The Nightvision Project

What's the point of learning all this stuff if you can't have some fun with it? And what better way to have fun than by challenging yourself in the process. That's the goal of the Nightvision project. I'll be honest up front: there are some tricky things in here. While it kind of *looks* like a video montage, there's far more to this project than anything we've done yet. We'll go over detailed steps where appropriate or just discuss approaches when that makes more sense. Either way, you should get a lot out of this one. It pushes the envelope.

The Concept

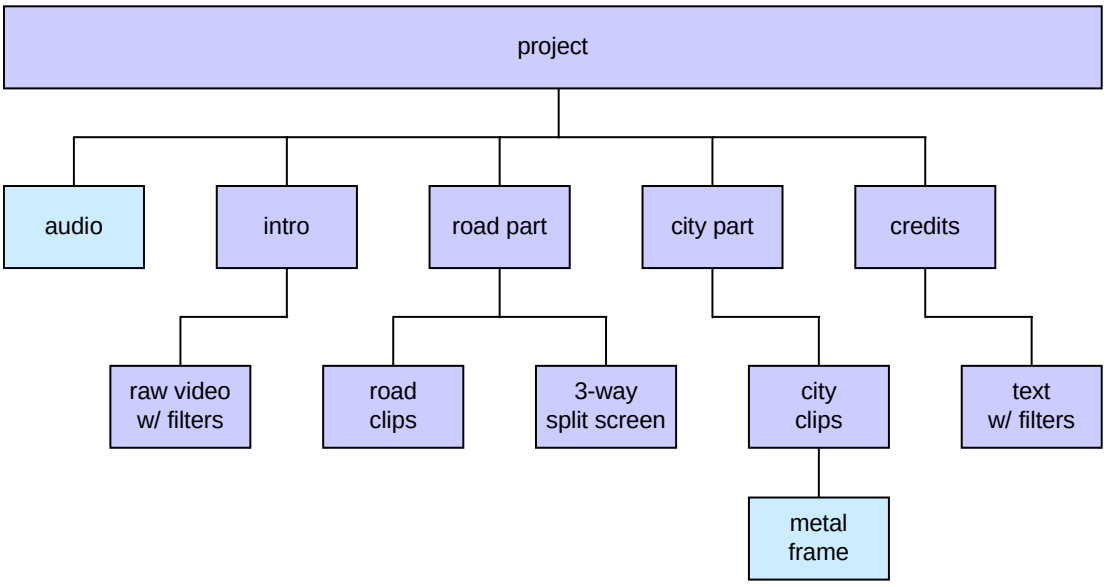
I wanted to create a video that didn't look like the typical "video." I wanted something that could demonstrate what VideoStudio was capable of, but that otherwise wouldn't be immediately obvious on the surface. I knew the project had to be uptempo, fast-paced, and be littered with special effects: but not to the point where it *looked* like it was littered with special effects. If I wanted to do that, I could have just dragged every F/X transition from the Library to the timeline and called it done after two minutes.

If you haven't done so already, go watch the video now. You'll find it on the CD down in e:\projects\nightvision\nightvision.mpg. When you're done, come on back.

Project Structure

This project has many parts to it, but like so many things that seem complex: it's nothing more than just a bunch of simple things stitched together. Properly broken down, it can be easy to grasp. Let's take a look.

I've discussed using intermediate renders as a way to get around various limitations within VideoStudio. They are also used extensively in this project, but not for these same reasons. You see, there are times when you'll hit a limitation of your own brain before you hit a limitation of the software. It's frequently easier to break things down into digestible pieces first, rather than succumb to the total weight of the project.



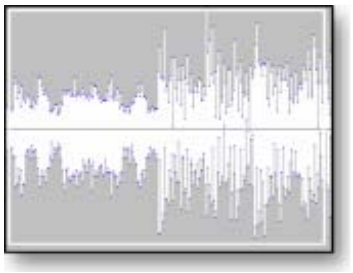
There are four main visual sections to the video: the intro, on the road, in the city, and the credits. I've classified audio as a separate section for reasons we'll discuss in a bit. Working on the project from the bottom up was actually a much easier task than doing it the other way around. I knew what parts and pieces I wanted in the final project, but was unsure what the final project was actually going to be. Funny how it all comes together sometimes.

In order to demonstrate the capabilities of VideoStudio, one of my constraints was that it had to be 100% VideoStudio. Once I started, though, I found I had the urge to cheat in a couple places. What these places were and how I overcame my moral dilemma, is explained here.

- **Audio Cues**. During the intro, you will see visible pulses synchronized with the beat of the synthesized kick drum. When I was first formulating the idea for this intro, I never intended to do this. The plan was for random pulsating brightness changes. But during an early test render, I noticed that a couple of these just happened to coincide with the drum and that gave me the idea to try and synchronize all of them. Exactly how, I will explain later. **Justification**: this effect didn't make or break the video. My original random pulses would have worked just fine and no one would have been the wiser. I also realized that this was an excellent opportunity to talk about how real life works.

- **Metal Frame**. During the city segment, the video is surrounded by an industrial-looking metallic frame. I created and rendered this in trueSpace and exported it as an image sequence: admittedly, something many VideoStudio novices probably won't be doing. **Justification**: an extremely similar effect can be done using Ulead COOL3D and the .C3D file can be used directly on the timeline. So, like the audio stuff above, this really didn't make or break the video. Using an exported image sequence was a good exercise for exploring alternative methods of doing overlays. **Is that really your justification**? Sure, why not? **Because it sounds like you just couldn't do it in COOL3D**. Okay, you got me. After I did the trueSpace version, I tried to recreate in COOL3D and just couldn't do it. I'm worthless and weak! I'll go back and work on it some more—I promise!

Audio Synchronization



Before we break apart the project into small, discussable pieces, I want to first talk about synchronizing audio and visual elements. You may not have consciously noticed, but throughout the entire project, all the cuts and transitions are timed with the rhythm and beats of the music track. Without this sync, you're sort of left with an uncomfortable feeling that something isn't right. The images feel more disjointed. It's as if the project were just *assembled* rather than *designed*. We don't want that. This has to be solid.

Further, this synchronizing is not limited to just cuts between clips. Often times events *within* a given clip must also be synched with the music. I've demonstrated this with the traffic light. Go back and watch the video, and notice how the light changes from red to green in time with the music. This wasn't by accident.

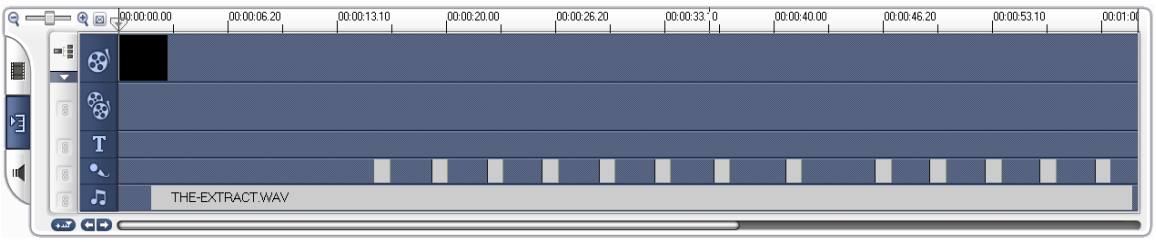
Approach

This is another area where Ulead did not expect us to wander. If they had, they certainly would have given us the ability to create cues in our clips and on our timeline. The lack of this feature indicates they just want us to drag and drop clips to the storyboard and leave it at that.

But like always, we won't let that stop us! As you know, with ripple editing turned off, clips on the overlay, title, and audio tracks will not move when changes are made to the main video track. This behavior can be annoying at times—like having to re-position twenty title clips after changing the length of your leader from 2 to 3 seconds—but it also has its benefits. You see, since they don't move, we can use image and audio clips to serve as cues.

How To Cue

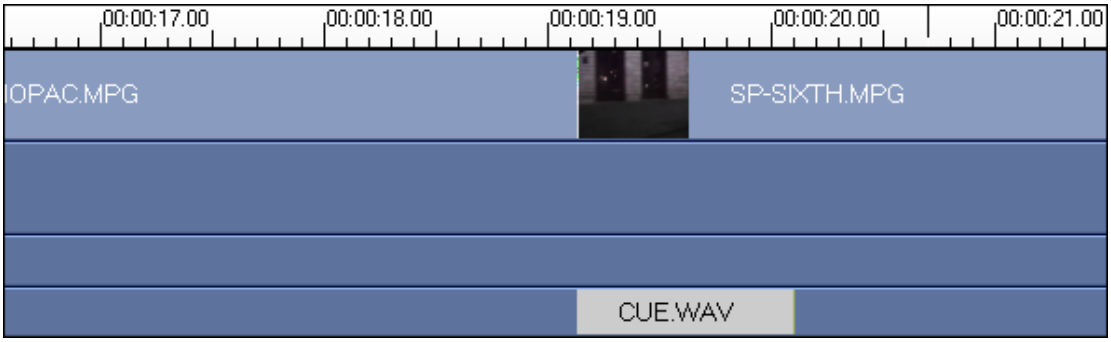
If you look at the nightvision.vsp project, in the voiceover audio track, you will see clips called cue.wav at various intervals. When I first laid in the audio track, I wanted to identify points in the music where I wanted an event or transition to happen. To do so, I listened to the song using the preview window, with my mouse hovering over the stop button. While the music played, I nodded my head to the rhythm and tapped my foot to the beat. 1, 2, 3, 4… 2, 2, 3, 4… 3, 2, 3, 4… 4, 2, 3, 4… I was doing nothing more than jamming along with the tune. As soon as I got to a spot where I wanted to mark a change, I'd hit the stop key *in time with the music.* At that point I'd write down the timecode, then place the audio cue.wav file down on the voiceover track exactly two seconds past that spot. (This is to offset our 2-second color clip leader.) This clip would then act like a bookmark: staying put no matter what else I did on the timeline. ***Example***. Once the music kicks in after the intro, I let it go eight beats before doing my first cut. While listening to the clip, I found this took place at 00:00:17.04. Given the 2-second leader, I placed cue.wav on the timeline at 00:00:19.04. With that one out of the way, I moved on to the next. This was repeated for each spot in the music where something needed to happen visually. Here's what the timeline looked like when complete:



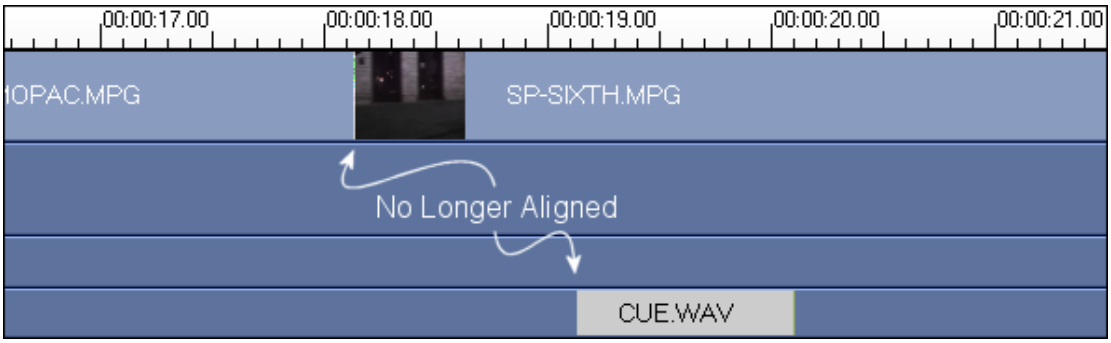
As you can see, this is another example of how the "Audio" step is not the next-to-the-last step. It's not an afterthought or a post script to the project. It's the anchor. Everything else locks to it. Once we know *when* something's going to happen *and for how long*, all subsequent editing decisions become much easier.

An Example

Using the above cue (00:00:19.04) I decided to make my first cut from the mopac.mpg clip to the sixth.mpg clip. Since VideoStudio automatically snaps edits to various timeline events, these frame-accurate trims were easy with the cue.wav file already there:



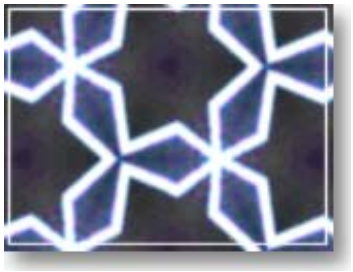
However, since any changes made to the video track change the relative positions of video track clips, I ran into a "de-synchronization" problem after dropping a cross-fade transition between the leader and the intro. Even though that's a different part of the timeline, it affects us:



You have to fix this—there's no question about it. But what about ripple editing? Well, for the same reasons we discussed on page 247, ripple editing cannot help us. We don't *want* the cue to move. It is a fixed point in time. The video tracks must sync to the cue, not the other way around. Therefore, if you know you've upset the timeline by one-second to the left, you'll have to make a corresponding edit somewhere else to move it back to the right. You *could* do this without the use of cues, but cues provide a solid method of checking your work. Otherwise it's too easy to shift something 30 frames one way and only 27 the other. Do this several times during a complex editing session, and you'll end up hopelessly lost.

Intro Clip

Let's start at the very beginning—a very good place to start. My idea was to kick off the video with something essentially unrecognizable as "video." The purpose was to set an expectation for the rest of the video—planting an idea that this is definitely going to be something out of the ordinary. It needed to both fit in and stand out.



Project
nightvision\sp-intro.vsp[10]

Output
nightvision\sp-intro.mpg

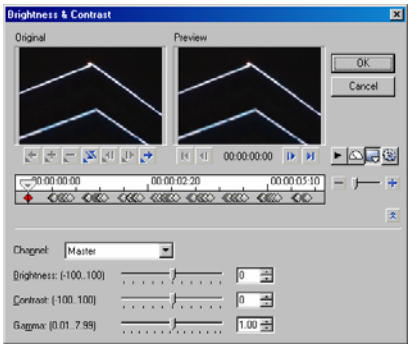
Approach
The strange blue lines in this segment are nothing more than neon lights on a building. You can look at the raw footage in clip uvs040202-001-12.mpg. That's the exact clip used as the background for this section.

As you might have guessed by now, the kaleidoscope video filter is the overriding visual effect for this clip. But that's far from the only thing going on. Here's what's happening:

- Clip playback speed set to 10% to give us the "choppy" motion effect.
- The "Light" video filter applied to enhance the illumination of the frame.
- The "Average" video filter to soften the overall picture.
- As mentioned, the "Kaleidoscope" effect to throw off our sense of what's real.
- The "Brightness & Contrast" filter with *lots* of keyframes for the pulsating flashes.
- Precise music synchronization.

Here's what's not happening:

- No "computer generated" imagery, per se.
- No special animation. Just real video and a few creatively applied filters.

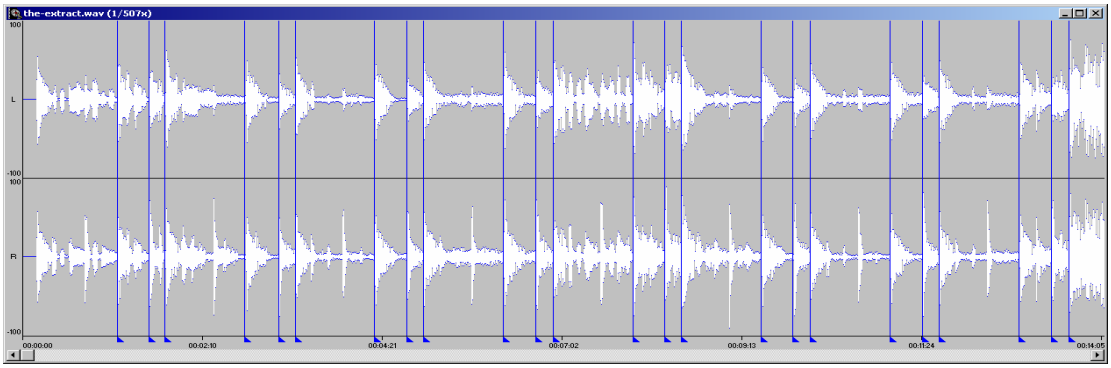


Those pulsating brightness levels were far and away the most complicated part of this clip. Just to give you an idea, take a look at the video filter options dialog box.

There are sixty-five keyframes on this six-second clip. I don't do that kind of work every day, but I thought it might be cool just this time. Creating keyframes is no big deal. Just click that little "add

[10] The abbreviation "sp" stands for "sub-project." When you are working on a project with many files, grouping them with similar names can be most helpful. Especially for complex, multi-layered projects like this. For me, I left the original source files named "uvs" which helped me keep track of the files I hadn't touched. The uvs files fed into the "sub-projects," which generated the intermediate sp files. The sp files then made up the final project.

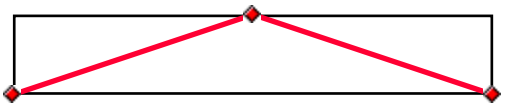
keyframe" button all you want. But setting exactly *where* each keyframe goes is another story. The precise location of the keyframes is exactly why went for outside help. Using the Audio Editor tool from MediaStudio Pro, I opened the file and examined the waveform.



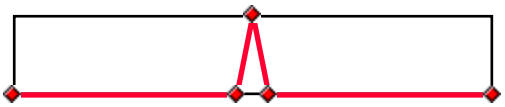
Looking at (and listening to) the waveform, I marked cues at every kick drum beat. These are the vertical lines in the picture. With the cues set, I wrote down the timecode for each one as follows: 00:00:01.07, 00:00:01.19, 00:00:01.26, 00:00:02.27, 00:00:03.10, 00:00:03.16, and so on.

Writing the locations of all groups of three beats on a separate sheet of paper, I then went back to VideoStudio, dropped the Brightness & Contrast filter on the clip, and clicked "Customize Filter."

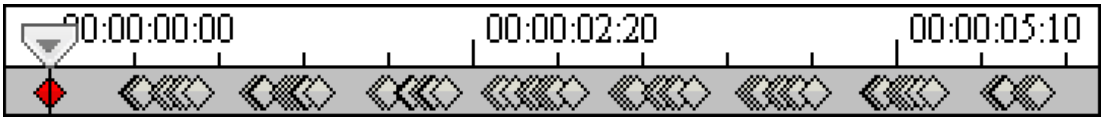
You might be wondering why there are 65 keyframes when there are only 24 beats. The answer's easy if you remember how keyframes work. Recall that VideoStudio will calculate every frame between two keyframes to figure out what the filter should look like at each given point. If my brightness level is 0 at the first keyframe, then 80 at the second, I don't want a gradual increase in brightness like this:



I want them to come in "bursts" which means I have to keep the level low for most of the clip. The only way to do this is with "helper" keyframes on either side of the main keyframe.



Therefore, each group of three beats ends up getting nine keyframes. I have seven full groups, so that's 7 x 9 = 63 keyframes. The eighth and final group did not get this treatment. That just leaves the standard start and end marks, bringing the total to 65:



When you set a new keyframe, it's initialized to whatever the current tweening values are. You may have inferred from the illustrations above there are far more keyframes where brightness = 0, than otherwise. For these reasons, it's best to set the default start and end keyframes to the 0-level first, then add all the intermediate keyframes. Only after that should you go back and set the "peaks."

It may seem like a lot of work—and it is—but once you understand what's really going on, handling it becomes much less of a chore. Just like anything, it takes practice. I recommend practicing. Grab a clip, a filter, and try this out. Don't worry about any synchronization: just get some keyframing experience. Make mistakes. Learn!

Drive Time

There are only two source clips in this section of the video. However, neither is used directly in the video itself. Like the intro, they serve only as inputs to intermediate renderings.

Projects

nightvision\sp-mopac.vsp
nightvision\sp-sixth.vsp

Output

nightvision\sp-mopac.mpg
nightvision\sp-sixth.mpg

Approach

Just to allay any fears, I was not driving 400 mph to shoot this footage. If you watch the source clip (uvs040202-001-02.mpg), you will see a car moving at very safe and legal speeds. ☺ Listen closely and you may even hear my turn signal.

Obviously, the clip was speeded up. When the camera is facing forward, setting the clip speed to 800% generated this frenetic look. When facing out the side, a more comfortable 300% is used. The clips were placed in individual "sp" projects, sped up, and rendered to intermediate files.

This same technique was applied to the sp-first.mpg and sp-fifth.mpg clips used later on in the project. (By the way, *first*, *fifth*, and *sixth* are the street names where the footage was shot, that's all. There's nothing inherently first about *first*.)

Title Text

Immediately after the intro, I wanted to work in the title of the video using VideoStudio's built-in title clips. True to form, I also wanted to make it look a little like something VideoStudio's built-in title clips weren't normally capable of.

Project
nightvision\nightvision.vsp (00:00:15.19)

Output
No intermediate output. Clips appear directly in the final project.

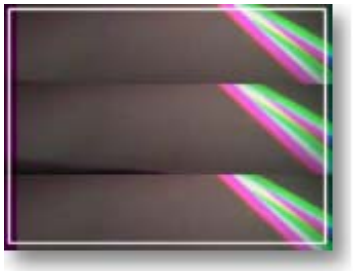
Approach
The title text is actually made up of two title clips run back to back to look like a single clip. This is the technique discussed back on page 188: one clip fades in a letter at a time then holds until the end. The second clip begins where the first left off, but zooms from 100% down to 0%. The effect is seamless. The fact that the background changes at the same time isn't a requirement for this effect. That is, in this particular instance, I changed both the title animation and the background clips at the same time just to synchronize with the music. But the titles and the background clips are two separate tracks and you can edit them independently.

end of one title clip


beginning of next

Triple Split Screen

On page 171 we learned how to do split screens. This three-way split screen is no different. We've simply run through the procedure multiple times to generate this particular effect. Also, we've split the screen with the clip itself. There's no law you have to use two different clips when you do a split screen!

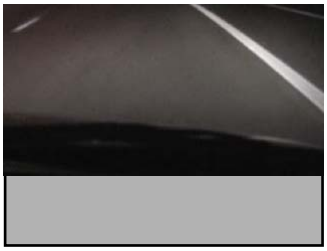


Project
nightvision\sp-triple*.vsp

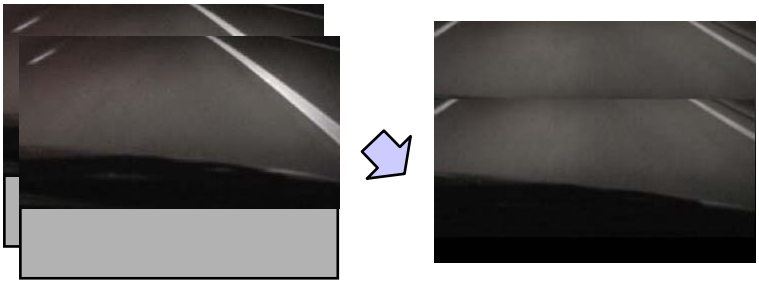
Output
nightvision\sp-triple*.mpg

Approach
The idea is to put the clip in an overlay track and position it so the center of the original image to the top of the frame. The space left unoccupied by the video frame is filled with the project's default background color, which I've shown in gray. (If I'd used black, it would be too difficult to discern where the clip ends and the background begins.)


original


sp-triple1.mpg

Once you've rendered sp-triple1.mpg, you then start a second project and overlay the clip with itself. That is, you'll place it on both the video track *and* the overlay track. Position the frame as before, but this time move the overlay down.



sp-triple1.mpg
once in video track
and once in overlay track

sp-triple2.mpg

Now repeat the process for yet a third overlay. You don't really need to keep the intermediate projects. I've left them on the CD just for educational purposes. However, if you ever want to go back and recreate one of the intermediate files, you'll definitely want to have these around to look at. So I'd leave them.

We're almost done. You may have noticed oddly colored road lines on the clip in the final video. These come from the "Color Shift" filter. Using this filter, you can not only split the image into its red, green, and blue components, but also spatially reposition these components anywhere in the frame. Check out the final nightvision.vsp project itself to look at the video filter settings.

Quick Flash

It only lasts for nine frames. That combined with the fact that the video's moving at a frenetic pace at this point, you may not have noticed this brief sequence. Short as it is, it's got a little punch to it.

Project
nightvision\sp-flash.vsp

Output
nightvision\sp-flash.mpg

Approach
At the very end of the road race are these frames:

This filmstrip shows the last nine frames of the sp-mopac.mpg clip. They were separated from the main clip with the split tool. After separation, two video filters were applied to them: the Zoom filter and the Brightness & Contrast filter. The Zoom effect adds little "trails" to the edges of objects and makes it look like the blur you'd see pulling the camera back (or forward) very quickly. The Brightness & Contrast filter—another one of my favorites—was used for the washed-out look at the end.

The last frame of this sequence was saved as a still image and appended to the end of this sequence for 15 frames. This is the exact freeze frame technique described on page 179. The frozen frame crossfades into the next clip, which just so happens to be where the "road" segment ends and the "city" segment begins.

Watching the final version of this project, I sometimes wish this particular sequence wasn't so short. I really like the way it looks. Maybe I'll try it out again in a future project. You never know!

Picture in Picture

I used two different frame effects for the downtown segment of the video. The first is a simple picture in picture technique. Just as we did with the three-way split screen effect above, this "pic-in-pic" effect uses the same clip for both parts.

Project

nightvision\nightvision.vsp (00:00:29.10)

Output

No intermediate output. Effect generated directly in the final project.

Approach

The inset is a copy of sp-sixth.mpg placed on the overlay track. It's been resized to about 75% of the screen and 20% transparency. No position or path applied. The background is the same clip. It begins with a 15-second crossfade from the previous freeze-frame. It has the Monochrome and the Brightness & Contrast filters applied. Check out the project itself for specifics.

This is actually a rather versatile effect. By applying or combining different filters for the background clip, you can create a variety of moods. Make sure you take some time to try it out. You might stumble across something useful.

And if you do, let me know. Maybe it will go in the next book!

Metal Frames

At some point in the project, I knew I'd have to throw in something using alpha channels. If I weren't trying to do a "product show off" type of project, I probably wouldn't have done this. That is, it feels a bit gratuitous. But as it is, it's a good learning tool.

Project

nightvision\nightvision.vsp (00:00:32.22)

Image Sequence

nightvision\metalframe\metalframe.uis