

How to GoDot #11

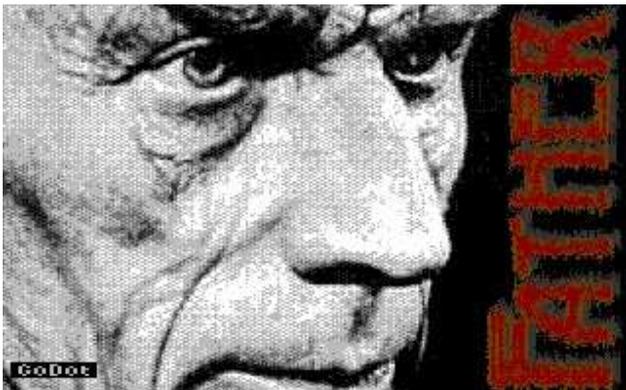
By Arndt Dettke

Moving and Turning

On poster-like images a headline must not necessarily be the topmost object of the artwork. Many times it is at the bottom or even turned left or right and placed at one of the sides of the image. So, what we need for such artwork is two facilities: define objects, and manipulate these to our wishes.

GoDot's objects are called "Clips", and we already have used them many times. They get defined with `mod.ClipWorks`. Their size is bound to how the C64 organizes its graphics memory (it is built from 1000 cards of 8x8 pixels each), delimiting the precision of a clip's width and height to card values. This is the first facility: define objects with `mod.ClipWorks`.

To manipulate an object we have to have the means to isolate it from its surroundings. The majority of GoDot's modifiers don't care much about clips and affect the entire graphics data. So, the first way to isolate a clip is to sever it from the rest of the data by individually saving it to disk or to a special memory area. For this purpose we have `svr.4BitClip` (which we covered in H2G #2 in August 2003).



Pic #1: Samuel Beckett, the „Father“ of GoDot.

Now that we have isolated the clips, we can individually manipulate them. These are the facilities to work on clips: `mod.TileClip` (H2G #8, Feb 2004), `mod.Flip&Mirror` (same issue), `mod.FrameClip` (H2G #1, July 2003), `mod.Stretch-`

`Clip/mod.Squeeze2Clip` (both covered in H2G #3, Sep 2003), `mod.TurnClip`, and `mod.TurnScreen` (both not yet discussed). In addition (and to be complete), we have some more modifiers which affect only parts of the image (but not clips). These are `mod.Tile` (H2G #10, July 2004), `mod.Double`



Pic #2: A Doodle image from the many many pictures on the VCS Super Mega CBM CD (Commodore Central Market).

(same issue), and `mod.Spin` (not yet covered). Also, all 4Bit loaders are capable of loading a clip to a definable location on screen (see H2G #2, Aug 2003, and H2G #9, May 2004) which e.g. is precondition to GoDot's writing capabilities.



Pic #3: A headline written with GoDot (Font: Snap).

This time you will learn about the modifiers you don't know yet - at first, `mod.TurnClip`.

`TurnClip` turns a clip by 90 degrees to the right or to the left. This is ideal for "sidebar" head-



Pic #4: A simple pattern of 1 card size created with `mod.PixelEdit`.

lines like in the Samuel Beckett image entitled "Father" (Pic #1). More details on Pic #1 later in this article. Some additional examples can be seen in the last issue of the DIGEST ("Buck Rogers" and "C64 Fractals"), and on my download site ("Gustav Klimt").

To show off how it works we create a new image now. This is the material we need for it:

“jjboob.bin” (a black&white dithered Doodle image from David Witmer’s Compuserve archives which can be found on Gaelyne Gasson’s Super Mega CBM CD in the Commodore Central Market; see Pic #2), “summer-hl.clp” (this is a 27x5 card clip of a headline written with GoDot using its Snap font; Pic #3), and at last a 1-card pattern of just four horizontal lines like shown in Pic #4.



Pic #5: The result is quite impressive!

This is how to process Pic #5 from these three snippets. First, the headline. It is too wide (27 cards). The clip doesn’t fit down. The height of the C64 screen is just 25 cards, and also a little distance from top and bottom should be added, too. A width of 23 cards would be nice. Next, the writing is too small in height. It would be more harmonic if it was 7 cards high instead of only 5.

Okay, that’s easy. Define a clip around the headline with **mod.ClipWorks** (Wid: 27, Hgt: 5) and perform **mod.StretchClip** (enlarging the clip to screen size). Now change the clip values to Wid: 23 and Hgt: 7 and apply **mod.Squeeze2Clip**. The headline has the size we first wanted. After that, we still have to smooth the letters a little with **mod.PixelEdit**.

To turn the headline we now install **mod.TurnClip** and execute it (confirm the warning at startup). The various settings of this modifier are explained in our sidebox. The headline must be shifted yet to a balanced position. Use **mod.Scroll**

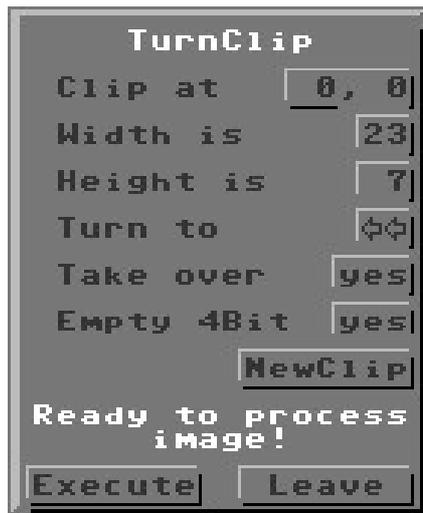
and displace it by 8 pixels right and 8 pixels down. To have it look nicely we finally rim the letters and colorize them using **mod.Histogram** and **mod.Cartoon** like described in H2G#4 in Oct 2003. Use colors blue for the background, red for the letters and light red for the rimming. The result already looks pretty much like in Pic #5. Save it to disk with **svr.4BitGoDot**.

The next step is to convert the 2 color hires Doodle image to a usable multi-color format as fits best for GoDot’s screen display. For this purpose I wrote a brand new modifier (combining two older ones: **mod.AverageColor** and **mod.VsmoothCol**), named **mod.Smooth**. It reduces the resolution from 320x200 pixels to only 160x100, but works wonders on dithered b&w images as can be seen in Pic #5. Save this image to disk like the headline before.

Third step: Adding the background pattern. You can quickly paint it using **mod.PixelEdit**, e.g. to the very first card on screen. Then you define a clip of 1x1 at position 0, 0 and apply **mod.TileClip** (H2G#8, Feb 2004), with setting “Snap to border”. This will fill the whole canvas. Colorize it to orange and yellow with **mod.Histogram**. This backdrop image must be saved, too.

Last step: Combine the three parts of the work to one final image. Reload the second image (the smoothed one), shift it two cards to the right (16 pixels) using **mod.Scroll** and clear the first two card columns to white with **mod.ClipWorks** (option *ClrClip*). Then generate a mask with **mod.QuickMask**. Color white must be transparent. The white portions in the shape of the person have to be cleared away with **mod.DrawMask** (or **mod.MaskEd**). Care for the woman’s hair. Then apply the background to the image: use **ldr.4Bit&Mask** and load the backdrop pattern you saved before. Save the result to disk. Finally load the headline

image, define blue as the transparent color (mod.QuickMask) and generate a mask of it. Then load the image you just saved using mod.4Bit&-Mask, and you're done. You can perhaps change the color of the rimming to black like in Pic #5 with mod.PixelEdit. Frame everything with a thin red borderline using **mod.FrameClip** (H2G#1, July 2003) after clicking *Full* in mod.ClipWorks.



Pic #6: The TurnClip GUI

which I updated with the feature of switching the preview pane on and off while working within the modifier. Thus, you can instantly see how the modifier changed the image! As an example look at Pic #7. In the preview there, you see Pic #8, but the (famous) person in it was turned *UpsideDown*. ;-) Now, to move the "Father" headline from left to the right you just *Mirror* the full image, and after that, *Mirror* only the clip to have it in the same direction as originally. The third way to move a clip is to save it (with **svr.4BitClip**), define a new clip at the wanted location, and then load it to (or into) this clip with **ldr.4BitGoDot** (see H2G#2, Aug 2003). Svr.4BitClip is also able to utilize the REU memory. The filename to use in this case is "temp00". Every 4Bit loader can then use "Temp" as a source to retrieve data.

Mod.TurnClip – This modifier turns a clip either to the left or to the right (by 90 degrees). You select the wanted direction with option "Turn to". Since TurnClip uses GoDot's display area as a

Back to Samuel Beckett (Pic #1). The headline is located on the right. How can we move it there? One way to perform this is to use **mod.-Scroll**. Nicer and more distinct is the use of **mod.Flip&-Mirror** (H2G #8, Feb 2004)

working memory, you are provided with a warning at startup. If you don't accept, the C64's memory remains unharmed. This working area is only 8 KB in size, so the size of a clip to turn is limited to a rectangle of 25x10 cards or a square of 16x16 cards (occupying 8 KB of 4Bit data each). If you exceed this limit, the modifier warns you, and offers the option to change the clip size ("NewClip"). It even tells you which value could best be changed. To still have control you can look at the graphics (with a blinking box for the clip) from within *NewClip*. "Take over" means that you can shift the clip values over to the new position ("yes") or not ("no"). You can also opt for clearing the 4Bit image data during the process or not with "Empty 4Bit". Clips **always** get moved to the topleft corner of the screen when turning them.

Mod.TurnScreen – This modifier is to turn a square area of 25x25 cards right or left. You need an REU for this purpose. The area to turn is selectable with "From Column" and can be controlled with "Show" (which gives you a blinking box in the graphics screen). The result gets written to the topleft corner of the screen.

Mod.Flip&Mirror – Changes to the description in H2G#8, Feb 2004: 1. While working with the modifier, you can switch the graphics preview on and off to better see what the modifier has changed. 2. The shifting of the clip values when affecting full screen can now be denied with "Shift ClipPos: yes/no" which is only active when the *Selected Area* is "Full".

Mod.Smooth – Reduces resolution to 160x100 p.



Pic #7: Previewing from inside the modifier! New in Flip&Mirror!

Command history

Moving clips which have already been defined before.

Option #1:

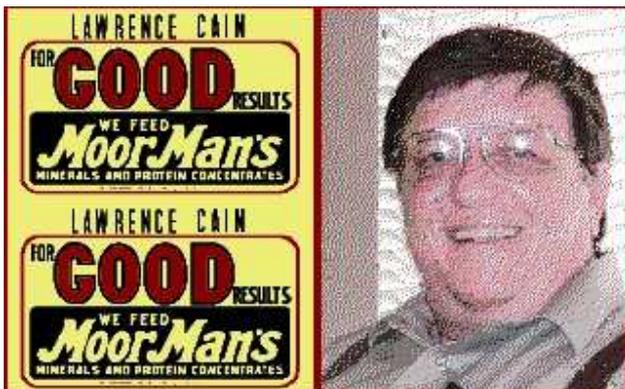
Move the image data to the wanted location (mod.Scroll) and define a new clip there (mod.ClipWorks). This way does not really move a clip but only data.

Option #2:

Inst: Flip&Mirror / Execute
Select Area: Full
Shift ClpPos: yes
Mirror (or *Flip* or *UpsideDown*)
Select Area: Clip
Mirror (or *Flip* or *UpsideDown*)
Leave

Option #3:

Save: 4BitClip
Save "any_name.clp"
Inst: Clipworks / Execute
Clip (Wid, Hgt,) Row, Col
Accept
Load: 4BitGoDot (or any other 4Bit loader)
Load to Clip (or "into Clip" when from Temp)
Leave



Pic #8: A JPG retrieved using GoDot with ldr.JPGViewer.

All images and all new modules used here can be downloaded from my site at <http://www.godot64.de/download>. For more information also visit my news area on that site.

Have fun using GoDot!