Linear Spread
illustrated guide to vvvv for newbies in computer arts
# Table of Contents

2. table of contents  
3. vvvv a multipurpose toolkit  
4. about this guide  
5. vvvv is real time  
6. multipurpose and fast  
7. some extra propaganda  
8. vvvv basics  
9. Renderer (EX9)  
10. primitives.dx9  
11. textures  
12. nodes categories  
13. the power of vvvv is in spreads  
14. vectors  
15. controlling spreads  
16. io boxes  
17. subpatches  
18. S and R nodes  
19. coordinate system  
20. 3d space  
21. 3d models in vvvv  
22. vvvv effects aka shaders  
23. logic  
24. timebased logic  
25. loops and feedback  
26. red nodes and other bugs  
27. boygroup  
28. external protocols  
29. girlpower  
30. user interface  
31. changing values  
32. Inspektor example IOBox  
33. Inspektor example Renderer (EX9)  
34. CodeEditor  
35. how to read vvvv patches  
36. vvvv links

**Introduction**

- Why should I use vvvv?
- Call yourself a ‘media artist’
- Enough to make complex things
- Enough to make beautiful things in 3d space
- Before becoming much more advanced, you may want to spend 20 minutes

**Appendix**

- To print pages 37-43 and keep them near you to you.
introduction
bla-bla-bla
vvvv is a graphical programming environment for easy prototyping and fast development. It is designed to facilitate the handling of large media environments with physical interfaces, real-time motion graphics and audio and video that can interact with many users simultaneously.

The VVVV Group is responsible for its continuous development.

http://vvvv.org/

The vvvv channel on Vimeo is a showcase of work produced by vvvv users. Seeing the power of vvvv will explain it better than a thousand words ever can.

vvvv is for Windows only
Because it is using the power of DirectX, to run vvvv you need either WindowsXP, Vista or 7, the latest DirectX Redistributable and .net3.5 installed.
To run vvvv on Mac, you need to run Windows via BootCamp.

vvvv not suitable for following tasks:
you can not force vvvv to run patches inside a web browser, and you cannot compile vvvv projects into self-running .exe files. You always need a copy of vvvv itself running.
If this is a requirement for you, then you should consider searching for other tools.
This is very first offline and experimental version of a vvvv guide. It made for people who have never used any programming languages before to create motion design, generative art objects or user interaction installations. We have made a very brief overview of vvvv. By reading this guide you will understand the main concepts of vvvv, but not in detail, or advanced usage. To find a more complete explanation and learn vvvv in detail, please follow the links provided on most pages.

Reading this guide should take about 20 minutes. It’s all time you need to understand one of most powerful software packages for creating interactive installations and generative art.

Please note, this is not official guide. The vvvv group is not response for any contents of this guide. For any questions about this guide, please feel free to contact me directly:

skype: andrey.bo.27

Credits and many thanks to:

Proofreading: Elliot Woods  http://www.kimchiandchips.com
             René Westhof
Illustrations: Georg Michri
Design: Nata Suntsova
Thanks to: vdmokstati, ain, alg, Kalle, m4d, vvvv group
          and to my family Nadya and Dominika
why should I use vvvv?
vvvv is real time

Where many other programming languages have distinct modes for developing and runtime, vvvv has only one mode - runtime. In other words, vvvv is constantly performing calculations and rendering graphics while you're editing the patch.

Patching in vvvv is like driving a car. Get results immediately when you want it.
Why am I choosing the wrong club... always?

One stick for all clicks!

You do not need many programmes running

vvvv is suitable for many tasks involving common computer media types like:
video,
audio,
3d animation,
the internet,
interactive multi-users interactions,
multiscreen setups...... to name only a few.

To deal with 3d models, audio analysis and network connections at the same time, you do not need many programmes running, but only the one. And the most important thing, vvvv is still realtime even when you are working with all those things at the same time. vvvv is more like hockey, not golf.
vvv is as cool as Leonardo
and as simple as Robin Hood

vvv is a powerful graphical environment with built-in C# and HLSL code editors and many input/output nodes for all types of external protocols such as TCP, UDP, DMX, MIDI, TUIO, OSC, RS232 and so on...
Thanks to the community, you can also interact with Wii, PSP, Kinect and many more devices.

vvv is free for evaluation and non-commercial use.
You can start using vvvv right now without any restrictions.

http://vvvv.org/downloads

For all commercial applications, you need to buy a license.

http://vvvv.org/licensing
call yourself a ‘media artist’
vvvv basics

vvvv uses graphical objects for programming instead of a textual programming interface. Programs are created within patches. Individual operations and functions are represented as nodes. Connections between nodes are called links, and are drawn and modified with the mouse. Links create a structure that sends data from one node to another.

In most cases a node has more options than you can see. All options are available in the Inspektor. You can open it by selecting a node with a left click, and pressing Ctrl-i.

Input Pin(s) store settled data or receive incoming data via a link.

Node calculates some magic.

vvvv stores most datas in node pins, so links are necessary for transferring data only.

Output Pin(s) sends outgoing data via link(s).

To create your very first patch, run vvvv, double-left click into the empty grey window and select any node from the list.
Welcome to vvvv.

http://vvvv.org/documentation/tutorial-introduction
To view any output we need a renderer node. There are several different renderer nodes in vvvv (with possibilities to visualize different kinds of data). We generally use the Renderer.EX9 node for most types of visual data. The Renderer node has an external window named DirectXRenderer and, of course, it may be FullScreen with any resolution that your graphic card supports.

http://vvvv.org/documentation/dx9-rendering#starting-with-directx

The Renderer node has an external window named "DirectX Renderer"

Click on the window and press Alt-Enter to force FullScreen mode. Before do so, you can check resolution settings in the Inspektor. Please refer to the Renderer’s Inspektor screenshot on page 40

You can either hide the window into a node by pressing Alt-3

or show the renderer window as a block in your patch by pressing Alt-2.

/girlpower/(Halo Ball) patches by roman
primitives (DX9)

The most simple thing that you can draw in the Renderer window is a Quad(DX9) node. But understanding Quad will open many doors in the world of computer media art for you. Don’t let the EX9-DX9 naming disparity puzzle you - they communicate each other perfectly.

So let’s connect Quad (DX9) to the Renderer(EX9)

To introduce basic pins and save your printer ink, we draw results below as you should see them in the Renderer (EX9) window:

Other DX9 nodes work in the same manner. Since pin names speaks for themselves, we do not explain them carefully now.

http://vvvv.org/documentation/quad-%28dx9%29

Feel free to play with following nodes:

- Grid
- GridSegment
- Pillow
- Rope
textures

vvvv supports many more textures formats than most people can imagine would exist. For us, it is a pleasure to know, that by using only Renderer, Quad, Transform and Textures, we can call ourselves contemporary media artists.

http://vvvv.org/documentation/dx9-rendering#texturing

The Renderer node has masses of settings. You should check them in the Inspektor. Same with the Texture nodes.

http://vvvv.org/documentation/howto-prepare-textures
**nodes categories**

vvvv supports many types of data, e.g. values, strings, textures, transforms, meshes etc...
Nodes dealing with same data type are grouped into categories and subcategories.

On starting every new link, vvvv highlights for you pins that can accept that link’s data type

There are many nodes to convert one data type to another. With them you can achieve very interesting and complex interactions.

The NodeBrowser is the central access point to all nodes that exist within vvvv.
Simply doubleclick inside an empty area of a patch to bring up the NodeBrowser.
Rightclick into the text field of the NodeBrowser to switch to a categorized tree view of the node list.

NodeBrowser with node list sorted by text input

Pressing Ctrl-N opens the NodeBrowser in an external window. From it you can drag-n-drop nodes into patches.
enough to make complex things
The power of vvvv is in spreads

The word Spread actually denotes nothing more than a list. A list of arbitrary data. Be it values, strings, colors, ... and a single entry in such a list we call a Slice.

In almost all cases where you want to deal with multiple instances of data, be it values, colors, files, textures, geometries... you can keep your patch pretty simple by thinking in spreads.

**Custom spread**  
I (Node) creates spread by range from # to #

<table>
<thead>
<tr>
<th>0.0000</th>
<th>1.0000</th>
<th>0.0000</th>
<th>1.1200</th>
<th>-0.5100</th>
<th>1.0300</th>
<th>0.8500</th>
<th>-1.4600</th>
<th>7.1700</th>
</tr>
</thead>
</table>

**CircularSpread** outputs circular values instead of you having to understand sin/cos

CircularSpread

<table>
<thead>
<tr>
<th>0.0000</th>
<th>1.0000</th>
<th>2.0000</th>
<th>3.0000</th>
<th>4.0000</th>
</tr>
</thead>
</table>

**Set of random values**

RandomSpread

| 0.4245 | -0.3661 | -0.4878 | 0.3507 | 0.4697 |
Vectors are not only maths tool but they are the most frequently used spreads.

A node looks first at the slicecounts of all spreads it receives. The maximum slicecount of the inputs is used as that node's output slicecount. Well. Nevermind.

Thinking in vectors is very similar to thinking in spreads. vvvv has many vector oriented nodes, like Translate (Transform Vector) etc.

In many cases, using vectors is easier than using separated values.

A spread will never end. When a node gets to the last slice, it goes back to the first one and simply repeats again.
You can select not only one slice, but spread of slices. Either you may sort and shift them by using special nodes from the Spread category, applying custom spreads to Index pins or using BinSize pins.

The BinSize determines how many slices the node will take from the spread.

Stallone is a very flexible node, which allows you to merge slices from any number of inputs and distribute them to any number of slices on any number of outputs.

Every slice has an Index number. Using the Index, you can select any custom slice.

http://vvvv.org/documentation/spreads
IOBox means Input/Output Box, meaning that these nodes are useful for both purposes:
As a means for the user to input data into the running program. On the other hand they can be used to
output/display data from the running program.
We have used many IOBoxes for the screenshots of this guide, to make important data visible.
However, these IOBoxes are generally not vital for the functioning of the patches in the examples.

http://vvvv.org/documentation/tutorial-ioboxes
subpatches

A Subpatch is nothing special at all: it is just a node containing a patch. The term Subpatch describes only the way you view the patch: it is a standard patch which has been placed inside another one.

Creating a Subpatch is extremely easy. Pins can be added to the subpatch node by entering a Descriptive Name to IOBoxes within the subpatch. You can give these names in the Inspektor or by making a middle-mouse-click on a connected IOBox. In this case, the IOBox acts as a parameter, and appears as a pin on the subpatch node.

Now, just save your patch that you want to become a subpatch, close it, and then drag-and-drop the .v4p file from Windows into another patch.

http://vvvv.org/documentation/subpatches

Right clicking on the subpatch node opens it in an external window (or hide it again if it is already opened). Alt-1 Alt-2 Alt-3 shortcuts also work. Ctrl-W - close subpatch window and REMOVE subpatch from the patch.
Sometimes you need to transfer data, but you don’t want to create new links. S and R nodes are for that. R nodes can be placed anywhere in currently open patches. A pull-down menu for R nodes is automatically generated from the all descriptions of SendString pins of all S nodes.

You may want to use them to keep your patches clean when one piece of data goes to many-many places, or the ‘deepness’ of your subputching means that making links is not comfortable at all. Of course, there are also other reasons to use S/R Nodes.
enough to make beautiful things in 3d space
coordinate system

The Renderer window area has a default range of -1 to +1 in both dimensions (XY). The origin has coordinates 0,0 and is in the middle of the window. There is also a z-coordinate, which is not shown in the diagram.

There are naturally no pixels, inches or millimetres in the coordinate system of vvvv, but only four digits values (0.0000). Actually this works well, because coordinates do not depend on the window size or screen resolution. We call this normalised coordinates.

http://vvvv.org/documentation/dx9-rendering#coordinate-systems

Of course, you may want a square to always look like a square, or have a different range of coordinate in some cases. You can do so by applying Transform nodes to transform pins of the Renderer node.

Scale by 0.1 maps the Renderer coordinates range to -10 to +10

TheAspectRatio node helps by making coordinates depend on your window size, you may also force aspect ratio e.g. 4:3 or 16:9 aspects
3d space

The coordinate system in 3d space will be generally be transformed by our view onto the 3D scene, this will create some distortion of the coordinates. The easiest way to look around a 3d space is to connect the Camera (TransformSoftimage) module to the View and Projection pins of a Renderer.EX9 node. Keep in mind that the world you see in 3d space always depends on your point of view.

Easy to use virtual camera for day to day work with the Renderer (EX9).
Keyboard & Mouse Usage:

- **Orbit** press `O` and hold while dragging to Orbit
- **Zoom** press `Z` and hold while dragging to...
  - [left mousebutton:] pan
  - [right mousebutton:] zoom
- **Position** press `P` and hold while dragging to dolly
  - [left mousebutton:] slow
  - [right mousebutton:] fast
- **Reset** press `R` and hold for a moment to reset the viewport

http://vvvv.org/documentation/ex9.spaces

The world you see, always depends on point of view.
In vvvv there are several ways to create a 3d model, aka Mesh:
- use a basic primitives like Grid, Box, Sphere from the EX9.Geometry node category;
- load an .x file from a 3d modelling application using XFile (EX9.Geometry Load);
- load a .dae collada file from a 3d modelling / animation application using the COLLADA plugin;

Each vertex is a container for data like position, orientation (normal), texture coordinates and so on. Since vvvv gives direct access to the vertex data, there are no limits to how you can deform 3d models.
vvvv effects aka shaders

Actually, all we give our worm to eat is the same sphere, but our shader determines how that sphere will appear.

Shaders are nothing but good news.
Firstly, without knowing anything about them, you can use them as usual nodes. Simply look in the EX9.Effects category in the NodeBrowser. The single thing you must always remember is that you must apply a mesh to the shader, to give it an object to visualise.

http://vvvv.org/documentation/tutorial-of-effects-and-shaders

Second, since vvvv has a built-in HLSL textual editor, you may edit and create original effects yourself. Simply right-click on an effect node to open the CodeEditor window, or start your own shader by cloning a template. Select Template (EX9.Effect) from NodeBrowser and press Ctrl-Enter to start editing a new shader.

And third, there are plenty of lovely shaders developed and shared by the community:

http://vvvv.org/contributions
becoming much more advanced, you may want to spend 20 minutes...
logic

You must stop on red light, and you can go on green. 
vvvv has switch nodes which can do this in your patches. 
You can also create complex algorithms with the boolean category nodes.

http://vvvv.org/documentation/logic

Classic if, else, then logic can be easily recreated with boolean nodes like OR, AND, NOT.

When programming sequences, interaction logic or any other more complex tasks, you should try NOT to interweave the logic parts of your patch deep into the graphical parts of your patch. It is generally best to have a 'central command' station where you handle all the logical states in an abstract way, and then send only the relevant data to other places in your patch.
timebased logic

If processes in your patch are going to be concerned with time (like a count down from 10 to 0), or you need timebased loops (like an animated walk or cycled rotation), or you need bangs to be toggled for a while, then there are many nodes for that. Most of them are from the Animation category.

LFO (Animation) counts up from 0 to 1 over a set time period.

Counter (Animation) counts up or down on every bang.

MonoFlop (Animation) keeps a bang for a set amount of time.

LFO (Animation) counts up from 0 to 1 over a set time period.

Counter (Animation) counts up or down on every bang.

MonoFlop (Animation) keeps a bang for a set amount of time.

CurrentTime (Astronomy) knows what day it is now.

Stopwatch (Animation) knows how long its been running.

http://vvvv.org/documentation/timelinersa

TimelinerSA is a very advanced timeline.
loops and feedback

The basic rule to remember is:

vvvv will only evaluate all nodes in the graph only once every frame, so if you want to loop any data, you'll have to wait a bit.

Nodes like FrameDelay (Animation) give an output which doesn't depend on its input this frame, but instead of the input from the last frame - this is perfectly allowed. So as soon as you have at least one node like this in your loop, then you can 'close' the loop. Other nodes also have this property.
red nodes and other bugs

Sometime you can find red nodes without pins. This can happen (for instance) when you download contributions or features from the forum. With red nodes vvvv, tells you that it cannot find a specific file at a specific place. So all you need to do is find the missing file and specify right path to it.

http://vvvv.org/documentation/troubleshooting

vvvv has nodes which help you to find problems. The first is Renderer (TTY) - it will report problems that vvvv encounters to you, in an external text window. It's generally a good idea to place the Renderer (TTY) node somewhere inside your patch, especially whilst you are coding.

The Timing (Debug) node and PerfMeter (Debug) module can report the framerate of your system. MainLoop (VVVV) allows you to limit the framerate that vvvv should run with.

http://vvvv.org/documentation/paths

To fix other bugs, try to search through the forum threads. To highlight any new bug that you've found you can start your own thread in the forum. You may note that vvvv is still a beta release.

Sorry girls, idea for the sketch comes from my girlfriend.
Boygrouping is the name for vvvv’s built-in client-server architecture. It allows you to control any number of render computers (i.e. clients) from 1 or more servers. This way, you can do all of your patching on the server whilst your patch runs across the entire network. VVVV makes sure that all the connected clients will run 'N'Sync'. Applications of boygrouping are usually multi-screen systems and seamless multiprojection set-ups.

Connect two or more PCs to the network via Ethernet and manually specify their IPs.

**Run vvvv on the Server**

with commandline parameter

```
/server
```

**Run vvvv on the Clients**

with commandline parameter

```
/client [SERVER IP]
```

```bash
vvvv.exe /server
vvvv.exe /client 192.168.1.1
```

**Specify clients IPs**

using the Boygroup (VVVV Server) node

Specify clients IPs

Do nothing here

```
192.168.1.2
192.168.1.3
```

**Welcome to the boygroup and still do nothing here**

*Note: same nodes on all clients*

Any data in links between grey and blue nodes are sent over the network. VVVV handles all this for you. You can Boygroup Value, String and Colour types.

http://vvvv.org/documentation/boygrouping-basics
New devices come onto the market around the world every day. For some, you need drivers, for others you don’t. We will just point out some nodes below. I guess you have a time to play with your device.

But lets start with the basics:

The Mouse (System) and Keyboard (System) nodes receive data from the most used devices. In many cases, that’s enough.

Thanks to the TUIO protocol you can receive data from many types of software and devices, e.g. from iPhone.

Human Interface Devices are things like joysticks, wheels, gamepads, touch-screens etc.

Some MIDI keyboards are cheap and effective to use for life performances.

DMX is worldwide standard to control lights (and sometimes sound) via a DMX network.

RS232 is professional standard to control various types of devices, from dvd-players and plasma screens to giant scrapyard claws and factory machines. You can also use this node to talk to your Arduino.

http://vvvv.org/documentation/io
Now all you need is girlpower!

The /girlpower folder comes with every vvvv release and contains many demo-patches of the main vvvv features that were introduced over the past years. Exploring and tinkering with the patches one by one inside the girlpower folder will teach you much than reading ever can.

/gvvv45_beta25/girlpower/
to print pages 37-43 and keep them near you to you
vvvv user interface

opening vvvv for the first time

* Left click on a pin to start a new link
** Double left click on empty space inside the patch opens NodeBrowser
** Double right click on empty space inside the patch opens IOBoxes menu
* Middle click in the patch opens the Main Menu
~ Right click and drag to scrolls around inside the patch window

All the standard Windows shortcuts are the same in vvvv. E.g. ctrl-c, ctrl-v, ctrl-s etc.

working with nodes

* Left click to select a node to change to
** Left click whilst drawing a link to make an anchor

A tooltip appears when you hover your mouse over a node, this shows NodeName (Category Subcategory)

* Left click on a node and drag the mouse to move nodes

* Left click on the right hand side of a node and drag to to resize nodes

** Double left click on any node opens the NodeBrowser

You can even change connected nodes. All pins with the same name will keep their links.

working with links

* Left click on a pin to start a new link

Left click whilst drawing a link to make an anchor

* Left click on a pin to end the link

You can start new links either from output or input pins. Whilst drawing a link, Right click to cancel the link. Right click can also delete links, so be careful.

1.0000
Scale
select link with left click

1.0000
Scale

Ctrl-Y with a selected link changes the link draw type

Changing link draw type is useful to keep your patches visually clean and readable.

http://vvvv.org/documentation/ui-reference

See online UI Reference or download official pdf guide

vvvv user interface

nodes types

There are several types of nodes in vvvv. A small icon left of the node title shows you which type a particular node is.

Subpatches and Modules
These nodes contain patches inside, you can open them and edit them like other patches

Dynamic Plugins
These nodes are C# plugins, you can open and edit them in the CodeEditor

Effects
These nodes are HLSL shaders, you can open and edit them in CodeEditor

Nodes with external windows
These nodes have an external window, you can open it by pressing Alt-1 whilst the node is selected

Native nodes without external windows
You can change the values of their input pins only, but nothing inside the nodes

changing values

You can change values of IOBoxes or nodes pins using the mouse and keyboard.

Move your mouse over an IOBox, hold down the right mouse button and drag up-down or left-right to change the value of the IOBoxes

Double left click on an IOBox to change its value using keyboard input

Editing IOBox (Color) is a bit more complex.

Hold down the right mouse button on an IOBox and move the mouse left-right to change the hue
Hold down the right mouse button on an IOBox and move the mouse up-down to change the lightness
Ctrl + Hold down the right mouse button on an IOBox and move the mouse to change the saturation
Shift + Hold down the right mouse button on an IOBox and move the mouse to change the opacity

Mouse over tooltip shows a pin’s current value
Hold down the right mouse button on a pin and move the mouse up-down or left-right to change the pin values.
Right click on a pin to change its value using keyboard input
Middle click on a connected IOBox to apply a relevant DescriptiveName

You can also change all these values in the Inspektor.
Always keep at least one Inspektor open. **ALWAYS.**

To open the Inspektor, **Press Ctrl-i.** Then whenever you select a node, it will be shown in the Inspektor automatically. To open another Inspektor, whilst you already have one opened, **press Shift-Ctrl-i**

---

Left click on "Attach to Selection", to force the Inspektor to always show data for that selected node.

Right click on values works the same as with IOBoxes.

Settings to create an IOBox which looks and acts like a table.

Settings to make your IOBox act like sliders or button.

Grey squares are for hidden pins

Dark Grey squares are for visible pins

Light Grey squares are for invisible pins

You can change the visibility of pins by clicking on these squares.

A triangle shows that the pin is spreadble

A double triangle shows that the pin is already a spread

Click on an array to show the whole spread in the right column of the Inspektor.

---

http://vvvv.org/documentation/tutorial-inspektor
Always keep at least one Inspektor open. **ALWAYS.**

To open the Inspektor, **Press Ctrl-i.** Then whenever you select a node, it will be shown in the Inspektor automatically. To open another Inspektor, whilst you already have one opened, **press Shift-Ctrl-i.**

The settings pins of the Renderer node are extremely useful. These determine half of the result that you finally see.

Always set a depth buffer whenever working with 3D scenes. Helpful in terms of performance tricks.

You can set any resolution your graphic card supports.

The Renderer node outputs the mouse position within the Renderer window. You can use it instead of the Mouse (System Window) node.

Note that the Renderer's transform inputs are spreadable.

http://vvvv.org/documentation/tutorial-inspektor
vvvv has a built-in textual editor for HLSL effects and C# plugins. With it you can edit contributions or start working on your own plugins or shaders. To start from scratch, find Template nodes within the NodeBrowser, select one and press Ctrl-Enter or Ctrl-left click to clone that template into your patch.

You don’t need to worry about choosing the right CodeEditor for C# or HLSL, vvvv chooses the right one for you.

http://vvvv.org/documentation/plugins

http://vvvv.org/documentation/ex9.effect.file
how to read patches

Since vvvv is very visual in nature, it is much easier to read its patches than textual programming code. To trace where things are going or where things come from, all you need to do is simply follow back along the links. And there are more options and tricks to make exploring more comfortable. Before you start, open an Inspektor window with Ctrl-i. Believe me, you will need it!

Most nodes have supporting help-patches. To see the help patch for a node, select the node and press F1.

A simple trick for faster reading of other peoples patches (e.g. help-patches or /girlpower).

Try to read the patch from down to up, rather than from up to down. As you can remember, most things in the patch finally end up in a Renderer. The things that you see in the Renderer are the result of the whole patch. So first, find the main Renderer (you may note, there are several renderers in complex patches). Usually it looks something like the patch to the right.

Now, just follow the link to the Renderers Layer input pin, back up from the previous output. In most cases connected to the Renderer, you should find a Quad(DX9), some shader node or group of both. Feel free to play with their transforms, color and texture pins.

All abstract mathematics usually goes into Transform nodes. Don’t worry too much about destroying links, since you can easily undo changes by pressing Ctrl-Z.

In case of a subpatch, there may be no Renderer at all. But you can actually do the same, by following from output to inputs.

To find nodes in complex patches use Finder (available since vvvv45_beta25)

Finder has an external window, it opens with Ctrl-F.

Simply click the node you want from the Finder list, and vvvv shows it to you in the patch.
vvvvv links

http://vvvv.org/
nvvv itself. Modern and very comfortable web-site, where you will spend a lot of time.

http://vvvv.org/forums
A place to find answers to most of your questions.

http://vvvv.org/contributions
Useful and lovely stuff, developed by vvvv users.

http://vimeo.com/channels/vvvv
vvvv channel on Vimeo showcases works created by users. The place to see the power of vvvv yourself. Much better than somebody trying to explain “the power of vvvv is...”

http://www.flickr.com/groups/vvvv/pool/
Beautiful screenshots shared by users. Most pictures are completely created with vvvv.

http://meso.net/DIProjects
MESO is a media design company from Frankfurt, Germany and single vvvv distributor. Overview of their various installations, mostly realised with vvvv.