in

COLLABORATORS			
	TITLE : in		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		July 26, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

in

1.1 Dust - Tutorial

```
                ##########################################################################

#                                                                                        #
#                 Dust V1.62 - Copyright ©1994 by A.Maschke                              #
#                              All rights reserved.                                      #
#----------------------------------------------------------------------------------------#
#                                                                                        #
#                                  Tutorial V0.2                                         #
#                                                                                        #
##########################################################################################


                 1. Introduction

                 2. Tutorial

                 3. Creating the staging-files

                 4. Informations about Dust

                 5. Copyrights

                 6. The Author
                                          (Last changed: 27 February 1995)
```

1.2 1. Introduction

```
Welcome !

This tutorial will help you to become familiar with Dust (I hope so).
It consists of lots of example-animations and this guide describing
how to create such (and better) animations.

The animation-format is ANIM5 with a locked color-map, the frame-size
is 160x128 (16 or 32 colors) - so every Paint-program or animation-
```

```
viewer should play the animations correctly. I use the SHAREWARE-
program "MainActor" to build up them because I prefer different
time-codes at every frame - so try to obtain "MainView" (from Aminet).

Note: The objects I used are very ugly (only spheres, tora, planes,...)
      because I lost my grafix-partition in case of a hard-error
      one week ago. But this tutorial has to show you what types
      of F/X Dust is able to create - not how how to render pretty stills.

Note: Creating animations using Dust and Imagine gives you a very high
      performance. Most of the supplied animations where created in
      20 minutes !
      (That means: creating 60 objects + rendering 60 frames
                   +converting 60 frames + building the animation = 20min!)
      (I own a A1200 with 68030 and 68881 running with 40Mhz.)
```

1.3 2. Tutorial

```
      Introduction:
How Dust operates
                Tutorial: How to create


AnimFunc.anim5

Build.anim5

BuildRND.anim5

Morph.anim5

Morph2.anim5

Morph3.anim5

PExplode.anim5

PFall.anim5

ScaleFaces.anim5

Spheres.anim5

Wave2DLongit.anim5

Wave2DTransv.anim5

Wave3D.anim5
```

1.4 Morph.anim5

```
The MORPH-procedures can morph all types of objects creating two
objects with the same faces-count.

Dust-commands:

 load(1,cube)
 load(2,sphere)
 pmorph(1,2)
 morph(2,1,60,obj)
```

1.5 Morph2.anim5

```
MORPH2 morphs closed surfaces translating points.

Dust-commands:

 load(1,cube)
 load(2,sphere)
 morph2(2,1,60,obj)
```

1.6 Morph3.anim5

```
MORPH3 builds up one object while killing the other.

Dust-commands:

 load(1,cube)
 load(2,sphere)
 pmorph(1,2) (looks better, but isn't necessary!)
 morph3(2,1,60,obj)
```

1.7 PExplode.anim5

```
Create a chess-shaped cube with sharp edges etc.
But try to use as less faces as possible.
Now build a sphere with 12 circle sections and 6 vertical sections
(not more!)

Translate the sphere 60 units along the z-Axis (we want have the cubes
falling down)

After loading these object into Dust create a particle-object and
explode it.

Dust-commands:

 load(1,sphere)
 load(2,cube)
 o2p(1,2,1,f)
```

```
 scalep(1,0.8)
 pexplode(1,60,obj,8.8,-10.0,-0.001,32,3,obj)
```

1.8 PFall.anim5

```
Create a torus having textures etc. rotate it and move it into the
positive z-space (We want have falling down all points.)

Dust-commands:

 load(1,torus)
 pfall2(1,60,obj)
```

1.9 ScaleFaces.anim5

```
Scalefaces scales all object-faces along their center. I used a torus
and scaled the faces with the factor 0.2.

Dust-commands:

 load(1,torus)
 scalefaces(1,60,obj,0.2)
```

1.10 Spheres.anim5

```
This is some difficult.

Create a real sphere in Imagine, apply textures, brushes, set the attributes.
Save the attributes.
Create a sphere from polygons with less (about 50) points, apply the saved
attributes to it, save it as "sphere".

Load into Dust, create a sphere-object using "O2S". Scale down the spheres
with a scaling of about 0.3. Create the 3D-Wave using PWAVE3D, save the
objects in particle-format.

Now write script wich uses Makeloop and Dust to convert the 60 particle-
objects into sphere-objects.

Dust-commands:

 load(1,sphere)
 o2s(1,1,f)
 scalep(1,0.3)
 pwave3d(1,60,obj,particle)


Script "s.bat"

 load(1,obj.%)
```

```
 savespheres(1,obj.%)

Run MakeLoop

 MakeLoop s.bat s2.bat 1 60

Convert

 Dust s2.bat
```

1.11 Wave2DLongit.anim5

```
Create a plane (e.g. 20x20 points), rotate it along the x-axis (90deg).
Drag some Points in z-Direction to get a 3D-Object.

Dust-commands:

 load(1,plane)
 wave2d(1,60,obj,l)
```

1.12 Wave2DTransv.anim5

```
I took the plane from the Dust-distribution (Be sure you take a plane
having its normal in z-direction).

Dust-commands:

 load(1,planedust)
 wave2d(1,60,obj,t)
```

1.13 Wave3D.anim5

```
Create plane 10x10 points and extrude it 10 times. Now center it translating
it 50 units along the y-axis.

Dust-commands:

 load(1,cube)
 wave3d(1,60,obj)
```

1.14 Build.anim5

```
BUILD tries to create objects "from nothing" killing points.

Dust-commands:

 load(1,torus)
 build(1,60,obj)
```

1.15 BuildRND.anim5

```
BUILDRND kills points randomly.

Dust-commands:

 load(1,sphere)
 build(1,60,obj)
```

1.16 AnimFunc.anim5

```
Create a plane having its normal along the z-axis.
Now we "morph" the function "sin((x^2+y^2)/4)" into "sin((x^2+y^2)/2)".
The additional scale-factors you must try or calculate from the
object-size.

Dust-commands:

 load(1,plane)
 animfunc(1,60,obj,1.0,4.0,x0,y0,"20*sin(x0*x0/(10*t0)+y0*y0/(10*t0))")
```

1.17 How Dust works operates

```
1. Dust loads one or two objects in and produces a sequence of transformed
   objects. So you can create F/X you never saw before.

    Example1:
     1. let explode a car
     2. let morph the exploded particles into the exploded particles of a
        tricycle
     3. let implode the tricycle !

    Example2:
     1. transversal particle-wave (20 cubes)
     2. now perform a mathematical distortion on the cubes

    Example3:
     1. build a sphere of many chess-shaped cubes
     2. let explode it (the sphere, not the cubes)
     3. let explode the cubes

    Be sure: No other program comes with this features and has a price of $25 !

      Dust loads and saves the following object-formats:
       -Imagine
         -all attributes
         -face-colors
         -sharp/soft edges
         -Imagine3.x-textures and -brushes

       -Lightwave
         -some attributes
```

```
          -only triangles (Dust only operates on triangles)
          -no textures and no brushes

        -Videoscape
          -color-codes are supported
```

2. After the object-creation you have to build a complete scene containing one of the transformed objects. If you use Imagine or Lightwave then Dust will help you to create the animation(-script) from this scene-file. See chapter 3.

1.18 3. Creating the staging-files

The following programs are supported by Dust1.62:

3.1 Imagine3.x

3.2 Imagine2.0

3.3 Lightwave

1.19 3.1 Imagine3.x

You have to obtain the archive "ISL3.x.lha" from Aminet and to extract the programs "destage" and "restage". Copy them to C:.

Now enter Imagine and create a new project "test".
(My Imagine-directory is located at "hd1:grafix/Imagine3.1", the objects created by Dust are at "tmp:" - so you have to replace this paths with yours.)

Go into the stage-editor, load the first object "tmp:obj.0001", place the camera, add some lights, backgrounds, etc.

Save it.

Enter the action-editor. Change "Highest Frame" into 60.
Now scale the time-lines of all actors except the one of "tmp:obj.0001": click in info-mode at the actor-line and change the "End Frame"-value into 60.

Save it.

Open a shell.
 "cd hd1:grafix/Imagine3.1/test.imp"
 "destage staging ram:st.a"

Enter Dust.
 "staging3(tmp:obj,1,60,1,60,ram:st3)"

```
 Exit Dust.

 Edit the file "ram:st.a".

 Search for a line looking like this:
  "ACTOR FRAMES 1 1 NAME "tmp:obj.0001" STATE "" CYCLE 0. 0. VELOCITY 1. 0. SPLINE ↩
      "
 Kill this line and insert the file "ram:st3" at the current cursor-position.

 Save it. Close the Editor.

 Enter the Shell.
  "restage staging.a ram:st.a"

 That's it-you can render now.

NOTE: It seems to be a big trouble - but it takes me all in all
       about 30 seconds !
```

1.20 3.2 Imagine2.0

```
 You have to obtain the archive "ISL2.0.lha" from Aminet and to extract
 the programs "destage" and "restage". Copy them to C:.

 Now enter Imagine and create a new project "test".
 (My Imagine-directory is located at "hd1:grafix/Imagine2.0", the objects
 created by Dust are at "tmp:" - so you have to replace this paths with yours.)

 Go into the stage-editor, load the first object "tmp:obj.0001", place the
 camera, add some lights, backgrounds, etc.

 Save it.

 Enter the action-editor. Change "Highest Frame" into 60.
 Now scale the time-lines of all actors except the one of "tmp:obj.0001":
 click in info-mode at the actor-line and change the "End Frame"-value
 into 60.

 Save it.

 Open a shell.
  "cd hd1:grafix/Imagine2.0/test.imp"
  "destage staging ram:st.a"


 Enter Dust.
  "staging2(tmp:obj,1,60,1,60,ram:st2)"
 Exit Dust.

 Edit the file "ram:st.a".

 Search for a line looking like this:
  "ACTOR FRAMES 1 1 NAME "tmp:obj.0001" CYCLE 0. 0. TRANSITION 0"
```

```
 Kill this line and insert the file "ram:st2" at the current cursor-position.

 Save it. Close the Editor.

 Enter the Shell.
  "restage staging.a ram:st.a"

 That's it-you can render now.

NOTE: It seems to be a big trouble - but it takes me all in all
      about 30 seconds !
```

1.21 3.3 Lightwave

```
 Suppose the objects created by Dust are at "tmp:".

 Start Lightwave, load the first object "tmp:obj.0001", create a
 perfect scene, add lights...

 Save the scene as "hd1:scene".

 Enter Dust and create 60 scene-files exchanging the objects
  "lwstaging(tmp:obj,1,60,1,60,hd1:scene)"

 An ARexx-Script "hd1:scene.rexx" will be created, too. This script
 will get Lightwave rendering the 60 frames:
  "rx hd1:scene.rexx".

NOTE: If you want to run a special LW-ARexx-command at every frame you
      can specify it via the Dust-LWCMD*-parameters.
```

1.22 5. Copyrights

```
 Imagine     - Copyright ©1993 Impulse Inc.
 VideoScape  - Copyright ©198? Aegis
 LightWave   - Copyright ©1990 NewTek Inc.
 ISL         - Copyright ©1993 Grizzly Bear Labs
 Dust        - Copyright ©1994 A.Maschke
 XPK         - Copyright ©1992 Urban Dominik Mueller, Bryan Ford and many others
 XFH-Handler - Copyright ©1991 Kristian Nielsen.
 IDEA        - Copyright ©1992 Andre Beck (XPK-Implementation)
 RTPatch     - Copyright ©1994 Nico François
 PowerSnap   - Copyright ©1994 Nico François
 Most        - Copyright ©1994 Uwe Röhm
 Pixel3D     - Copyright ©1993 Axiom Software
 ARexx       - Copyright ©1987 William S. Hawes
 MainActor   - Copyright ©1984 Markus Moenig
```

1.23 6. The Author

Feel free to send the registration-fee, suggestions, bug-reports,
NICE animations upto 8 MB, ... to:

Andreas Maschke
Zenkerstraße 5
06108 Halle/Saale
Germany

Telephon: ++49 (0)345/5170331
EMail: epgbc@cluster1.urz.Uni-Halle.DE

1.24 4. Informations about Dust

The version 1.62 or higher is soon available from aminet. Because the
programming costs me a lot of my spare time (I study physics...) Dust
is shareware. But I'm fair (I think). The version you can download
from Aminet has only SOME functions disabled. Most of this tutorial
you can do with the unregistered version !
The price of a registered version is fair: $25 of 25 DM (cash only).

If you want to contact me, please do it via EMail.

Now get an overview of all commands Dust1.6x will come with.

1.25 Commands of Dust1.6x

ABOUT	- show program-information
ADDFACE	- create a face
ANIMFUNC	- modify face-colors algorithmically (animated)
ANIMCFUNC	- modify object-points algorithmically (animated)
ANIMPPOSFUNC	- modify particle-positions algorithmically (animated)
ANIMPROTFUNC	- modify particle-angles algorithmically (animated)
ANIMPSCLFUNC	- modify particle-sizes algorithmically (animated)
AVAIL	- prints available memory
AXALIGN0	- set axis-alignment to 0,0,0
AXPOS	- modify axis-position
AXSIZE	- modify axis-size
BRSAXALIGN0	- set brush-axis-alignment to 0,0,0
BRSAXPOS	- modify brush-axis-position
BRSAXSIZE	- modify brush-axis-size
BRSDIR	- change the path of all brushes of a object
BRSNAME	- modify the brush-name
BUILD	- kill sequentially points (and faces)
BUILDRND	- kill randomly points (and faces)
CALC	- calculate mathematical expressions

```
CD                - change Directory
CENTERAXIS        - center object-axis
CENTERBRSAXIS     - center brush-axis
CENTERTXTAXIS     - center texture-axis
CFUNC             - modify face-colors algorithmically
CLOSEWINDOWS      - close all windows drawing specified object
COLOR             - change object-color
COPY              - copy an object
COPYATTS          - copy object-attributes
COPYBRS           - copy/append the brushes from one object to another
COPYP             - copy particle-objects
COPYPPOS          - copy particle-positions (to combine ANIMP*FUNC)
COPYPROT          - copy particle-angles (to combine ANIMP*FUNC)
COPYPSCL          - copy particle-sizes (to combine ANIMP*FUNC)
COPYTXT           - copy/append the textures from one object to another
CREATEFACES       - create two objects of same face-count (PMORPH step 1)
DISTORT           - distort points (randomly)
DITHER            - change object-dither-amount
EXEC              - execute a batchfile
EXPLODE           - create a realistic explosion
EXPLODEFRAME      - dito, single object
ECHO              - type a message (useful in batchfiles)
FILETYPE          - identify the object-format of a file
FUNC              - modify object-points algorithmically
GET               - show program-parameter(s)
GETOCOUNT         - writes the number of particles in a particle (EXFILE)
GETPSIZE          - writes the size of the shape-object (EXFILE)
GETPPOS           - writes the position of all particles (EXFILE)
GETPROT           - writes the alignment of all particles (EXFILE)
GETPSCL           - writes the size of all particles (EXFILE)
HARDNESS          - change object-hardness
JOIN              - join two objects
JOINP             - join two particle-objects
KILL              - erase an object
KILL0EDGES        - kill unused and illegal edges
KILL0FACES        - kill illegal faces
KILL0POINTS       - kill unused points
KILLBRS           - kill one or all brushes of an object
KILLEDGE          - erase specified or random edge
KILLFACE          - erase specified or random face
KILLP             - kill one or all particle-object(s)
KILLPOINT         - erase specified or random point
KILLTXT           - kill one or all textures of an object
LIMITS            - show actual program-limits
LOADCONFIG        - load a config-file (default s:.dustrc)
LOAD              - load an object (TDDD/LW/VS/Particle-format)
LOADGROUPOBJ      - load an object of a group (TDDD)
LOADSEQ           - load an object-sequence (TDDD/LW/VS/Particle-format)
LWSTAGING         - create multiple Scene-Files from one (for LIGHTWAVE)
MEMORY            - show memory consumption of an object
MEMORYP           - show memory consumption of a particle-object
MERGE             - erase unnecessary points
MORPH             - morph two objects (triangle-morph)
MORPHFRAME        - create specified frame of a morph
MORPH2            - morph two objects (closed surfaces (fast))
MORPH2FRAME       - create specified frame of a morph2
MORPH3            - morph two objects (build-morph (very slow))
```

```
MORPH3RND         - morph two objects (build-morph using random-numbers)
O2P               - convert two objects into a particle-object
O2S               - convert one object into a sphere-object
P2O               - convert a particle-object into an ordinary object
P2OSEQ            - convert a particle-object-sequence into an
                    object-sequence
PEXPLODE          - create a realistic particle-explosion
PFALL             - gravity
PFALL2            - gravity (looks better than PFALL but isn't realistic)
PMORPH            - preprocess objects to morph them
PMORPH2           - slower and better than PMORPH
POSITIVE          - move an object into the positive space
PPOSFUNC          - modify particle-positions algorithmically
PROTFUNC          - modify particle-angles algorithmically
PSCLFUNC          - modify particle-sizes algorithmically
PSTATS            - information about all or one particle-object(s)
PSTATS2           - show information about all buffers (compact)
PWAVE1D           - transversal/longitudinal particle-wave along x-axis
PWAVE1DFRAME      - dito, single object
PWAVE2D           - transversal/longitudinal particle-wave along x- and
                    y-axis
PWAVE2DFRAME      - dito, single object
PWAVE3D           - 3D-particle-wave (along x-,y- and z-axis)
PWAVE3DFRAME      - dito, single object
RANDOMPPOS        - modify particle-positions
RANDOMPROT        - modify particle-rotation
RANDOMPSCL        - modify particle-scale
REFL              - change object-reflectivity
RENAME            - rename object-sequences
REQUEST           - let the user confirm actions (in batchfiles)
REXX              - enter the Dust-ARexx-mode
ROTATE            - rotate object-points
ROTATEAXIS        - Rotate the local axis of an object
ROTATEBRSAXIS     - Rotate the axis of a brush
ROTATETXTAXIS     - Rotate the axis of a texture
ROUGHNESS         - change object-roughness
SAVECONFIG        - save a config-file (default s:.dustrc)
SAVE              - save an object (format: SFORMAT)
SAVELW            - save an object as Lightwave-Object
SAVEP             - save a particle-object (DUST-Format)
SAVEPSEQ          - save a particle-object-sequence as particle-objects
SAVEPOBJ          - save a particle-object as object (format: SFORMAT)
SAVESEQ           - save an object-sequence (format: SFORMAT)
SAVESPHERES       - save a sphere-object as TDDD-Group
SAVETDDD          - save an object as Imagine-Object
SAVEVS            - save an object as VideoScape-object
SCALE             - scale object-points
SCALEFACES        - scale object-faces arround face-centre
SCALEP            - scale the particles of a particle-object
SET               - set program-parameters
SETCLST           - set color of a single face
SETPOINT          - set the position of a single point
SETPPOS           - set position of a single particle
SETPROT           - set alignment of a single particle
SETPSCL           - set scaling of a single particle
SHININESS         - change object-shininess
SHOWTDDD          - shows the hunks of a TDDD-File, good for LOADGROUPOBJ
```

```
SHOWBRS           - show information about one or all brushe(s) of an object
SHOWTXT           - show information about one or all texture(s) of an object
SIZE              - calculate size of specified object
SORTFACES         - sort faces of 2 objects (PMORPH step 2)
SORTFACES2        - dito (slower and better than SORTFACES) (PMORPH2)
SORTPOINTS        - sort points of 2 objects (PMORPH step 3)
SORTPOINTS2       - dito (slower and better than SORTPOINTS) (PMORPH2)
SPEC              - change object-speculum
STAGING2          - create a multi-object-stagefile ("ISL2.0")
STAGING3          - create a multi-object-stagefile ("ISL3.x")
STATS             - show information about loaded objects
STATS2            - show information about all buffers (compact)
TIME              - print execution time of the last command
TRANS             - change object-translucency
TRANSLATE         - translate object-points
TRIANGULATE       - create single triangles, so every face has its
                    own points
TXTAXALIGN0       - set texture-axis-alignment to 0,0,0
TXTAXPOS          - modify texture-axis-position
TXTAXSIZE         - modify texture-axis-size
TXTDIR            - change the path of all textures of a object
TXTNAME           - modify the texture-name
TXTPARAM          - modify one of the 16 texture-parameters
WAVE1D            - transversal/longitudinal wave along x-axis
WAVE1DFRAME       - create a single wave-object
WAVE2D            - transversal/longitudinal wave along x- and y-axis
WAVE2DFRAME       - create a single wave-object
WAVE3D            - 3D-Wave (along x-,y- and z-axis)
WAVE3DFRAME       - create a single wave-object
WINDOW            - open a preview window
WINDOWSEQ         - open well-arranged windows showing an object-sequence
WINDOWCLOSE       - close the specified window
WINDOWDRAWMODE    - change the perspective-flag of the specified window
WINDOWFRONT       - bring the specified window into foreground
WINDOWOUTLINED    - change the outlined-flag of the specified window
WINDOWPERSPECTIVE - change the perspective-flag of the specified window
WINDOWPOS         - move the specified window
WINDOWREDRAW      - redraw the contents of the specified window
WINDOWRESCALE     - rescale the specified window (keepscale=TRUE)
WINDOWROTX        - increase the angle of X-rotation
WINDOWROTZ        - increase the angle of Z-rotation
WINDOWSAVE        - save the image of the specified window as ILBM
WINDOWSIZE        - size the specified window
WINDOWZOOM        - increase the zoom-factor of the specified window
WRITEATTS         - show object-attributes
WRITEAXIS         - show axis-properties
WRITECLST         - show color of each face
WRITEEDGES        - print object-edges
WRITEFACES        - print object-faces
WRITEPOINTS       - print object-points
WRITEPPOS         - print out particle-positions
WRITEPROT         - print out particle-rotation
WRITEPSCL         - print out particle-scale
!                 - execute DOS-commands
```

```
Settings:
   SFORMAT              - global save-format for ALL object-saving commands
   ALIGNP               - align particles along the face-normals of the
                          structure-object
   BACKFACES            - create backfaces writing Videoscape-objects
                          (default: FALSE !)
   OPTEDGES             - optimize edges while loading Lightwave-/VS-objects,
                          makes loading slow
   SAVESPHEREP          - create additional particle-files while saving
                          sphere-objects
   EXFILE               - output-file for the particle-info-commands
                          (programmer-interface)
   EXFORMAT             - output format (float (default) or long (for GCC)
   RANDOM               - global random-seed (you get the same objects at
                          the same seed)
   CHECKMOUSE(boolean)  - check mouse to abort drawing
   WARNINGS(boolean)    - warn before killing objects (default: false)

   HELPDIR              - directory containing all help-files
                          (default: "DustHelp")
   HELPDIR2             - directory containing external help-files, searched
                          before HELPDIR
   PAGER                - pager to view help-files, if set to "" no pager
                          is used
                          (best choice: "most")

 Window-Prefs:

   ASPECT(real)         - window-aspect (default: 0.7 (Euro72 on my monitor))
   KEEPASPECT(boolean)  - keep aspect  after window-sizing (default: false)
   KEEPSCALE            - keep initial scale
   LEFT(integer)        - window-leftedge
   TOP(integer)         - window-topedge
   WIDTH(integer)       - window-width
   OUTLINED             - draw outlined faces
   ROTX(real)           - x-rotation Angle in degrees
   ROTZ(real)           - z-rotation Angle in degrees
   ZOOM(real)           - zoom-factor
   DRAWMODE(drawmode)   - drawmode
   WINDOWSTACK(longint) - stack-size of draw-tasks (default: 12000)
   WINDOWPRI(-3..3)     - priority of draw-tasks (default: 0)

   BREAKWIN(boolean)    - suppress break-requester opened by the most
                          commands
   BWLEFT               - x-coordinate of the command-break-window
   BWTOP                - y-coordinate of the command-break-window

   SCREEN(boolean)      - open the preview-window on a custom-screen
   SCREENWIDTH          - Minimum 160
   SCREENHEIGHT         - Minimum 128
   SCREENDEPTH          - 4..8
   SCREENID(longint)    - decimal ! (e.g. 430116 for Euro72:Multiscan)

   LWCMD1,LWCMD2,LWCMD3 - special Lightwave-commands called at every frame,
                          MakeLoop-format allowed ('$' and '%')
                          Example: "saveimages hd1:pic%" will Lightwave cause
                          to create the Images in the format "pic.0001", ...
```

(this is the default)