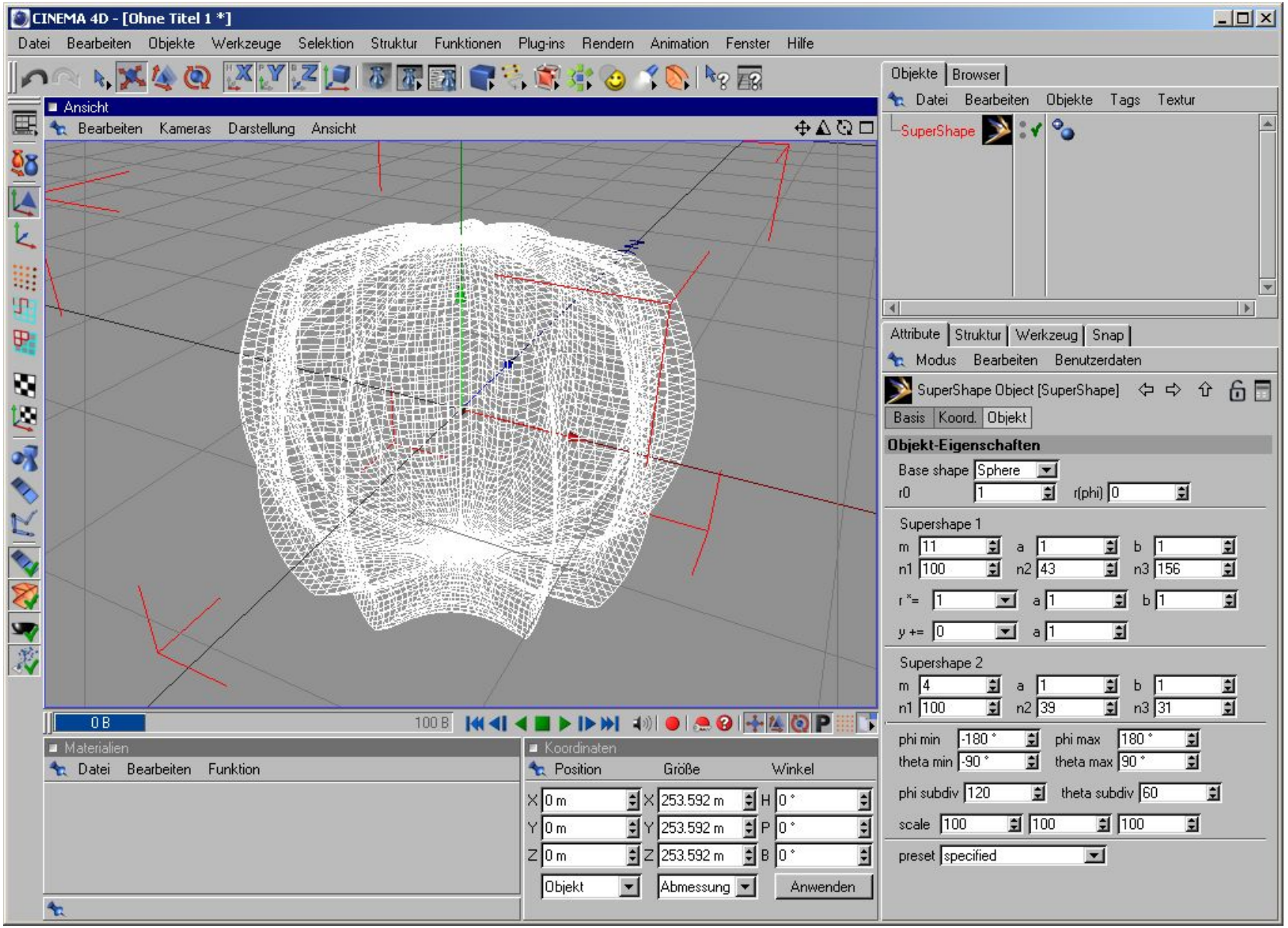
SuShI Introsystem
Script Compiler

- **Features**
  - Pattern analysis
  - Instruction independence check
  - Data/State flow analysis
  - Scriptcode reordering
  - Removal of unnecessary statements
- **Facts**
  - 15000 lines of code
  - Optimisations lead to compression ratios up to 10:1
  - Without them compression ratio was about 4:1

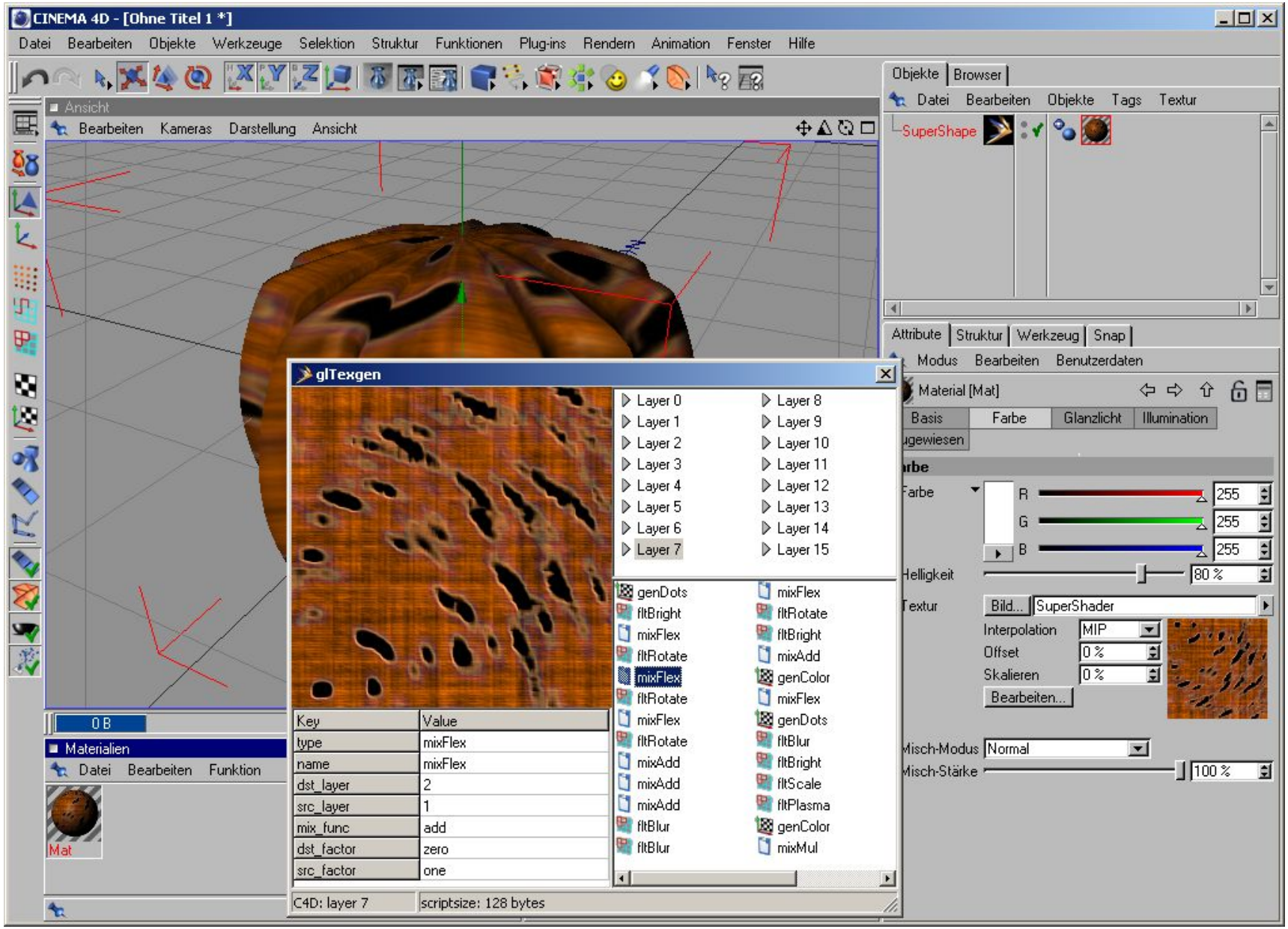
SuShI Introsystem Tools

- **Tools as plugins for standard software**
  - Artists do not need to learn a new interface
  - Additional functionality can be embedded in a „subinterface“
- **Modelling**
  - Cinema 4D object plugins
  - Standard materials
- **Texturing**
  - Custom texture generator
  - Embedded in Cinema 4D as a shader
- **Sound**
  - Custom software synth
  - Any MIDI sequencer can be used
  - Sound can be „programmed“ directly in the synth

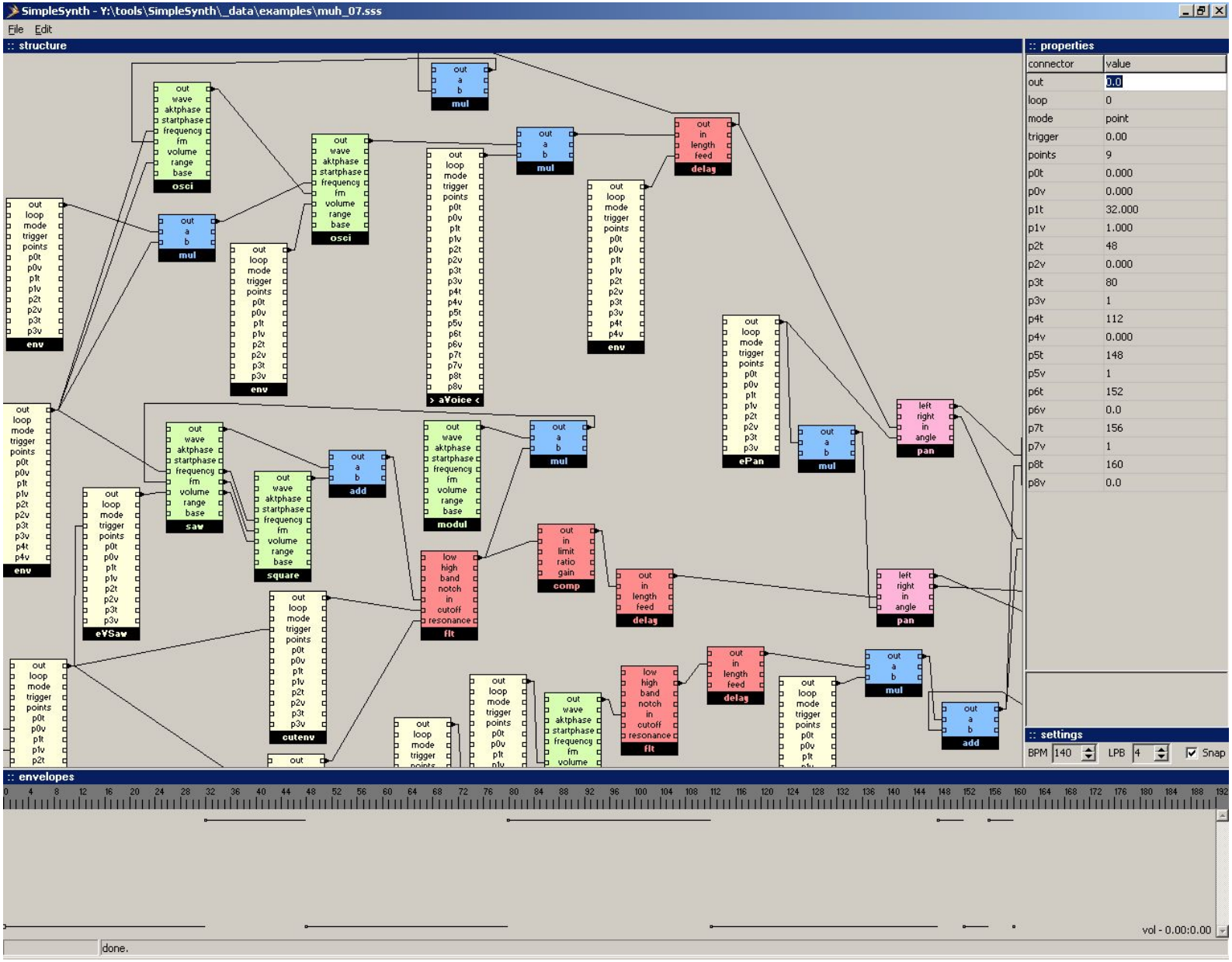
SuShI Introsystem
Object Generator



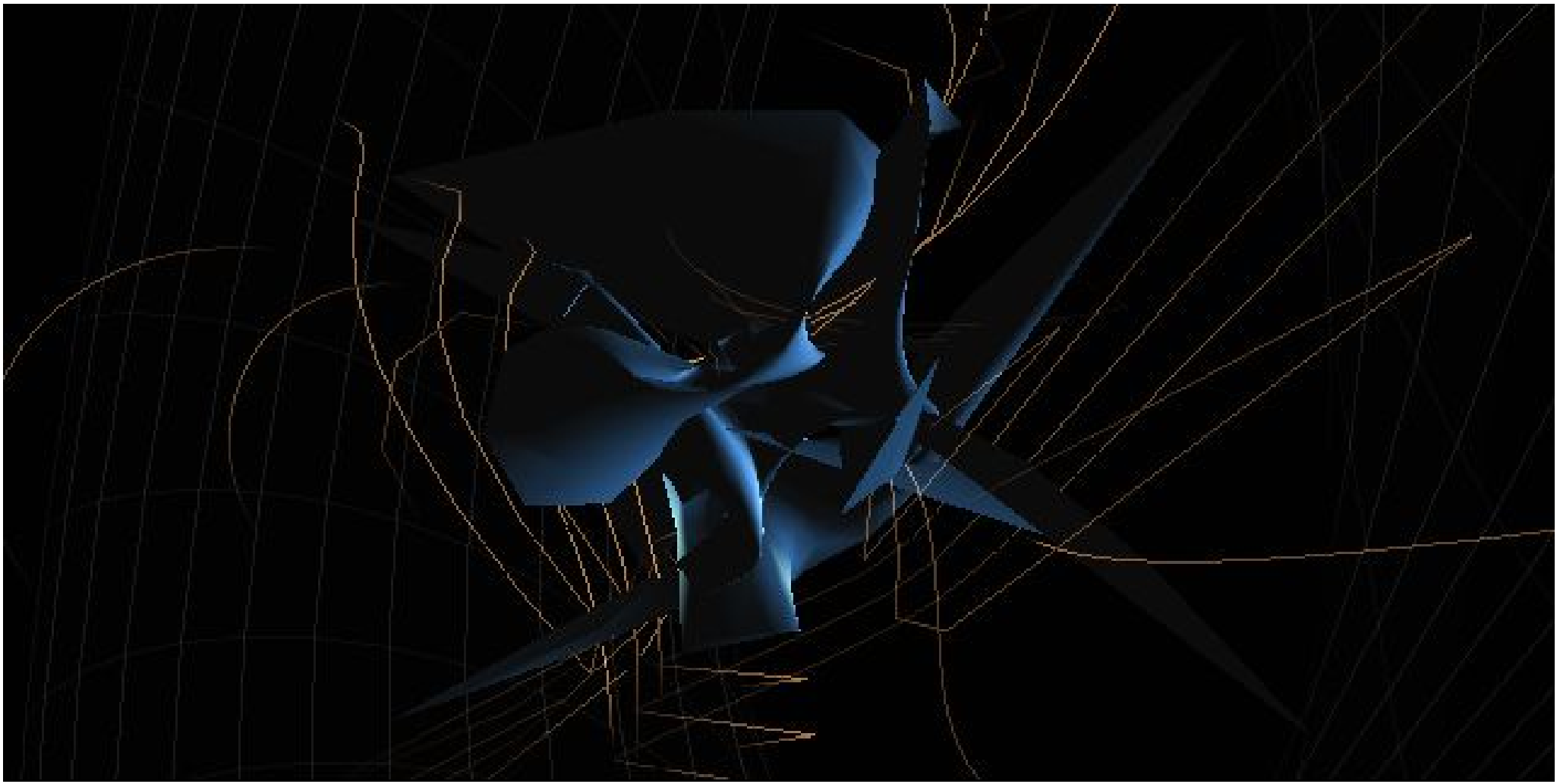
SuShI Introsystem
Texture Generator



SuShI Introsystem
Software Synth



End



- **Have a nice day!**