

TWIXTOR for OFX Hosts



What is Twixtor?

Twixtor intelligently slows down and speeds up your image sequences.

Twixtor enables you to "time warp" your image sequences with visually stunning results. In order to achieve its unparalleled image quality, **Twixtor** synthesizes unique new frames by warping and interpolating frames of the original sequence... employing RE:Vision's proprietary tracking technology that calculates motion for each individual pixel.

Twixtor Features:

- Calculation of motion vectors at each pixel
- Warping and interpolation of frames to "time warp" input footage
- Stretching of a sequence using a single "speed" scaling parameter
- Keyframeable retiming for complete control on a frame-by-frame basis.
- Ability to mark cut points so that material with edits can be retimed or transcoded without breaking the material into pieces first
- 8 and 16 bits per channel and floating point image support.
- The ability to separate the source material into multiple layers that are inbetweened individually. Layers in **Twixtor** are specified through the use of mattes that you create (independently of **Twixtor**). Separate layers are particularly useful when objects in the scene cross each other. With **Twixtor**, up to 4 separate layers can be specified.
- Up to 12 tracking points can be specified to help guide **Twixtor**'s motion estimation. By using the tracking points you can explicitly tell Twixtor where a pixel moves from one frame to the next in order to guide **Twixtor**'s calculation of motion vectors. You can set the position of each point at each frame by hand, but more importantly, these points can be positioned from frame-to-frame using the host application's point-tracking features, when available. (Please read your host application manual for availability and instructions for setting plugin position settings via point tracking).
- Ability to export and import Motion Vectors sequences

OFX version of Twixtor

Twixtor supports 8 and 16 bits per pixel and floating point processing. The plugin has been tested on Mac, Windows (32 and 64b) and Linux (32 and 64b), and in the Foundry Nuke and Autodesk Toxik. Other hosts probably work but are left to you to properly test for now. Verify our website for hosts / versions currently supported as we regularly add more.

For all applications:

- **Twixtor** works most intuitively on progressive material and in projects, compositions or sequences that are specified to be progressive.
- If you have 3:2 pulldown in your source footage, you will want to **remove 3:2 pulldown** before processing with **Twixtor**. If there is no motion between two frames then there will be no blur. This is useful to know when trying to figure out why **Twixtor** doesn't apply proper processing to frames from a 3:2 pulldown sequence or animation done on "2"s, etc.
- **Twixtor** usually works best on material with fields when the material has been deinterlaced with field blending. For example Toxik provides a deinterlacer operator which allows you to perform that operation prior to applying DE:Noise.

The plugins of **Twixtor** are installed on Windows in C:\Program Files\Common Files\Ofx\Plugins , on Mac in "/Library/OFX/Plugins" and on Linux in "/usr/OFX/Plugins". The path defined by the environment variable OFX_PLUGIN_PATH is supposed to be searched as well by the host application. Consult your host user manual for alternative OFX path location.

Toxik 2008 and over

- Within toxik the plugins of Twixtor show up in a grouping in the tools palette named OFX REVision Effects..

Making a sequence longer than the original sequence: An effect in Toxik cannot render frames outside the main input's time range defined by the first and last frame. One simple way to work around this is as follows: To extend the duration, you must go to the image import control and set the "Repeat" mode to hold, then a new frame will be requested for render anywhere in the timeline. If you do extend the duration to create a longer sequence,, simply apply our free tool ClampTime and set Start and Stop Frame in that tool to the segment you are interested in keeping.

Known issue in Toxik 2008: occasionally while interacting you might end up with a red frame and that red frame will be kept in the cache. This is due to a problem in Toxik 2008 when interrupting (that is changing a slider during a render will cause an

interruption). This should not happen during a real sequence render. If you get a red frame, you will have to interact with a non animated parameter to force a redraw. This is supposed to be fixed since Toxik 2009.

Updating Plugins: For the future when we update a plugin and add and remove parameters it might break your comp/project/database when reopening a comp with the effect in it, either the tool will be said “not present” in which case you need to create a new instance and copy the values OR it might be missing some parameters entry in which case you will need to destroy the .xml file of the tool cached. For example on Win32 the plugin settings cache is located here: C:\Program Files\Autodesk\Autodesk Toxik\resources\toolUi\ofx

Premult-Unmult: Toxik assumes unpremultiplied (straight) rgba inputs, and we provide an unpremultiplied image back to Toxik. There is an Unpremult button in the image import tool as well as a unpremult node tool which should be applied before our tools when images are in a premultiplied format.

Nuke

- Within Nuke the plugins of Twixtor show up in a RE:Vision Effects menu in the main menu bar

Premult-Unmult: Nuke assumes premultiplied rgba inputs, and we provide premultiplied images back. If your input is straight/unpremultiplied, use the premultiplied tool before our tool. If you import Motion Vectors as RGBA (primary) buffer make sure to press the “raw” checkbox in the “Read” node to avoid color space transform (linear to sRGB...).

Others

- We do not test all the possible OFX host s, If you get unexpected behavior from another host other than those listed above, please forward the problem to your host application and to us. Currently we only officially support the listed host applications in the Compatibility section of our website.

Usage of Twixtor

Twixtor comes as set of 4 plug-ins. We describe here the basic Twixtor and then add further explanations for the additional 3 plug-ins.

Source Inputs

Twixtor has 2 inputs. The main input “Source” is mandatory. A second input is provided “Track Source” if and when you decide you need to provide an alternate clip to use for

tracking (particularly useful when the source clip is dark or does not have enough contrast for adequate tracking).

Display

1. Twixtored Output: This setting displays the final **Twixtor** result. This setting has all the layers that you have specified (via mattes) retimed individually and composited back together.
 2. Source: This setting displays the original color source; no retiming is done. This setting is particularly useful when setting **Twixtor**'s Src: Mark Segments setting.
 3. Track Source: If an image sequence is connected to the Alt Track Source image input, then this display setting displays the Alt Track Source image at the current time; no retiming is done. This setting can be useful for determining and inconsistencies with your source and track source.
-

Changing sequence timing.

Okay, now here's the meat of the plugin. How do I retime my footage?

Time Remap Mode: this menu has 2 options for the retiming method:

- **Speed%**

When Speed is chosen as the Time Remap Mode, then you can set (and animate) the speed of the input clip. To obtain the speed of the clip, **Twixtor** gets the value from the Speed% slider. A value of 50 (percent) tells **Twixtor** to play the clip 1/2 as fast as the original, a value of 300 tells **Twixtor** to play the original clip at 3 times its normal rate.

You may ask *"I know that I want to change the duration of the clip by a certain amount, how do I figure out how to set the Speed setting?"*

In general, if you have a constant factor for a duration change, set Speed% to: $100 / (\text{duration change})$

- For 2X duration change, set Speed% = $100/2 = 50\%$
- For a 0.5 duration change (speed up), set Speed% = $100/0.5 = 200\%$

Valid Speed% settings are from -10,000% to +10,000 (that is -100 to +100 times original rate). However, the initial slider display in most host applications will be -1000% to +1000% (-10 to +10 original rate). Note that you can type in a higher value even though the slider shows a smaller range than the valid range.

Note on negative speed: A Speed% of -100.00 does not simply flip the original sequence so that it plays backwards starting at frame 0. Imagine the play head on your clip. If you are at the beginning of the clip and you tell the machine (**Twixtor** in this case) to start playing backward, then there's nothing before the clip starting position to play. That's why you will not see a result with a negative speed as the first keyframe. If you play the clip at 100% for a while, then put in a -30% keyframe (or other negative value) at the place where you want to start playing backward you will see proper results.

- **Frame Num**

If Frame Num is chosen, then **Twixtor** knows you'll be keyframing your time resequencing by exact frame numbers. **Twixtor** looks at the value of the Frame Num slider to figure out what frame from the input sequence to compute, even if it's a fractional value.

This section describes keyframing the remapping of time using frames. Again, let's assume that the input frame rate is the same as the output frame rate.

If you wish to keyframe the timing using direct frame numbers specification, then make sure Time Remap Mode is set to Frame Num. The Speed% setting is ignored in this case, and the Frame Num parameter is used instead. The value of the Frame Num parameter tells **Twixtor** which input frame should be returned for the current time (fractional values are interpolated as described in the previous section).

To use the Frame parameter for retiming:

Immediately after applying **Twixtor** to a clip, it is advisable that you:

First, make the Frame Num parameter animatable.

Second, you are advised to make keyframes for the Frame Num parameter as follows:

- make a keyframe with value 0 at time 0
- make a keyframe at the last time value in the sequence, with a value equal to the last frame number

For example: Let's say you have a sequence of 300 frames.

- make a keyframe with a value of 0 at time 0
- make a keyframe with value 300 at time 300 (time, in this case, is in "frames" instead of timecode).

This will give you an identity mapping (frame 0 returned at time 0, frame 300 returned at time 300, frame 47 will be returned at time 47, etc.).

If Time Remap Mode is set to Speed% then the Frame Num parameter is ignored. Conversely, when the Time Remap Mode is set to Frame Num, the Speed % slider is ignored when key-framing destination frames." TIP: If you are doing a complex dynamic remapping of the video action, you can quickly set your key-frames by setting Motion Vector Quality to None and Frame Interp to Blend. When done setting keyframes, you can go back and set the Motion Vector Quality to the desired quality.

Track Control

Motion Vectors: A menu option with 5 choices for determining the quality of the motion vector calculation.

1. No Motion Vectors: no motion vectors are calculated
2. Sloppy: motion vectors are calculated at 1/4 resolution. Furthermore, a faster but less accurate method is used.
3. Medium: motion vectors are calculated at 1/4 resolution.
4. High: motion vectors are calculated at 1/2 resolution.
5. Best: motion vectors are calculated at full resolution.

Important: **How output frames are calculated.** For demonstration purposes, let's assume that we are slowing down a clip by a factor of 3 (Speed% set to 33.3333). As such, output frame 25 would conceptually need input frame 8.33333. How is frame 8.333 calculated?

- Case 1: Frame Interp: **Nearest** with Motion Vector Quality: **No Motion Vectors**. In this case, no warping can occur (there are no motion vectors!), so the closest input frame in time (input frame 8) is returned.
- Case 2: Frame Interp: **Blend** or **Motion Weighted Blend** with Motion Vector Quality: **No Motion Vectors**. In this case, no warping can occur (there are no motion vectors!). However, we've asked the module to blend frames, so we'll get a return frame that is a dissolve of frame 8 (at 67%) and frame 9 (33%). This is similar to Final Cut Pro with frame blending turned on.
- Case 3: Frame Interp: **Nearest** with Motion Vector Quality set to **anything other than No Motion Vectors**. The motion is calculated from input frame 8 to input frame 9. Frame 8 is then warped using the motion calculated (scaled by 1/3) and returned.
- Case 4: Frame Interp: **Blend** or **Motion Weighted Blend** with Motion Vector Quality set to **anything other than No Motion Vectors**. The motion is calculated from input frame 8 to input frame 9. Frame 8 is then warped using the motion calculated (scaled by 1/3) and returned. Frame 9 is warped using the motion calculated from frame 9 to frame 8 (scaled by 2/3). Finally, these two warped frames are blended as in Case 2.

Note that if fewer frames are being generated for the output sequence, that is, the input sequence is being speed up (total time duration is shrunk) OR a lower output frame rate than input frame rate is specified, then motion blur may then be applied (see Motion Blur option below).

Image Prep: A menu option with 2 choices

1. None
2. Contrast/Edge Enhance

The default image preparation is to do nothing special with the source material that is used for tracking. If your material is dark or has poorly defined edges then **Twixtor** might perform better if you choose Contrast/Edge Enhance. Note that the Contrast/Edge Enhance setting may perform more poorly than using the None setting on material that already has reasonable contrast or has many thin edges close together.

Motion Sensitivity: this setting will be explained later in this document.



Frame Interp

A menu option with 3 choices.

1. Nearest: warps the nearest calculated input frame by the right amount to make the current output frame.
2. Blend: Performs bi-directional warping and then blends the resulting two warped images. The more you stretch a clip, the more this choice is recommended.
3. Motion Weighted Blend: When a new frame is created from existing frames it is often the case that some softening is introduced. This softening can be more or less strong (depending on how far away in time the new frame is located from existing frames in the original sequence. This can result in a pulsing of what appears to be the Twixtored sequence coming in and out of focus. The Motion Weighted Blend option attempts to reduce this pulsing.

Smart Blend

Smart Blend gives better results on a pan or zoom or other footage where objects (or parts of objects) are entering or exiting the frame. When this option is off on such shots, you might get pixels that smear at the edge of the frame. When turned on, this option can produce much better blending when objects are moving off screen. This option will increase the calculation time and is only necessary on certain shots (like shots with pans and zooms with greater than 5%), so that's why it's an option.

	
Frame 1	Frame 2
Slow Mo 5x without Smart Blend. Note streaking on right edge of Twixtor created frames.	Slow Mo 5x with Smart Blend.

Warping

There are 2 Warping settings.

1. Inverse: Inbetweened warped images are calculated using an inverse mapping method. This setting is much faster than the Forward setting. This is the method that the regular version of **Twixtor** uses (and is the default setting for **Twixtor Pro**). It is often better to use this setting when
 - you are **not** guiding **Twixtor**'s motion estimation through the use of Track Points, and
 - portions of objects are **not** moving off the edge of the image (or onto the image from offscreen)
2. Forward: Inbetweened warped images are calculated using a forward mapping method. This method is slower than Inverse, but can produce much less ghosting. The forward method includes the Smart Blend method. It is often better to use this setting when
 - you **are** interactively guiding **Twixtor**'s motion estimation through the use of guiding geometry
 - you are using extreme slow motion (more than 3x slow down) and inbetween frames do not produce desirable results (ghosting or "sliding" of imagery can occur if you use the Inverse method).

Motion Blur Compensation

When speeding up a clip, you are essentially telling **Twixtor** that there is "more action" happening per frame. In order to simulate what happens in a real camera, and to reduce flickering, **Twixtor** applies motion blur when speeding up footage in order to simulate what happens when more motion takes place in the same amount of (virtual) shutter opening of a camera. You can dial in how much motion blur is added with the Motion

Blur Compensation slider. **New in Twixtor 4:** Twixtor also compensates for motion blur by removing motion blur when slowing down footage. Note that the removal of motion blur may be more or less effective depending on how much blur there is in the original sequence.

The default setting for Motion Blur Compensation is 0.0 (that is, by default there is not motion blur compensation). A value of 0.5 is a good place to start if you wish **Twixtor** to automatically compensate for motion blur (this assumes a shutter angle of 180 degrees).

Note that the application of motion blur in **Twixtor** only occurs when footage is sped up or slowed down, to compensate for different virtual shutter speeds. This is different from our ReelSmart Motion Blur product, which adds motion blur based on the motion calculated in a sequence without speeding up or slowing down the footage.

When I retime the footage I'd like Twixtor to respect information I know about the source material. How do I do that?

When you retime the footage you may notice that **Twixtor** inappropriately morphs across a hard cut edit in the source material. Or there may be a section of the footage, such as a dissolve, where you know frame blending without any motion vectors would produce better results. Because of these situations we have provided the following setting.

Src: Mark Segments

This setting allows you to specify information you know about the source material. When animating this option, make sure to do it when Display is set to Source. Application like Nuke who don't support animated menus will see a second control (a popup) to help automatically set integer values. Note as well that the tidbit info that displays when you rollover this parameter lists the values. Be careful that this is not interpolating between key-frames.

Twixtor will not interpolate between two segments marked with different Cut settings. For example, if one frame is marked as part of Cut A and the next frame is marked as part of Cut B, then Twixtor will not interpolate between the two when time remapping. In fact, let's say frame 10 is part of Cut A and frame 11 is part of Cut B. Let's say **Twixtor** is asked to slow the footage down: if **Twixtor** is asked to make frame 10.4, then instead of taking 60% of frame 10 and 40% of frame 11, Twixtor will simply return frame 10. If **Twixtor** is asked to make frame 10.6, then frame 11 is simply returned. In fact, when two adjacent frames are marked as part of different cuts, Twixtor simply returns the nearest frame in the source instead of blending between the two frames. However, when two source frames to be interpolated are marked with the same Cut setting, then the Motion

Vectors and Frame Interp settings will be used to interpolate between the two frames to create the output frame.

Input Start Frame

The Start Frame parameter must be used to get proper result with when using the Speed Mode so the time is stretched from that point.

Input Last Frame

The Start and Last Frame are used to prevent getting invalid frames when you are playing with speed mode and can get out of range, this way you are guaranteed that it will repeat the first and last frame always in such case.

Nuke users: Note an additional **Set Time Range** button shows up. Some Nuke operators might not set this so these values are not guaranteed to be accurate in all cases.

Note as well a utility to Clamp Time is also provided, that tool simply sets to transparent frames outside of the defined range and can be applied after Twixtor.

The tracking is not working so well on my source footage, what can I do?

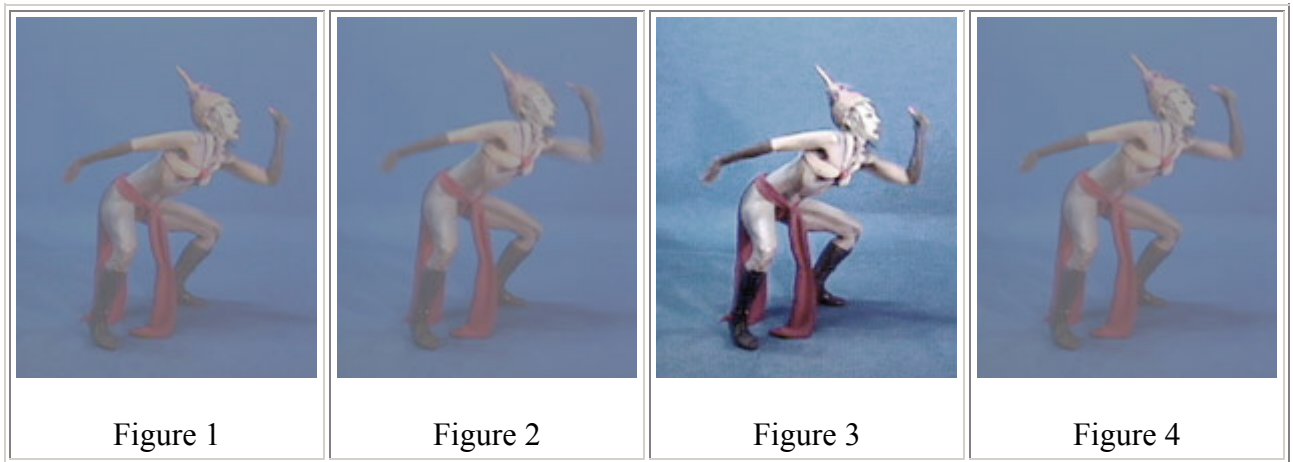
Tracking problems show up as motion not being properly tracked: either objects in the output sequence "gloop" together, or objects don't move enough causing what seems to be simple cross dissolves of the input images. (for what constitutes problems for our tracker, see the "PROBLEMS IN INPUT SEQUENCES" section below.

Alt Tracking Source

You might say, "if only I could enhance the colors used in the tracking process, but then use the colors from the original sequence for the Twixtored sequence." Or, the reverse, you've heavily filtered the original (which you've used as the color source for **Twixtor**), but you think you might get better tracking results by using the original footage as the imagery to be tracked. So, we've provided this parameter to set an Alternate Motion Source: If Alt Tracking Source is set to None then the Color Source is used for both tracking and the color of the **Twixtor** sequence. However, if an Alt Tracking Source clip is specified, then this sequence is used for tracking and the Color Source clip is used for color. Of course, **Twixtor** assumes that both the Alt Motion Source and Color Source (Main Input) clips line up exactly in time

If an alternate motion clip is supplied, then it must have the same pixel dimensions as the color source; otherwise a **semi-transparent red image** will be produced to let you know that there is an error.

So, how can you enhance tracking using an Alt Motion Source? The following set of pictures seeks to explain:



Using an alternate tracking source:

- Figure 1: Original low contrast sequence
- Figure 2: Inbetween frame, note that the rear arm is not tracked well because the arm is similar in intensity to the background (of course our eyes see the difference!)
- Figure 3: Image enhanced, but used only for the tracking portion of **Twixtor** by specifying it as the Alt Motion Source.
- Figure 4: The original colors in Figure 1 are used, but the tracking is performed on a sequence represented by Figure 3. Note that the rear arm is now tracked nicely!

Motion Sensitivity

Okay, you might say "I don't have time to experiment with image processing my footage to specify an alternate motion source," or worse, "I try to color correct my footage and it does not help the tracking!" If objects still "gloop" together when they shouldn't, then you can try manipulating the Motion Sensitivity parameter. **Motion Sensitivity limits how much pixels can move.** A value of zero assures that pixels can't move very much, and a value of 100 allows pixels to move as much as the motion interpolator can calculate. The default value of 70 might sometimes be a bit too "ambitious" for some material... by reducing the sensitivity you might be able to create a more interesting result (giving a more traditional frame blended look) where there is large motion rather than some sort of inappropriate swirl that **Twixtor** might introduce. Conversely, if there is only one object in the scene and it moves in large amounts, the default value of 70 may not be ambitious enough.

Twixtor Pro Features

Concepts and Terminology for Twixtor Pro

Twixtor Pro allows you to separate your footage into multiple layers by supplying mattes.

When the image sequence is not separated into layers (that is, you have not specified any mattes for FG1, FG2 or FG3), then there is only one layer: what we will refer to as the Main layer. When you separate the image sequence into two layers by supplying one foreground matte, then the foreground layer will be referred to as FG1, and the rest of the image will be referred to as the Background layer. Continuing in that manner, if you supply two matte sequences, then the sequence will be separated into 3 layers: FG1 (the topmost layer), FG2 (the intermediate layer) and the Background layer (which is the remainder of the image NOT in FG1 or FG2). Similarly you can separate an image sequence into 4 layers by supplying 3 matte sequences.

As such, the Background layer will be what is NOT in any foreground layers that are specified (via your mattes). If no foreground layers are specified the Background layer is also called the Main layer. A description of how mattes for layer separation should be constructed will be detailed below.

These concepts will become clearer as we discuss how to use the layer options of **Twixtor Pro**.

We often refer to "interactively guiding **Twixtor**." This can be through the Tracking Points settings of **Twixtor**.

Settings of Twixtor Pro

Twixtor supports up to 5 inputs: the main input (also the Color Source), an alternative sequence which is the one used to calculate the motion and 3 matte segmentation layers. All the inputs must be the same size in Twixtor PRO.

Display

By default, this displays the Output render. This menu allows you to display the color source or the tracking source, in which case no time remapping is performed.

Display Layer

This setting determines which layer is to be displayed when separating the footage into multiple layers using mattes. We will explain this setting in more detail in the next chapter. For now, leave Display Layer set to All.

DrawGeom will be discussed in the section that describes using geometry to help guide the motion. Note this setting only shows up when in Source or TrackSource mode for Display as it only makes sense in source time.

The use of Motion Vectors, Image Prep remains just as it does for the regular version of **Twixtor**.

Motion Sensitivity gets renamed in **Twixtor Pro** as Main_BG Sensitivity. This setting works just as described in for regular **Twixtor**. However, in **Twixtor Pro**, this setting is used only for the Background layer if the sequence is split into more than 1 layer through the use of mattes (described below).

When using tracking guidance, you will be telling **Twixtor** exactly where things line up. When separating footage into layers using **Twixtor**'s separation feature you will be telling **Twixtor** where differing objects are. In both these cases, you will be reducing **Twixtor**'s tracking errors and it will probably be desirable to set the Main_BG Sensitivity to 100.

Cache Optical Flow

The PRO version allows you to turn off optical flow caching. Optical Flow caching can produce much faster results when you render a slow-motion in frame order but will provide no extra joy in a distributed render context. Also in memory limited context, optical flow caching will retain many floating buffers (vary per amount of layers) and this is memory some other tools might want to use somewhere else in the graph ☺

Splitting your input sequence into one foreground layer and a background layer.

Let's assume you have a sequence with one foreground object. Let's also assume that you have a matte for this object at each frame. We now describe how to improve **Twixtor**'s tracking by using this matte. In the next section we will describe how to use more than one matte to separate your input sequence into more than 2 layers.

Twixtor Pro allows you to specify a matte (or more) so that you can track a foreground object separate from the background. This can sometimes help immensely with "gloopiness" in the Twixtored sequence. Except for specifying nice anti-aliased edges (and perhaps incorporating some motion blur) of the foreground, make sure that the matte you supply is full-on white. Be careful for example of not having tiny holes in the supplied mask.

Important: Make sure that pixels are ZERO (not near-zero) where the **background** should be seen. Keying a foreground can sometimes leave a matte near-zero, but not actually zero. Also, make sure that the areas are absolutely the foreground are full-on (not *almost* full-on). Areas of transition between the background and foreground (the edges of a foreground object, for example), may be supplied values between full-on and full-off.

If a foreground matte is supplied, then it must have the same pixel dimensions as the color source; otherwise a **red image** will be produced to let you know that there is an error.

Please note that while using a foreground matte can dramatically improve results, **Twixtor** needs to calculate one or two additional motion estimations per frame (because **Twixtor** generates motion for the foreground, and motion for the background), so it can take more than twice as long as without a foreground layer. (See Cache Optical Flow)

You can have 3 FG layers, each has it's respective set of controls.

- **FG1 Matte Chan:** This specifies which channel of the FG1 Matte clip the mask should be retrieved from. Often this will be set to Alpha. If a single channel image is provided as input then this control is ignored.
- **FG1 Motion Sensitivity:** It is helpful to specify a foreground motion sensitivity that is separate from the main layer motion sensitivity (for example, something moving over a stable background), so it is provided here (see above for the Motion Sensitivity description that is used for the background layer).
- **FG1 Inv Matte Shrink:** This allows you to specify (in pixels) the amount the inverse of the foreground matte should be shrunk when tracking and creating the main (background) layer. If you see a dark line or similar artifacts where you know the edges of your mask are, just raise this value.

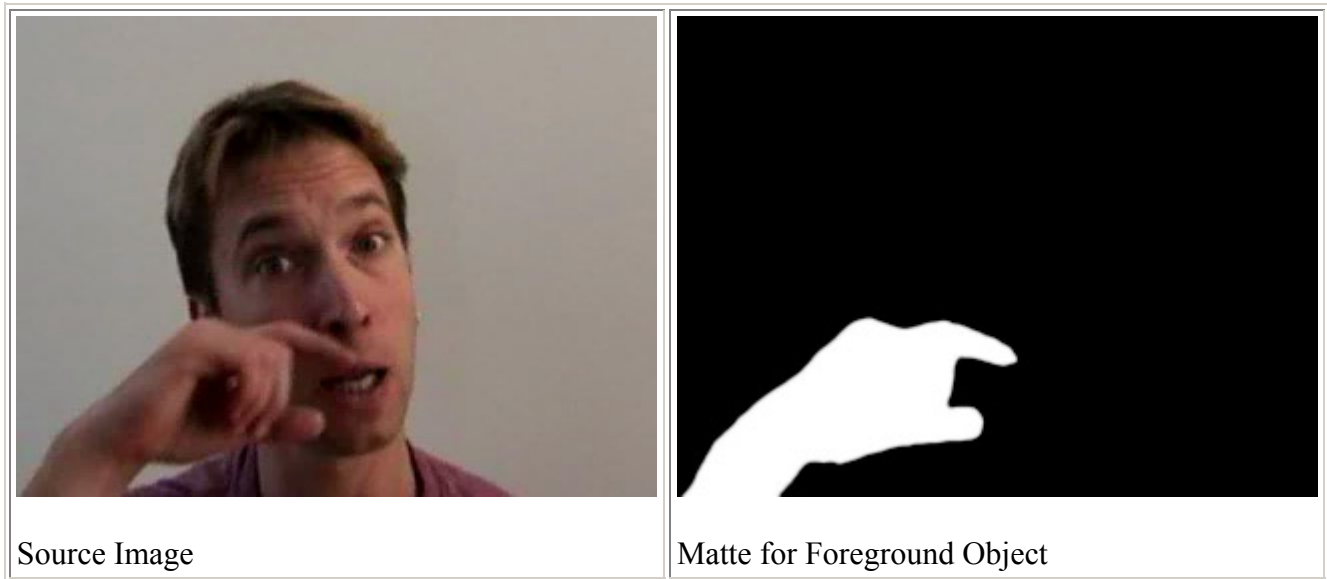
As you add foreground mattes for objects, note that the DisplayLayer menu option becomes more relevant.

DisplayLayer

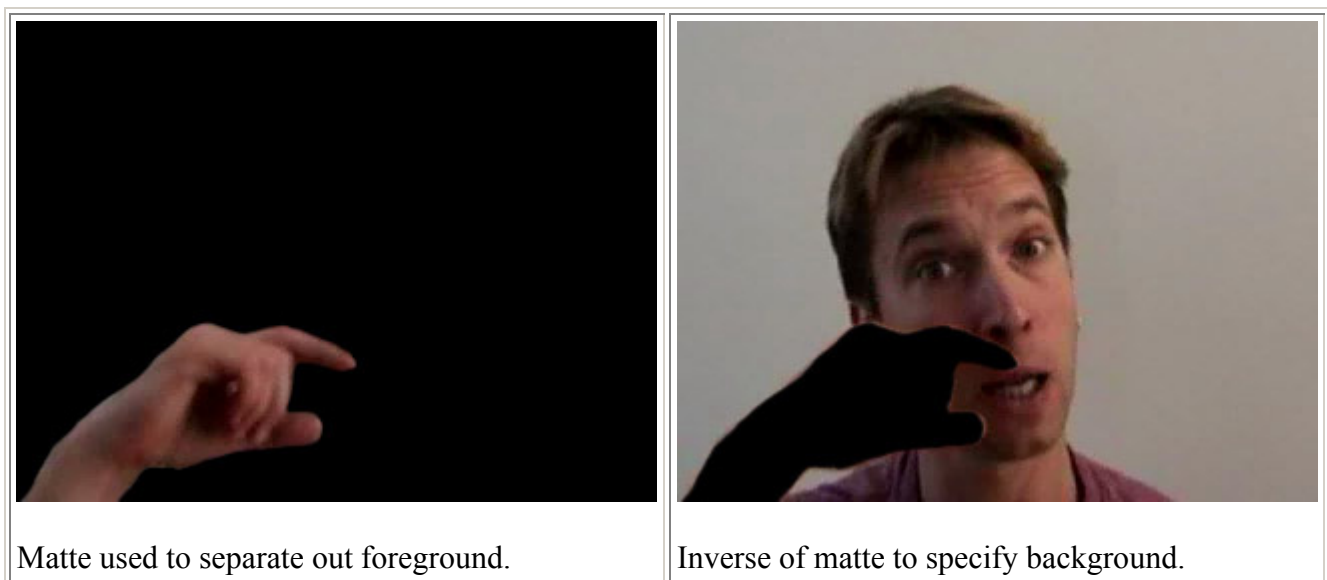
1. All: This setting displays the final **Twixtor** result. This setting has all the layers that you have specified (via mattes) retimed individually and composited back together.
2. FG1: When Display is set to TwixtoredOutput, this displays the top-most layer, retimed. If Display is set to ColorSource then the Display, in the source timing, is matted through FG1. In this way you can tell whether or not your FG1 matte is the proper matte. Similarly when Display is set to TrackSource, your TrackSource is matted using FG1 as a track matte.
3. FG2: When Display is set to TwixtoredOutput, this displays the second to top-most layer, retimed. If Display is set to ColorSource then the Display, in the source timing, is matted through FG2. In this way you can tell whether or not your FG2 matte is the proper matte. imilarly when Display is set to TrackSource, your TrackSource is matted using FG2 as a track matte.
4. FG3: When Display is set to TwixtoredOutput, this displays the third to top-most layer, retimed. If Display is set to ColorSource then the Display, in the source timing, is matted through FG3. In this way you can tell whether or not your FG3 matte is the proper matte. imilarly when Display is set to TrackSource, your TrackSource is matted using FG3 as a track matte.

5. Main BG: When Display is set to TwixtoredOutput, this displays the main layer (that is, the farthest-away layer), retimed.

In order to describe the foreground layer option, pictures are worth a thousand words! So let's say we have the following image, and matte for the foreground object:

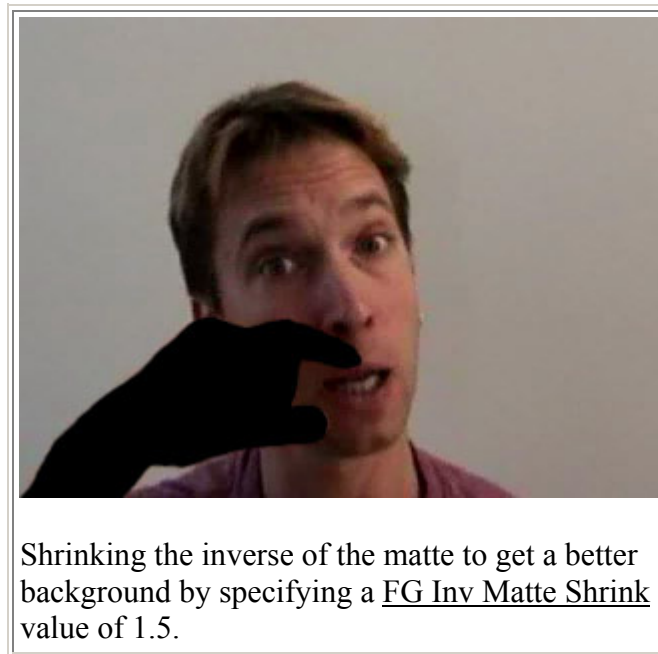


Using the matte for the foreground and the inverse of the matte for the background we get the following two pictures:



Note that while the matte specifies the foreground quite nicely, some of the pixels of the foreground spill into the background! By specifying how many pixels to shrink the inverse of the foreground matte (the FG Inv Matte Shrink value), we can reduce or

eliminate the spillage. By specifying a value of 1.5 we get the following picture:



Note you might ask, "won't that create a gap in my final output?" The answer is no! Through clever filling and compositing, we track the background separately from the foreground, and fill in missing information in the background as best we can!

If we display just the FG1 layer by setting the Display menu to FG1 Twixtored we get the image of the hand above. However, if we set this menu to BG Twixtored we see?



Twixtor will be compositing the foreground layer on top of this.

Splitting your input sequence into more than one foreground layer.

Let's assume you have a sequence with three foreground objects. Let's also assume that you have a matte for each object at each frame. We now describe how to improve **Twixtor**'s tracking by using these mattes.

The FG2 Matte, FG2 Matte Chan, FG2 Motion Sensitivity and FG2 Inv Matte Shrink settings work for the second foreground layer as described above for the FG1 settings. The same goes for FG3 settings.

Important: Mattes must be specified in order (FG1, then FG2, then FG3, in that order). If a matte is not specified for FG1, then FG2 and FG3 will be ignored. If there are no mattes specified for FG1 and FG2, then FG3 will be ignored.

Note that when you have more than one matte, the FG1 Matte should correspond to the object that is on top and then FG2 Matte should correspond to the next object, etc.

Also, it should be noted that the mattes for FG2 and FG3 should be drawn so that they represent the whole object on that layer, even if an object appears of it. For example, it might just be that the object on FG1 obscures not only the background but also part of the object on FG2. When creating the matte for FG2 you'll need to create it as if the object on FG1 was not there. Once again, a concrete example will be very helpful.

Lets say that we have the following source image, and that after applying **Twixtor** we have decided that we need to create mattes for 3 foreground objects:

- FG1: the arm on the right (and is the one that is closest to the camera)
- FG2: the torso
- FG3: the arm on the left (and is the one that is farthest away from the camera)
- BG: everything else in the background

For the foreground object we simply roto it:



Source Image

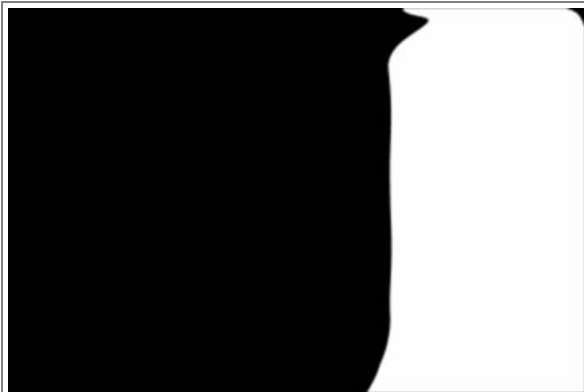


Matte for FG1

It turns out that in this case FG1 (the front arm) is also in front of FG2 (the torso).



In a traditional roto job you would probably produce this matte for FG2.



However, for **Twixtor**, we want to create the matte shown here; that is, FG2 (the torso) as if the foreground arm were not there.

For the case of FG3, the arm is slowly extending past the edge of the torso. As such, part of the arm is hidden behind the torso. As such, we want to extend the arm matte for FG3 a bit to tell **Twixtor** where we think it is hidden behind the torso. We only need to extend the matte by the amount that we think the arm is moving from frame to frame.



In a traditional roto job we would produce this matte for FG2.



However, for **Twixtor**, we want to create this matte; that is, FG3 (the rear arm) extended to show that the arm extends past the left edge of the torso.

Guiding Twixtor through the use of Track Points.

It is often useful to guide **Twixtor**'s motion estimation. This section describes how to guide **Twixtor** motion estimation using **Twixtor**'s Track Points.

IMPORTANT: **Twixtor**'s Track Point settings tell **Twixtor** which pixels correspond from frame-to-frame in the source image sequence. As such, Track Points need to be animated on the input sequence in the original source timing... so remember to animate the Track Point positions when Display is set to Source.

Note that a particular Track Point is used to guide the motion estimation for a layer when:

1. The Track Point's Use menu is set to appropriate layer, and is not used when set to the default Don't Use setting. However, keep in mind that you need to set the popup menu in the Source timing (make sure to animate the Use menu when Display is set to Source). **Nuke users:** Since Nuke (last time we checked) does not support animated menus, we display in the message tidbit that shows up when you roll over a parameter the menu values, be careful that these values are not interpolated. Note that 0 means "Don't Use" the tracking point, 1 means use tracking point as part of "FG1" (Foreground layer 1), 2 means use tracking point for "FG2", 3 means "FG3", and 4 means "Main/BG".
2. The Track Point is used when within the image boundary for both of the source frame times that are being used to create the output frame. If a given Track Point is off the image for both of the source times used to create a Twixtored output frame, then that Track Point is not used.

Sometimes it can be difficult to figure out when a Track Point is being used. However, know that **Twixtor** only draws guiding geometries when the geometry

(in this case, a Track Point) is actually being used at a source frame time. When not used, a Track Point is not drawn.

Also note that when drawn via the Draw Geom menu that a Track Point is displayed in the Track Point's color with a "+" symbol.

DrawGeom

When interactively guiding Twixtor it is often helpful to draw the geometry that is doing the guiding. As such, the DrawGeom menu tells Twixtor to draw the guiding geometry. If there is layer separation, then only the Track Points for the selected display layer's TrackPoints are drawn (layer selection is performed using using Twixtor's DisplayLayer setting).

Note 1: Track Points are only drawn if Twixtor's Display mode is set to Source or TrackSource (that is, with the original timing of the input footage).

Note 2: if a particular geometry is not being used to guide the tracking at a particular original source frame (via it's Use setting), then is also not drawn at that frame. In this way you can "debug" when geometry (Track Points) is being used or not to help guide the tracking.

An example: Let's say that we want to create a frame between the following two pictures.



Source Image 1



Source Image 2. Note the objects in the image have moves significantly to the right.

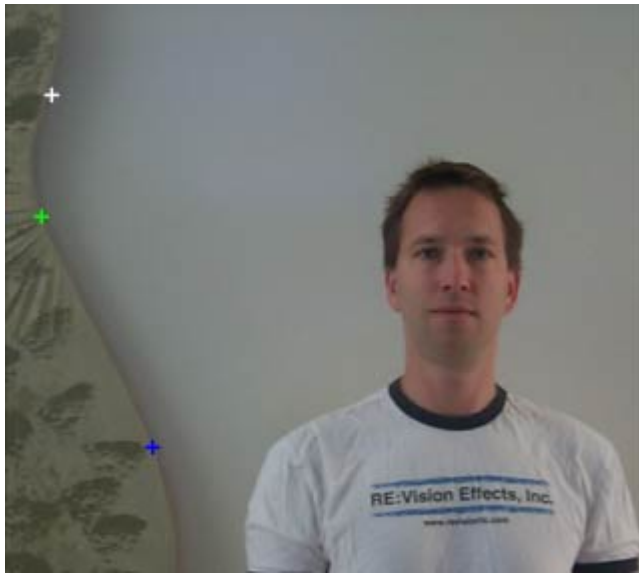
To create a frame that is halfway between the two images we set the Speed% of **Twixtor** to 50. The inbetweened frame, when Motion Vectors is set to Best, looks like this:



Inbetweened image with no tracking points used.
Note that many of the features do not line up satisfactorily.

In the next pictures, we set the Use menu for the first 3 Track Points to Main_BG layer, then animated them to be in the proper positions (as drawn by the "+"s below) for the two source frames, at the two source times.

Track Points		
<input type="checkbox"/> Pt 1: Use	Main/BG Layer	Points 1,2 and 3 used for Main/BG Layer.
<input type="checkbox"/> Pt 1: Pos	73.1, 220.8	
<input type="checkbox"/> Pt 1: Color	Green	
<input type="checkbox"/> Pt 2: Use	Main/BG Layer	
<input type="checkbox"/> Pt 2: Pos	73.1, 220.8	
<input type="checkbox"/> Pt 2: Color	Blue	
<input type="checkbox"/> Pt 3: Use	Main/BG Layer	
<input type="checkbox"/> Pt 3: Pos	22.9, 44.9	
<input type="checkbox"/> Pt 3: Color	Yellow	
<input type="checkbox"/> Pt 4: Use	Don't Use	Points 4 through 12 not used.
<input type="checkbox"/> Pt 4: Pos	-16.0, -14.2	
<input type="checkbox"/> Pt 4: Color	Yellow	



3 Tracking Points used, and positioned for the first source frame.



Same 3 Tracking Points animated at the second frame (with Display set to Source) to correspond to the same features in the first frame, at the first source time.



Inbetweened image with 3 tracking points used. Note that in this case we set the Warping Method menu to Forward because we used guiding geometry (and also because an object moved significantly offscreen).

What is Motion Vectors Create?

The Motion Vectors Create plugin makes available the per-pixel motion vectors calculated with RE:Vision Effects' proprietary tracking technology.

Motion Vectors Create Features:

- Calculation of motion vectors at each pixel.
- **Debug** mode that enables you to intuitively tell when the motion vectors are accurate.
- The ability to separate the source material into multiple layers that are tracked individually. Layers in **Motion Vectors Create** are specified through the use of mattes that you create (independently of **Motion Vectors Create**). Separate layers are particularly useful when objects in the scene cross each other. With **Motion Vectors Create**, up to 4 separate layers can be specified.

Before You Start, EXTREMELY IMPORTANT.

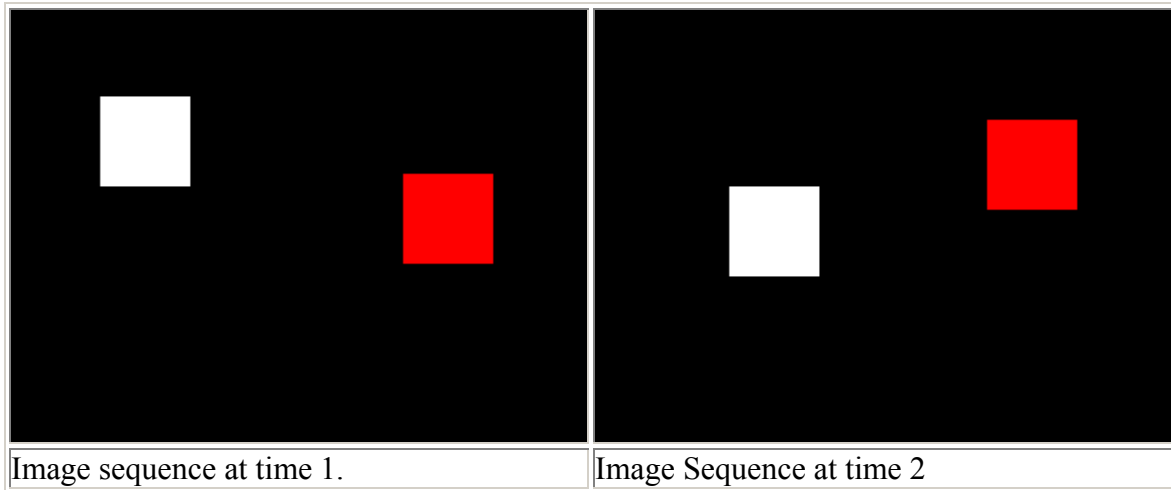
Common problems discussed here!

- **Important:** In order to maintain accuracy of the motion vectors upon output it is essential that you save the motion vectors in a lossless format in 16bpc or floating point. Also be aware of any color space conversion that might affect the value (for example sRGB to linear).
- The plug-in calculates the motion between two successive frames in a sequence, calculates the motion from one frame to the next, and outputs the vectors as an image.
- The plugin expects deinterlaced material (without fields, 3:2 pulldowns or duplicated frames)

Concepts and Terminology

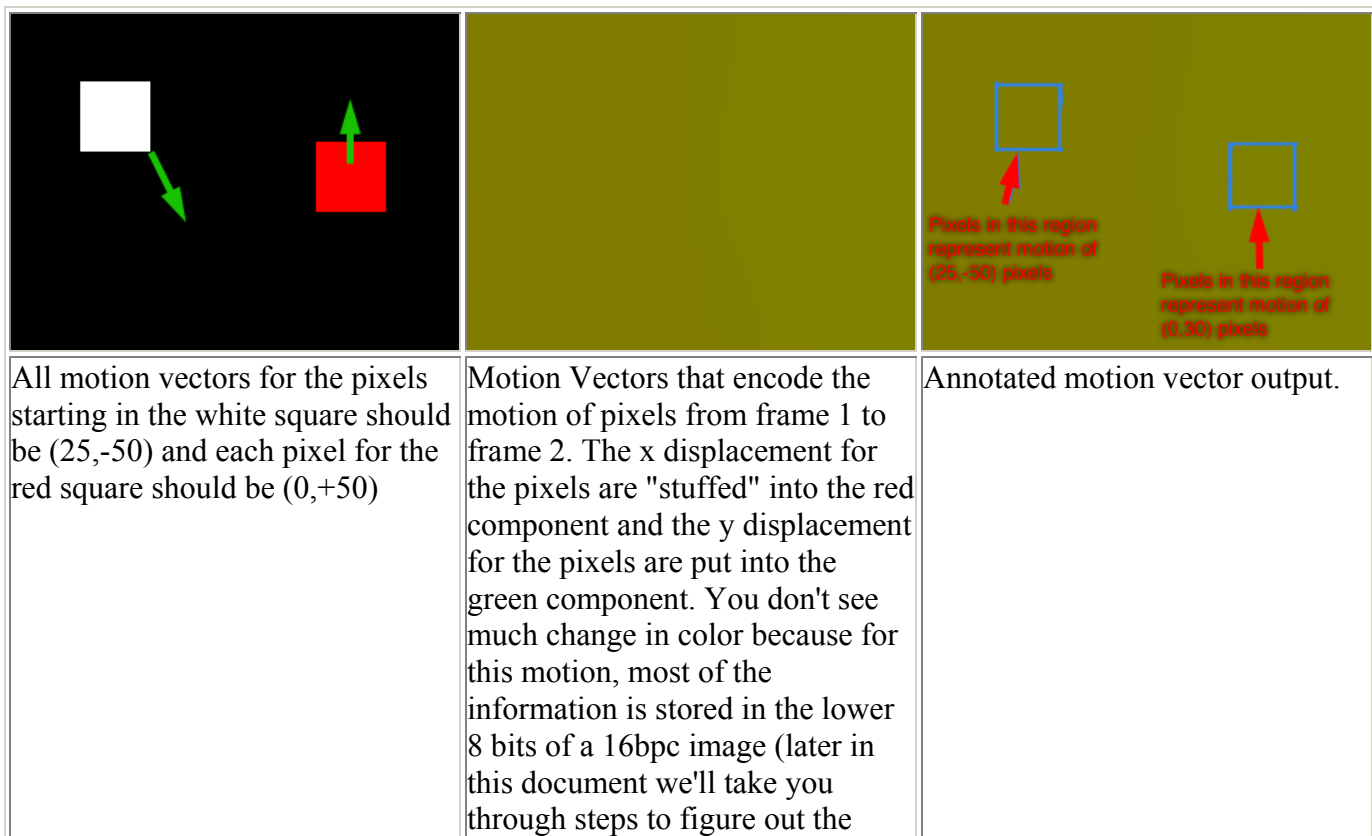
Motion Vectors Create allows you to create motion vectors by tracking every pixel in the image sequence. Motion vectors are created for each individual pixel. At each pixel the plugin determines the movement in both the x and y directions.

Let's look at two successive frames from an image sequence.



The white square has moved 25 pixels to the right (x direction) and 50 pixels down (-50 in the y direction). The red square has moved 30 pixels up (+30 in the Y direction). As such each output motion vector "pixel" should be (25,-50) for the white square and (0,30) for the red square.

The x motion for each pixel is stored in the red component of the output image and the y motion for each pixel is stored in the green component. The motion is stored as the amount of pixels moved.



Motion Vector Format

We assume that X is positive going from left to right. And the positive Y designates motion going UP (which is the opposite of After Effect's coordinate system, for example).

We assume that X,Y vector information is encoded in the red and green channels of the image. In addition we assume the vector information has been normalized so that both X and Y range from -1 to 1, presumably using some constant value to divide the X and Y component values. We'll call this normalization value the Max Displace value.

In 16 bits per channel, we assume that -1 corresponds to pixel value of 0 and that +1 corresponds to 65534. We have chosen to map (-1,+1) to (0,65534) because with this scheme we can represent a 0 displacement with a pixel value of 32767 (in a scheme that maps (-1,+1) to (0, 65535), a 0 displacement value corresponds to pixel value of 32767.5, which cannot be represented exactly).

- ```
// First, map -MaxDisplace to MaxDisplace to 0 to 1
float fred = ((x/MaxDisplace)+1)*0.5;

/* clamp values if needed */
if (fred<0) fred = 0;
if (fred>1) fred = 1;

/* assign pixel value */
unsigned short red = fred*65534.0 + 0.5; /* rounding is preferred to truncation, but that's your choice */
```
- ```
float fgreen = ((y/MaxDisplace)+1)*0.5;
if (fgreen<0) fgreen = 0;
if (fgreen>1) fgreen = 1;
unsigned short green = fgreen*65534.0+0.5.
```

[You can read more about our motion vector format here.](#)

Multiple objects and motion tracking

There will often be multiple objects in the image sequence moving in opposite directions. To help track the motion for each object individually (which can often produce better results), the plugin allows you to separate your footage into multiple layers by supplying mattes.

When the image sequence is not separated into layers (that is, you have not specified any mattes for FG1, FG2 or FG3), then there is only one layer: what we will refer to as the Main layer. When you separate the image sequence into two layers by supplying one foreground matte, then the foreground layer will be referred to as FG1, and the rest of the image will be referred to as the Background layer. Continuing in that manner, if you supply two matte sequences, then the sequence will be separated into 3 layers: FG1 (the topmost layer), FG2 (the intermediate layer) and the Background layer (which is the remainder of the image NOT in FG1 or FG2). Similarly you can separate an image sequence into 4 layers by supplying 3 matte sequences.

As such, the Background layer will be what is NOT in any foreground layers that are specified (via your mattes). If no foreground layers are specified the Background layer is also called the Main layer. A description of how mattes for layer separation should be constructed will be detailed below.

These concepts will become clearer as we discuss how to use the layer options of **Motion Vectors Create**.

Usage of Motion Vectors Create

MV Create Source Requirements

must not be interlaced material. Only progressive material can be properly processed. If interlaced material is supplied a **semi-transparent red image** will be produced to let you know that there is an error.

Note: vector images created need to be 16bit or floating point, because 8 bpc vector images are not recommended. Internally if inputs are of variable bit depths, they will simply be bumped to the highest bit depth.

Display

Choices are Vectors, Source, Debug

DisplayLayer

Choices: All, FG1, FG2, FG3, Main_BG.

The individual layer output settings (FG1, FG2, etc) allows you to view a particular single layer using the Display setting you've chosen. (or if All is selected, then all layers are displayed composited together) There are two reasons why you might want to view an individual layer:

1. The single layer settings allow you to see an individual layer's output without having to wait for all layers to be calculated and composited. By viewing a single layer you can determine whether or not your interactive guidance for a particular layer has been successful.
2. Internally, **Motion Vectors Create** calculates a full frame of motion vectors for each layer. By viewing a single layer, you will get the motion of that layer extended to entire frame. This can be useful, for example, for input to a particle system where you want particles to follow the motion of an object even when it is not strictly on the object.

NOTE: When Display is set to Vectors and you select a single layer to display, the matte for the visible portion of the layer is put into the blue channel. Strictly speaking we do not pay attention to the blue channel when looking at motion vectors (for example, in the plugins of **ReelSmart Motion Blur** or **Twixtor Pro** that use vector input). Because the blue channel is free to be used as we wish we thought it would be useful to put the visible portion of the layer into the blue channel for your use later in your compositing pipeline. Even though the blue channel holds the visible areas of the layer, please note that the motion vectors in the red and green channels are valid for every pixel, independent of what's in the blue channel.

Debug Mode

This setting allows you to interactively view the accuracy of the motion vectors. We will discuss this option in more detail later in the manual. For now, note that this mode is only active when the Display menu is set to “Debug”.

Tracking Quality

The two supported modes are “Best” and “Ultra”. “Best” is like the Best quality in Twixtor while “Ultra” is a different algorithm that will create higher precision motion vectors yet takes much longer to render. “Ultra” can also be more sensitive to noise.

Image Prep

A menu option with 2 choices

1. None
2. Contrast/Edge Enhance

The default image preparation is to do nothing special with the source material that is used for tracking. If your material is dark or has poorly defined edges then **Motion Vector Create** might perform better if you choose Contrast/Edge Enhance. Note that the Contrast/Edge Enhance setting may perform more poorly than using the None setting on material that already has reasonable contrast or has many thin edges close together.

Track Frame

Two choices: Previous and Next. Set to Previous to calculate the motion from the current frame to the previous frame. For example, if you are at frame 15 then setting Track Frame to Previous calculates motion vectors from frame 15 to frame 14. Likewise, setting Track Frame to Next calculates the motion vectors from the current frame to the next frame.

Max Displace

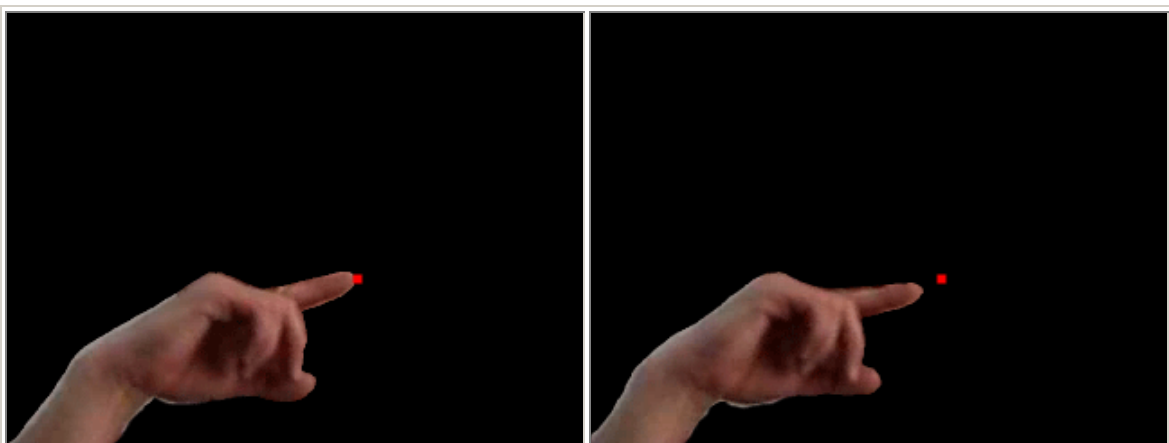
This setting controls how the internal floating point motion vectors are transformed into 16bpc or floating point images. We describe the motion vector format and a complete description of MaxDisplace [here](#). If you don't want to read all about the motion vector format right now simply set Max Displace to 2048 (the default is 64).

If you are using **Motion Vectors Create** for the first time we'll assume that you are not yet splitting up your footage into layers with the use of mattes. If this is the case then you have only 1 layer, the Main layer. So the following two settings apply to what you are doing:

Debug display mode

So you've set up the settings above and are interested in knowing how accurate the motion vectors are. To do this, you first set the Display setting to Debug. Then you have 1 of 5 debug viewing modes.

Let's say that you have the following two successive images in a sequence.



Images at frame 10 and 11. Red dot included to show that the hand moves to the left and slightly down between frames 10 and 11.

When viewing the motion vectors, we get the following image:



So how do we tell if the motion vectors are tracking accurately or not?

That's where the Debug Display mode comes into play.

Let's say that we are calculating the motion vectors to the Previous frame. That means at frame 10 we calculate the motion from frame 10 to frame 9 (and at frame 11 we calculate the motion from frame 11 to frame 10).

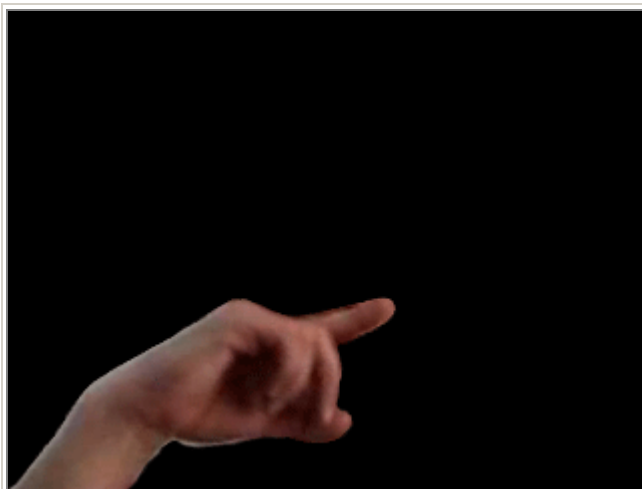
In the Debug mode we might consider when at frame 11 to calculate the motion vectors from frame 11 to frame 10 and then warp frame 11 using the motion vectors. In essence this would use the motion vectors calculated at frame 11 to warp frame 11. Once frame 11 is warped then it should match frame 10 because, after all, we have just calculated the motion vectors from frame 11 to frame 10. However, if we were to composite the warped frame on top of the original footage, we would be 1 frame off and wouldn't be able to meaningfully compare the images.

For example, if at time 11 we warped frame 11 so that it matched frame 10, then if we composite the warped Debug sequence over the original we get the following image even when the track is perfect:



The comparison of a warped image with the source image (we've composited the Debug sequence over the original). This image shows what you would get in Debug mode at frame 11 if the plugin were to warp frame 11 using the Previous motion vectors, and then composite over the original at time 11. This undesirable because we can't compare the warped image to the source sequence without adjusting the timing of everything, which can make the workflow that much more tedious.

If fact what the Debug mode does, **when at frame 10**, is use the Previous motion vectors calculated at frame **11**, and warps source frame 11 to frame 10. When we do that and composite over the original image at frame 10 we get the following image.



What the debug does at frame 10 when Previous motion vectors are being calculated: it uses the motion vectors at time 11 and warps the source

image at time 11 to match frame 10. When this warped image is composited over the original you get a meaningful comparison.

Of course we show an example with an alpha matte so that you can see the undesirable comparison. If you are viewing a full frame image you might want to place the src image on a track below the plugin-applied sequence so that you can turn on and off the track that's on top (the one with **Motion Vector Create**) to see how the warped version of the sequence compares to the original sequence.

Important Note: When calculating motion vectors to the Previous frame and you are in Debug mode, then you are really debugging the motion vectors calculate at time N+1 when at frame N in the timeline. Similarly when calculating motion vectors to the Next frame and you are in Debug mode, then you are debugging the motion vectors at time N-1 when at frame N in the timeline. When you are calculating motion vectors to the Next frame you are at the first frame in a comp or sequence then the plugin produces a green frame to let you know that there are no vectors for the frame time before the first frame.

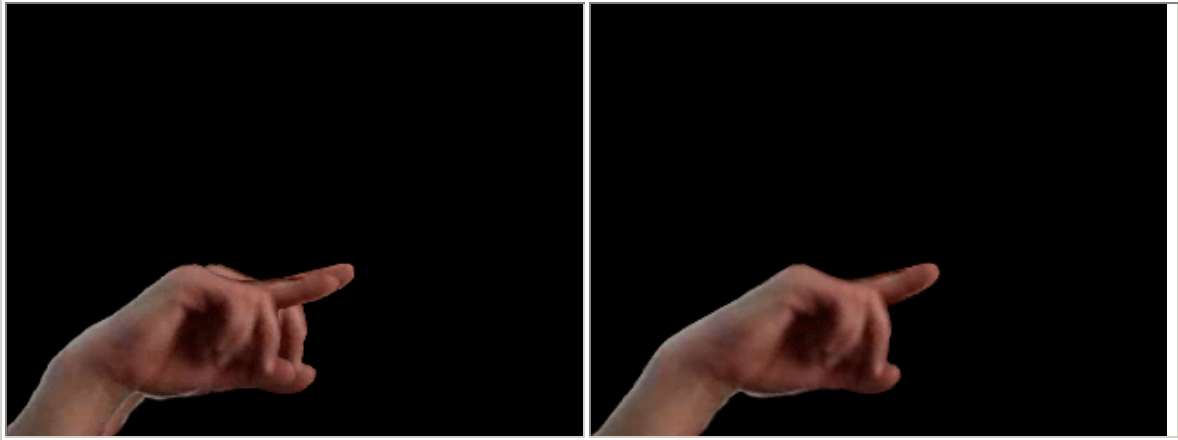
Now we describe the 5 debug modes. For all Debug modes we use the motion vectors to warp the image as described above. The different Debug modes differ in how image is compared to the original image sequence.

Important Note: For the following examples, we've purposely turned down the motion sensitivity in the "inaccurately tracked" examples, handicapping the tracker on this example so that it doesn't track well. This allows you see the difference in the Debug display modes when tracking is working well and when it isn't.

Warp: No comparison is done. The warped image is simply output. It is expected that you will compare the result to the original sequence by toggling the Display mode from Source to Debug, or that you will place a copy of the image source sequence on a track underneath the plugin-applied version and use the host application to toggle between the Debug displayed version and the original to see how well the motion vectors have been calculated.

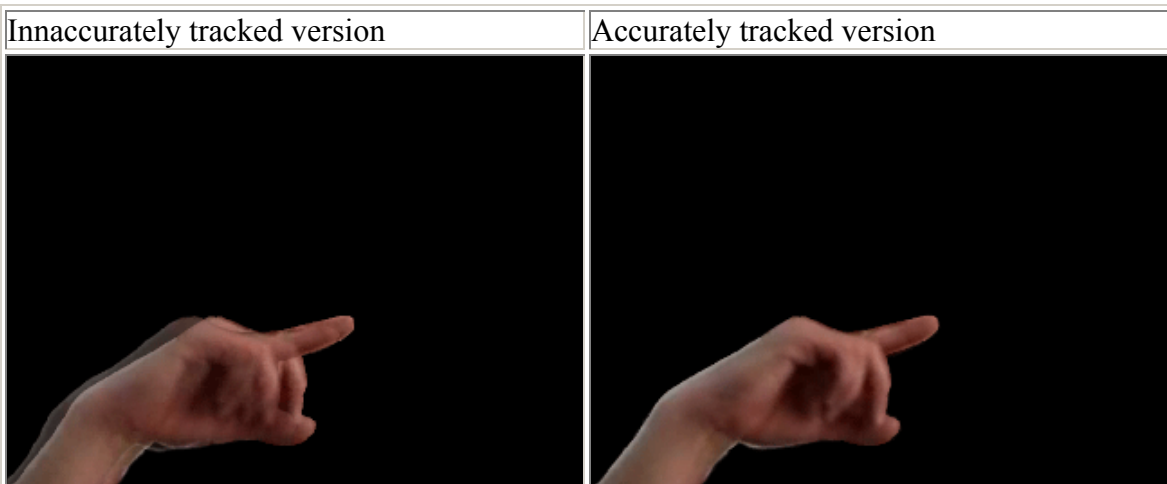
Innaccurately tracked version

Accurately tracked version



Debug mode: **Warp**. In both cases the Debug mode version is composited over the original sequence. You can see the "obvious" inaccuracies in the version on the left.

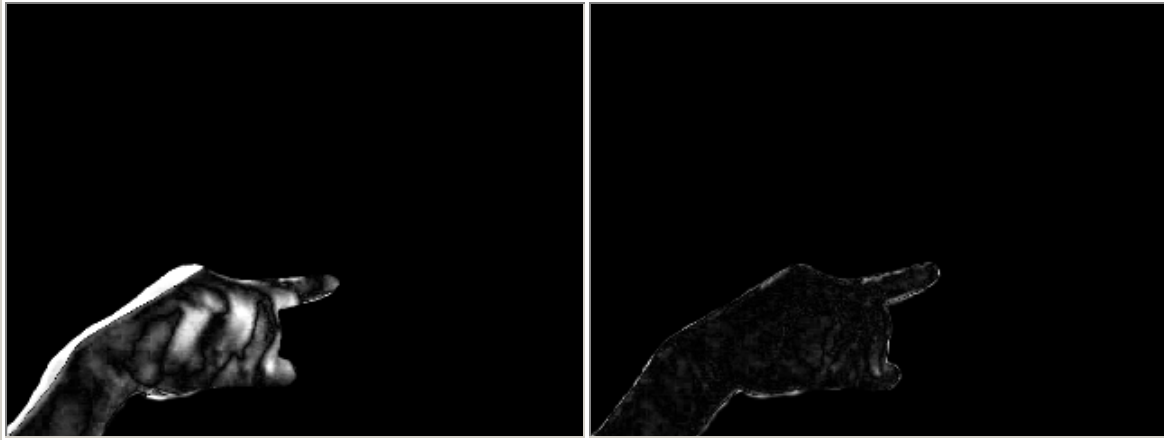
Blend: In this mode the warped version is blended at 50% with the original sequence. Where there blurred or double-edges you know that the tracking is not performing correctly.



Debug mode: **Blend**. You can see the "obvious" inaccuracies in the version on the left, which show up as double-exposed portions of the image.

Diff: In this mode the warped version is differenced with the original sequence, and then sent through a gamma correction that expands the differences so that they can be more easily viewed. Inaccuities show themselves as whiter parts of the image.

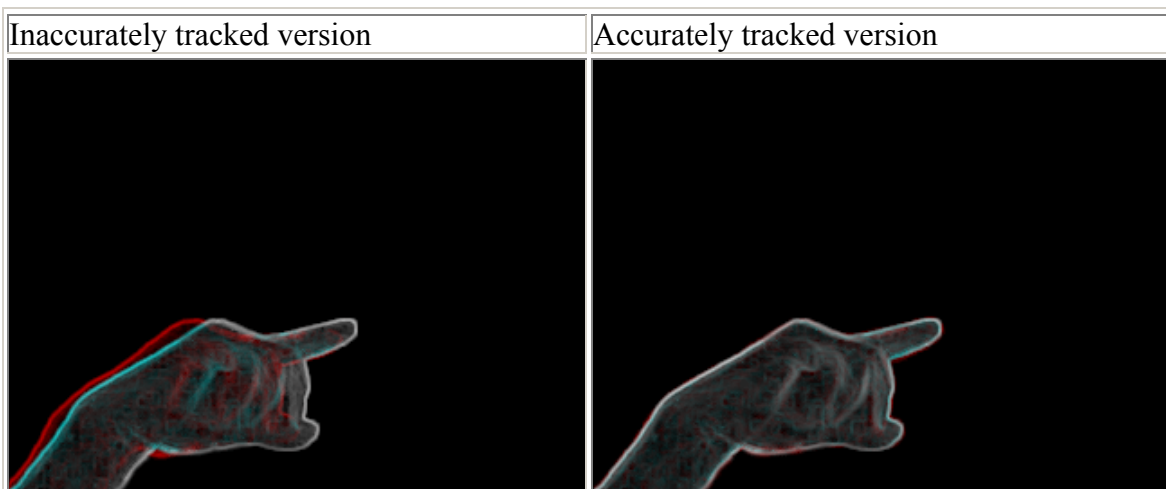




Debug mode: **Diff**. You can see the "obvious" inaccuracies in the version on the left, which show up as bright white.

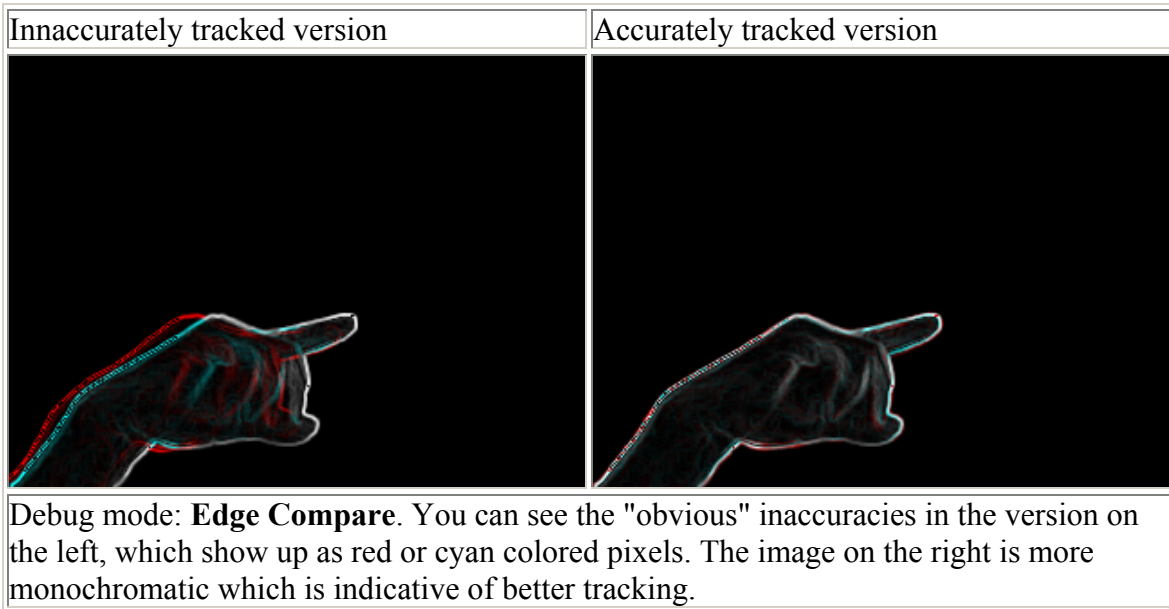
Detail Compare: In this mode we put into the red channel a high frequency pass of the luminance of the warped image . We then do the same with the original source image and put the result into both the green and blue channels (a cyan color). Tracking is poor when you see either red or cyan coloring. When the tracking is more accurate, the features will line up and create white (actually, where the tracking is good, pixels will be produced that have variable luminance but will have no coloring. Said another way, pixels where the tracking is accurate will be shades of grey).

You can switch between two filtered images (that is, the warped version and the original source image) by viewing only the red or green channels using the host applications viewing tools.



Debug mode: **Detail Compare**. You can see the "obvious" inaccuracies in the version on the left, which show up as red or cyan colored pixels. The image on the right is more monochromatic which is indicative of better tracking.

Edge Compare: This mode is similar to the Detail Compare mode. However, in this mode we use a less-sensitive high pass filter. This is useful on image sequences that have a lot of noise or other confusing high frequency content in the images.



The other settings of this tool behave exactly like Twixtor PRO, so refer to that part of the documentation for these.

Using Input Motion Vectors to retime footage

Important: Nuke users. When importing Motion Vectors image sequences, make sure you press the [x] RAW button in the Read / Loader, otherwise the data might be converted in your back.

Setting Twixtor to use input motion vectors.

Note that the Retiming Settings are used just as described in Chapter 1. These settings set up how the timewarping is set up.

Prev and Next Vectors

Prev Vectors: This setting specifies the input clip to use as motion vectors that describe motion from each current frame to the **previous** frame.

Next Vectors: This setting specifies the input clip to use as motion vectors that describe motion from each current frame to the **next** frame.

Note: If either Prev Vectors or Next Vectors are connected, then **Twixtor**'s internal tracking is not used.

Note: Because parts of objects may become obscured or unobscured between two adjacent frames, or because an object may move from on screen to offscreen between successive frames, the **previous** frame motion vector sequence may not simply be the inverse of the **next** frame motion vector sequence. As such the plugin allows you to specify a sequence for both Prev Vectors and Next Vectors for more accurate time warping.

The additional settings that need setting are in the Vectors In tab:

MaxDisplace

The maximum displacement used when you created the motion vector image files (see the above discussion on the file format).

VecScaleX, VecScaleY

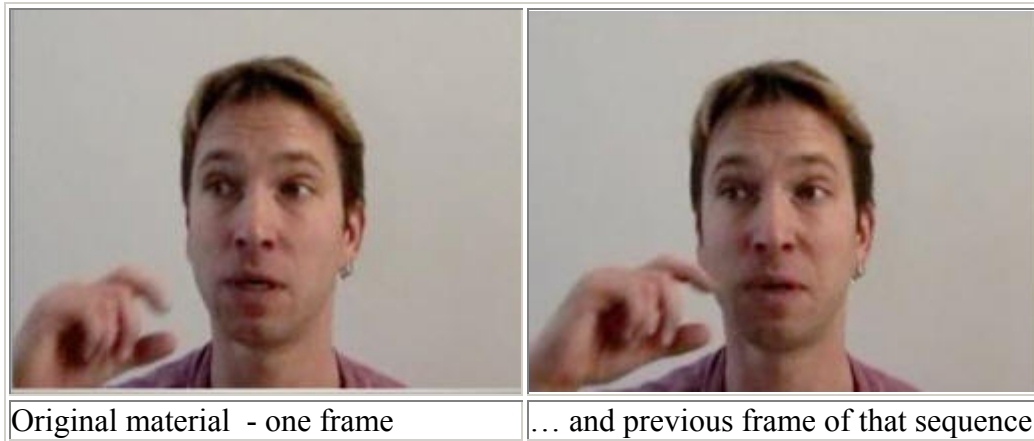
Used to scale the motion vectors internally once they've been converted to floating point. Note: a -1 value for VecScaleY can correct for the Y component of your motion vectors pointing in the incorrect direction. Note all motion vectors from CG are not necessarily going forward. You should test with a simple rectangle animation and play with the scale X and Y -1, and 1 to find once and for all which way the motion vectors go (yes forward or inverse and scale -1 or 1.0).

Multiple Layers and Motion Vectors

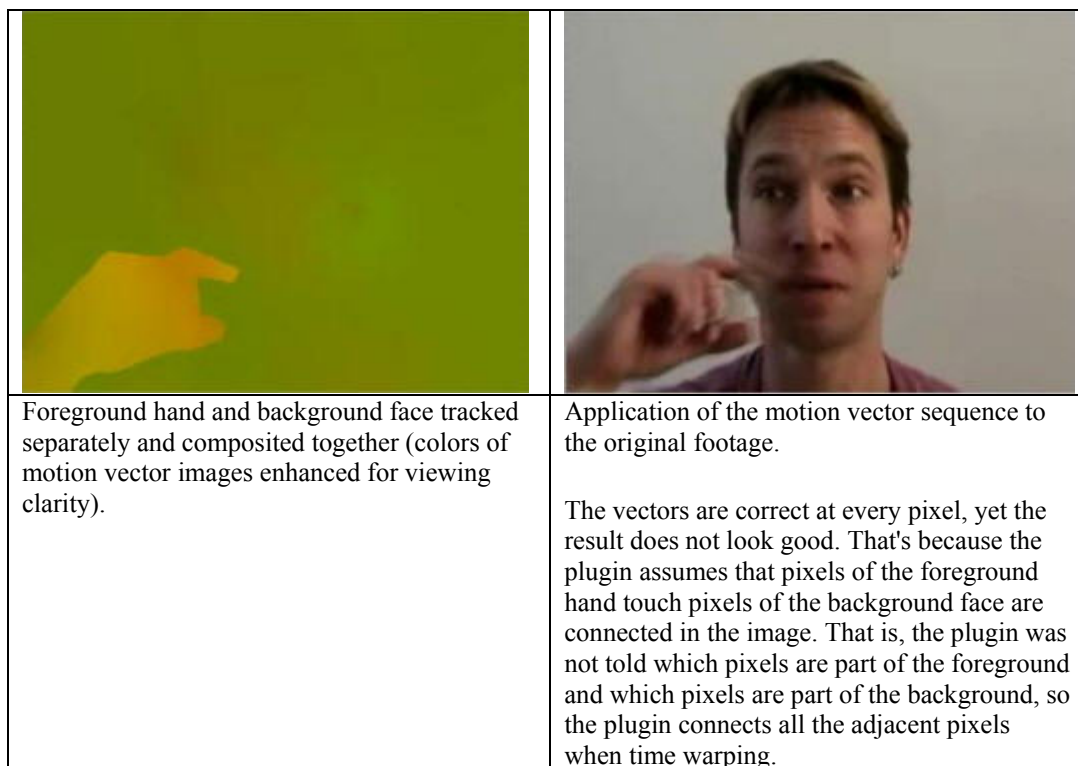
Let's say that you have sequence with multiple layers, such as the following sequence with two objects: a background face and a foreground hand. This is a live-action sequence but the setup below also applies to any 3D scene with multiple objects. What follows does not always happen, the following is to indicate what to do if it happens. However since it can happen even with 3D scenes it is suggested if you use this technique all the time that you simply time-warp each object individually and composite the retimed sequences after the application **Twixtor** to each object.

Examples of incorrect and corrected setup of **Twixtor** using input motion vectors when multiple object motions have been calculated and composited together before application of **Twixtor**.

Lets say that the scene has been tracked so that the hand and face have been tracked separately and composited together.





If we simply apply the motion vectors to the input sequence and time stretch by a factor of 3, we get the following result.



To get good results, you must also provide the plugin with a foreground matte the effectively shows the plugin which pixels in the source image are part of the foreground and which pixels are part of the background.

Lets say that the scene has been tracked so that the hand and face have been tracked separately and composited together.

	
<p>Foreground matte (passed to the plugin in the <u>FG1</u> settings).</p>	<p>Resulting slow-motioned sequence (by 3x) when the motion vectors and the foreground matte are used. Because the plugin can determine which pixels are part of which layer, the resulting sequence is much more satisfactory.</p>

Does the motion vector sequence need to be the same resolution as the source being blurred?

Yes.

If they are not the same resolution? You can resize the motion vector images to the same resolution as the source being time warped. However you have to apply the proper VecScale X and VecScale Y. For example if your image size is 640x480 and your motion vector images are 320x240 and you resize the motion vector images to 640x480, then VecScale X and VecScaleY need to be set to 2 and 2.

PROBLEMATIC FOOTAGE

If you have not used **Twixtor** yet, this section might prove to be a little bit esoteric. This information will be of more use to you as you become more familiar with **Twixtor** and want to understand how to generate the best possible results.

It might happen that **Twixtor** might do poorly in specific instances. The idea here is not to discourage you and tell you it does not work but, rather, to give truth in advertising. As you become an expert user, you learn to predict the kind of material that can cause problems.

1. "Picket Fence": A very regular pattern in motion (for example someone wearing a t-shirt with fine stripes) with an object moving in front of it (e.g. the same person hand for

instance) might confuse the motion vector calculation. Any very structured pattern rotating can cause "blind spots" for the analyzer.

2. "Transparency": Overlay of semi-opaque surfaces might create unexpected results. Some cameras for example will streak under fast motion and that can create disappointing tracking results.
3. "Short Interval Defects" : Sudden global illumination change (e.g., a flash), strobing, dust, ... can create unexpected / undesired results. Also, if there is a piece of hair or a scratch on the scanned film for a frame this would influence the tracking so you really should try to clean such defects before processing.
4. "Fields": We provide some basic field handling. Always remember that by definition, even and odd scan lines in material with fields are considered half a frame apart timewise. Because fields are spatially a line offset from each other, regions where there is no motion and horizontal lines might produce a slight up and down motion in phase with the source frame rate.
5. "Duplicated Frames": **Twixtor** does not provide automatic duplicated frame detection support. As such, you will need to remove duplicated frames before the application of **Twixtor**. Also, you should be aware that if your material has 3:2 pulldown you should remove it beforehand. The same applies to animations on 2's. If you leave the duplicate frames (or fields, in the case of 3:2 pulldown) the freeze-frame, inherent in the duplication of the frames, will be stretched (or sped up, as appropriate).
6. "Alternate Motions": When motions going in different direction are layered it is possible that the dominant motion affects (spills into) the background motion.
7. "Specular Highlights": If you have moving lights, e.g., a shiny object that reflects the light as it moves, it might cause problems because when the motion estimator attempts to match two images, as the motion estimator will tend to follow the highlight as if it was an object. (of course, sometimes this is what you want).
8. "Ultra-Fast Structured Motion": We are very particularly perceptive to human actions. We have sometimes seen that certain complex rapid motion such as someone doing a frenetic dance creates interframe displacements that are just too big for our motion estimator to resolve satisfactorily. When planning a shoot for an effect that involves **Twixtor** in the pipeline, consider that for **Twixtor** fast articulated motion should be easier for front facing subject then sideway views as there will be less pixels traveled per frame on the screen, which is really the only thing that **Twixtor** cares about.
9. "Occlusions": Problems caused by object motions tend to be one frame problems and localized in an area of the frame which is called an occlusion, which is some pixels that you see on one frame but are not visible on the other as a result of camera and/or that object motion.

10. "Limited Reach": As a rule of thumb considers that **Twixtor** will be most accurate for pixel displacements that are a maximum of 5% of your image resolution (for 720x486, this comes to a maximum horizontal displacement of 35 pixels or so). With displacements larger than that **Twixtor** will start to become less precise as it tries to separate motions from one another in an image sequence.

11. "Compression Artifacts": Certain video coding techniques such as DV compressors use 8 by 8 pixels blocks (intraframe) based compression. What this means for you is that if for instance you have a sharp edge that moves, it will switch of 8 by 8 block and therefore locally it's neighbor values will be all different (substantially not like 2-3 values over 255 but sometimes 40 off near an edge). This is why green screen like setups perform badly with DV... Without smoothing the source, this can sometimes certainly create tracking problems.

MORE AESTHETIC / COSMETIC CONSIDERATIONS:

Gee! It breaks on these 3 frames what do I do?

* All solutions mentioned are not necessarily available as OFX plug-ins at this time.

Now that we have discouraged you, let's help you to be creative in repairing problems. We know that most of the time, there is nothing we can do to prevent problems at the source because the material already exists, and you need to fix it. So let us share a bit our experience. Experience we acquired playing with this kind of process in production, in trying to make a product that works, and in supporting our users. There is an astronomic quantity of possible images and it's easy to develop assumptions that prove not true if you always work on the same few clips. So, so you know we fully appreciate short movies (5 frames is usually enough) where **Twixtor** fails somehow, this is how we learn, and can hope to make an always better product.

1. 1 or 2 frame defects on source material: Sometimes you receive a movie with one or two frame defects (could be a flash for instance). Take the good frames on both ends of the bad ones and slice them together into another comp, create a short **Twixtor** clip of the proper duration and reinsert it back over the bad frames. Naturally if you also want to apply **Twixtor** to the whole movie, perform the repair before applying **Twixtor** to the whole sequence.

2. Splotches: Although we try to accommodate for separate layers with the Crossing Objects?/Separate Motion? button there are times when you will still get some splotches (unwanted warping artifacts). You can try one of two things:

- In applications that support the animation of pop up menus, you can animate the Motion Vector Quality menu so that it is set to Best for the non-problematic areas

and set to None for the problematic areas. For host applications that do not support the animation of pop-up menus (like FCP), you can run **Twixtor** twice (once with Motion Vector Quality set to Best and once with Motion Vector Quality set to No Motion Vectors, for example) and simply replace the problematic images of the Best setting with the images made with the None setting. If your problem only occurs in a small part of the image, you can hand-create a matte to relace the bad portions of the "Best" version with the corresponding parts of the "No Motion Vectors" version, using the compositing functions of your application. You might want to do a soft matte for nice blending. And naturally if you have the **Twixtor Pro** version, you can provide Foreground/Background separation mattes to **Twixtor**, which you should do probably in a manner that respects the shape of the object in the foreground. Additionally, in the **Twixtor Pro** version you can provide guidance to the tracking (see the **Twixtor Pro** manual for more info).

- Try turning down the Motion Sensitivity setting for the problematic portions of the image sequence.

3. Ghosts: Since errors such as ghosting are usually due to occlusions (pixels that were not visible in the previous frame, so don't have correspondence in the other frame due to motion), they are usually easy to paint out so when the result is played in motion no one will see it. Some camera motion such as driving on a bridge tend to create a multi-frame ghost which usually is not too problematic to paint out in apps like Commotion or Combustion that have clone offset paint tools. You might also prefer to not apply motion blur at this stage, paint/fix and then use **ReelSmart Motion Blur** to add some motion blur.

4. Using existing mattes: Sometimes, rotoing out a fast moving foreground object will make the background very stable and produce tracking data without the foreground motion influence. If you have the luxury of having mattes for a layer, and don't have the PRO version, you might want still to try to apply **Twixtor** to the foreground and background separately (so that the motion tracking influences on the layers go away or are reduced by Twixtoring them separately). Note in some cases you would need to fill the foreground hole with some form of clean plate image so it has the general motion of it's surrounding area. This process is somewhat what the FG motion masking services of the PRO version would do for you automatically.

5. Shakes...: Although by definition we deal with camera shakes... note that when you slowmo something you will also slowmo the camera shakes and vice-versa, so you might consider to pre-stabilize the sequence on which you want to apply **Twixtor**.

6. Noise, compression artifacts: Our algorithms should not be too sensitive to small grain noise at the analysis level, however it might look weird to extrapolate a lot of frames per frame on material that contains a lot of noise (as it essentially slows down the noise also) so you are advised to remove some noise in the color source, perhaps running a form of smoothing or noise-reduction filter before. With material (such as some "really damaged" MPEG 1 material), lower accuracy motion estimation might even produce

more desirable results. Also with DV-like sources, avoid nearest Frame Interp mode. The smoothing of blending actually reduces noise.

7. Very Large SlowMo: The amount of clean frames you can generate is often more a factor of the action in the scene as there is probably no answer to the question of how many frames can be inbetweened with a process like this, as it depends... As a rule of thumb, after 5% of the image size displacement for a single frame, consider yourself lucky if it works. (Again there are no absolute answers). Perhaps also you should consider that there is a limit to how much you slow down things as even perfect inbetweening with a computer animation systems can look weird as it won't have any local dynamics, that is it becomes similar to doing global shape key framing, everything will sort of move at the same rate.

8. Softening: Another related thing to watch out is that you will notice that inbetweens will "soften" the texture details. The thinking is that it's sort of OK as you don't need as much detail on moving elements (you assume you will have some motion blur) and the image where it does not move will remain very clear. If unwanted softening occurs, and the Weighted Blend is not enough, you can consider little post tricks including slightly sharpening (maybe simply an unsharp filter, such as one found in our [SmoothKit](#) plugin set) and adding a bit of noise... Depending on what you are doing this might be more or less realistic.

9 Handling Multiple Motions: A "rough" garbage matte roto that isolates the foreground object might be necessary on the few frames where the problem occasionally occurs. After applying **Twixtor** to the frames with the garbage matte (on both foreground and background, with inverse mattes of course!), the inbetweened frames can be recomposited back together. When you make a garbage matte for this purpose, you can typically be a bit more "more fat" than the usual roto matte for compositing. In general since Layer selection is not animated it might be simpler sometimes to run the whole thing once and then animate something like the Motion Vector Quality or Motion Sensitivity and then cut back the relevant parts of the two comps together.

Of course, remember that **Twixtor Pro** has many advanced features specifically designed to handle crossing objects and multiple motions.

10. Fast Action: So, with fast action content, you will probably get better results with **Twixtor** from material captured with a 60i FPS camera than with a DV camera that does 12 of 15 real FPS, because the same action will move farther, per frame, in the 12fps footage than in the 30fps footage. If you have the option of shooting progressive or interlaced video, a basic rule of thumb might be if you have fast action (or because you are running with a hand-held camera) shoot interlaced (because we construct 60fps from the 60 fields, which gives us more frames to deal with, which reduces the motion per frame), otherwise shoot progressive (because we don't have to reconstruct a frame from a field, you will get better quality results).

11. Changing Field Order: While **Twixtor** will transform the footage from Upper to Lower field first (or vice-versa), you may want to consider doing the field order conversion first, then applying **Twixtor** with the same input and output field order. The most effective way to change dominance would be to drop the first field, and then reinterlace the fields starting at field 2 of the footage. By doing the field order conversion first you more likely reduce sampling artifacts by **Twixtor**'s creation of whole frames from individual fields. Of course, your mileage may vary :-).

12. "Film Look" and Other Fields Related Problems: If Fields cause you problems, remember that we offer also a full [FieldsKit](#) which allows to flexibly organize your workflow to possibly create a better deinterlaced source for **Twixtor**. Note that some people want to simulate a telecine look by going down to 24 fps and then adding a 3:2 pulldown. This is certainly possible and [FieldsKit](#) will eliminate the need to create an intermediary render. But, it is our experience that you obtain something closer to a film look by simply Deinterlacing and applying ReelSmart Motion Blur and in that case maybe you don't need **Twixtor**. For more on our products and "film look" issues, please read our film look FAQ ([located here](#)).

13. Flickering: Similarly weird flickering as sometimes happen with problematic film capture can have an impact. Note **Twixtor** mostly uses the black and white information and is by default hardwired to video brightness, which largely discounts blue. Sometimes just doing an RGB invert of the color source and using it as the tracking source will help. So particularly for shots that have a lot of blue, it might be worth to supply an alternate track which remaps colors and pushes up certain values. This also applies with certain dark shots (low-lights level), it might be worth to expand their range and lift them up (making sure not to clamp values that would make entire regions "look the same" to **Twixtor**'s motion estimator).

Twixtor, ReelSmart, FieldsKit and SmoothKit are trademarks of RE:Vision Effects, Inc.
Other product names might be trademarks of their respective owners.

Copyright 2000-2009 RE:Vision Effects, Inc.

