MEDIA ALERT

NukeX: What and Why?

NukeX... What?

NukeX 6.0 has all the features of Nuke 6.0, including the brand new roto and paint tools and the Foundry's acclaimed keyer, Keylight, as standard. In addition, it has an integrated 3D camera tracker, depth generator, tools for automatic lens distortion correction and also includes FurnaceCore, the nucleus of the Foundry's Academy Award®-winning Furnace tool set.

NukeX... Why?

So why do we feel the need to offer NukeX alongside the conventional Nuke? Well, we've noticed that our customers are increasingly using Nuke's 3D capabilities to push the boundaries of what has traditionally been possible inside a compositor, as witnessed by many of the recent case studies on our website. The work of Image Engine on District 9, for example, is a case in point, with its efficient 3D workflow for clean plate generation and inventive use of point clouds for novel applications such as lighting and colour grading.

We're continually impressed by our customers' creativity and the ingenious tricks they manage to pull off inside a compositor which, though flexible, is after all still a compositor. We decided we wanted to do our bit to encourage such inventiveness and make it even easier for people to access and use 3D information and techniques inside Nuke... and so NukeX, with its integrated 3D camera tracker, was born.

Now, camera trackers are nothing new, but the advantage of having one inside Nuke is that it allows far greater interactivity and control over the tracking process. The other additional tools in NukeX are also targeted at increasing the artist's creative control during live-action compositing. The new LensDistortion node is a natural accompaniment to the camera tracker, there largely to facilitate the combination of computer-generated (CG) elements and characters with live-action footage. It provides three different automated analysis methods, so the artist doesn't have to spend time tweaking the parameters in order to remove distortion but can often undistort an input sequence with a single button press. Similarly, the FurnaceCore plug-ins automate common compositing tasks which are time-consuming to perform manually and which require little creative input – tasks such as wire and rig removal, retiming and noise and grain removal. NukeX also contains a new DepthGenerator plug-in... but we'll explain where that comes in later. For now, let's take a closer look at the CameraTracker and the new possibilities and workflows its integration inside NukeX opens up.

The CameraTracker... What?

The camera tracker can derive 3D scene information from an image sequence that was acquired with a moving camera. In order to do this it tracks a set of distinctive, fixed features through the sequence and looks at how their positions in the image change between frames. From this, it can work out not only the positions of those features in 3D space but also the path taken by the camera. The result is a point cloud showing the positions of the tracked features in 3D space, along with a Nuke Camera whose position is keyed appropriately for each frame of the sequence.

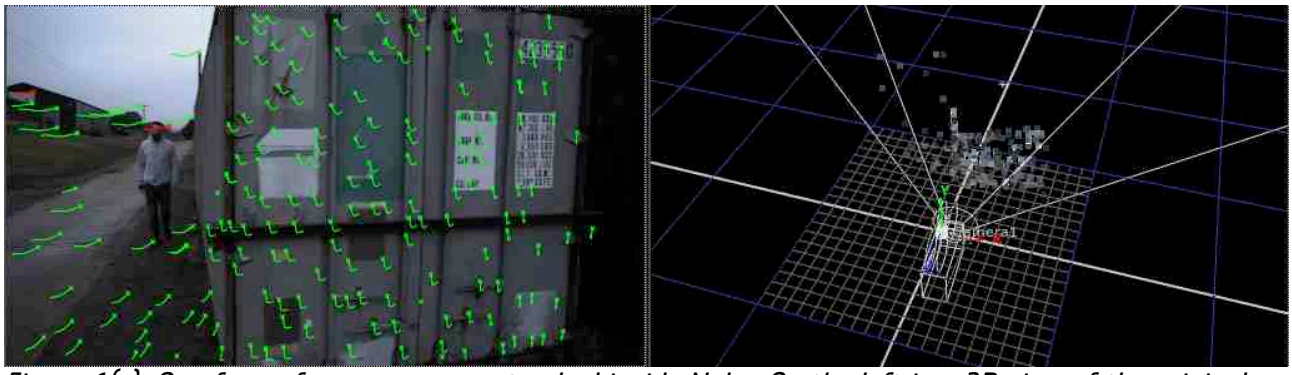

Figure 1(a): One frame from a sequence tracked inside Nuke. On the left is a 2D view of the original image with the tracked features overlaid. The right frame shows a 3D view of the point cloud derived from the tracked features, together with the position of the tracked camera.

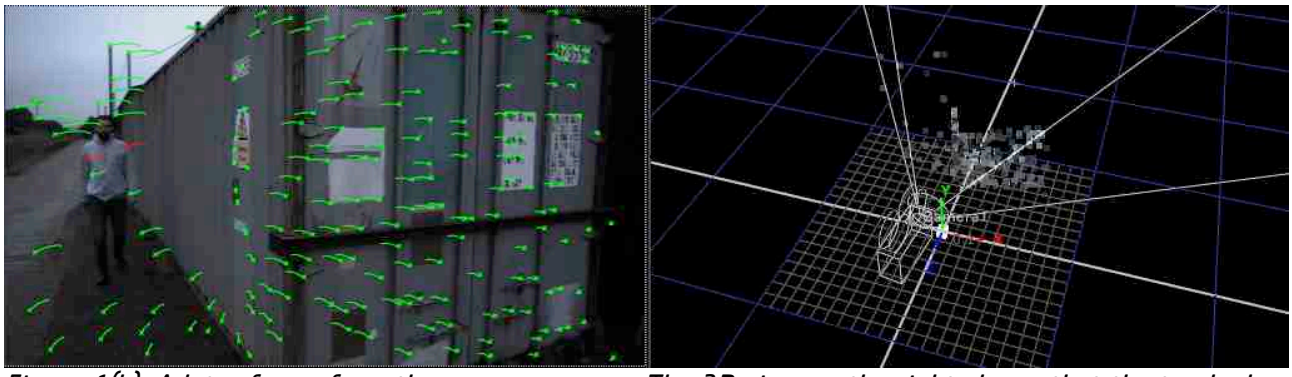

Figure 1(b): A later frame from the same sequence. The 3D view on the right shows that the tracked camera has moved further round the corner of the container.

Subsequently, the point cloud can be used to add new geometry into the sequence. A new object – a CG character from an .fbx file, for example – can be located at a particular point and will then appear to stick to that point as the camera's view of the scene changes. You can also orient the 3D scene by adding an axis (x, y or z) to a few points, or marking points that should lie on the ground to set a ground plane.

As well as locating geometry at a point, you can use the point cloud to add new images onto planar surfaces in the scene. Selecting a number of points which lie on the same surface will allow you to create a new Card node, oriented so as to line up with the selected points as closely as possible. An image applied to the Card will then appear to stick to your chosen surface as the camera moves around the scene.

The CameraTracker is also fully stereo-aware. It can take a stereo input stream and solve for both cameras simultaneously, outputting a point cloud for the scene together with the left and right camera paths.

Figure 2(a): Applying a Nuke logo to the container using a Card node. The tracked features on the front wall of the container were selected and a Card created which lies on the same plane. An image containing the Nuke logo was then applied to the Card and merged with the original view.

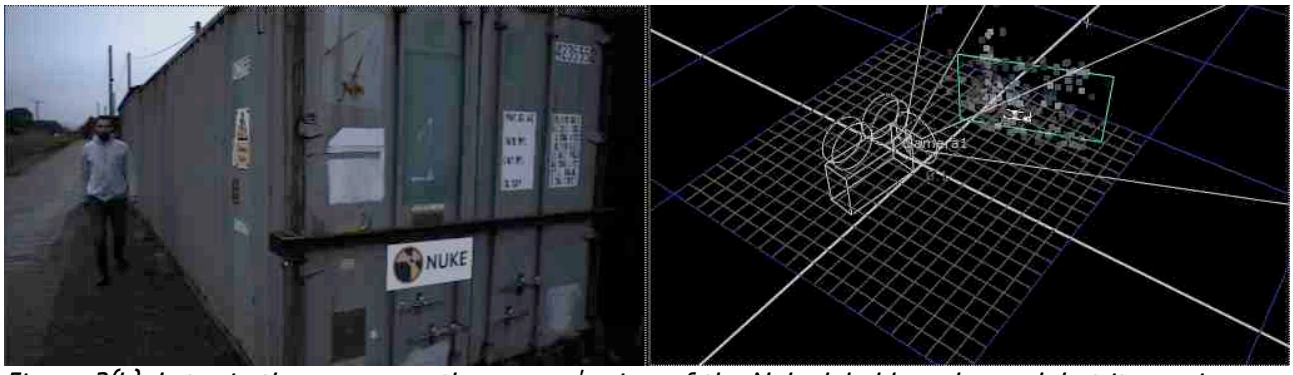

Figure 2(b): Later in the sequence, the camera's view of the Nuke label has changed, but it remains

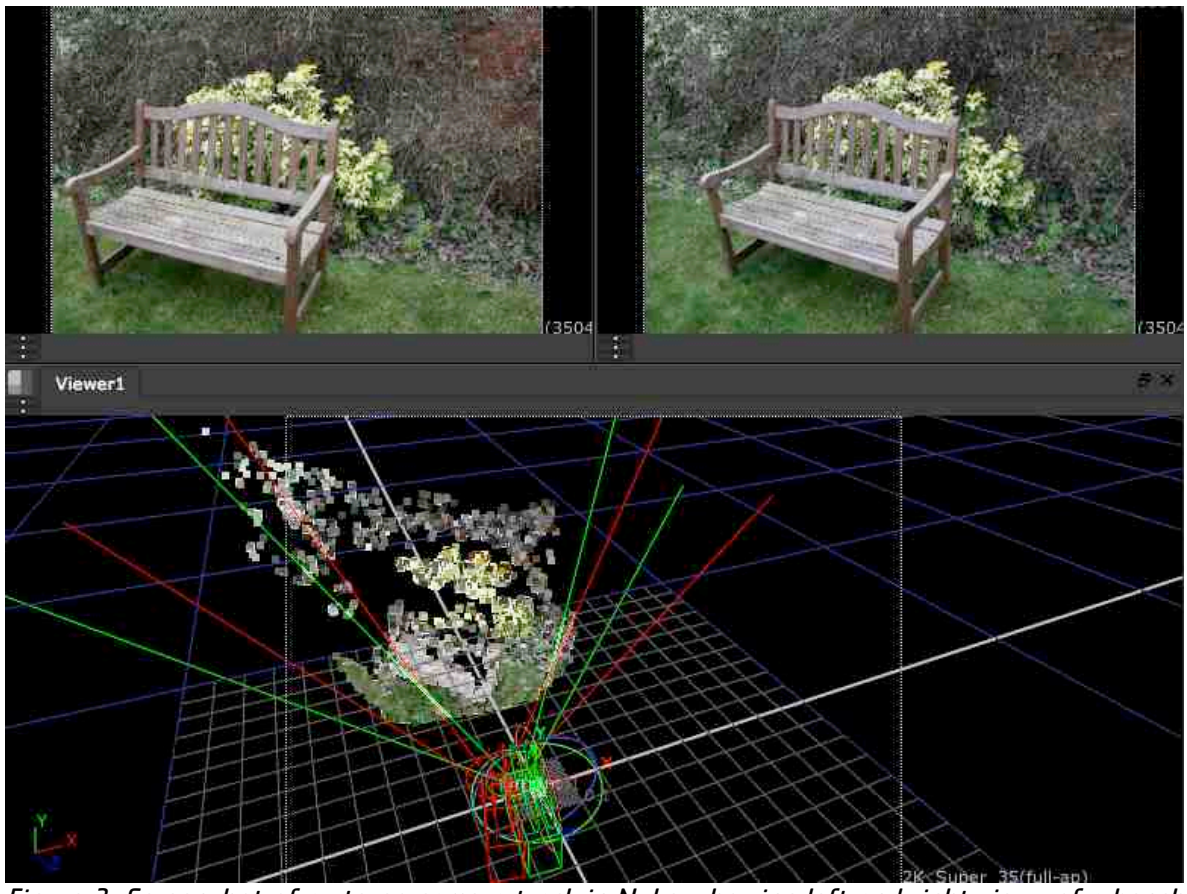

Figure 3: Screenshot of a stereo camera track in Nuke, showing left and right views of a bench

All this is interesting, but in fact the applications of the CameraTracker go far beyond point clouds. Having easy access to the camera geometry can dramatically increase the efficiency of core compositing tasks such as rotoscoping and clean plate generation. For example, Figure 4 shows an example of a shot in need of repair (left) together with a clean plate (right). In this shot, the camera starts off further into the room and runs backwards along the dolly tracks visible in the left image. We wish to cover up these dolly tracks using a patch from the floor in the clean plate on the right.

Figure 4: Original shot in need of repair (left) and a corresponding clean plate (right).

Now, the normal 2D workflow in this scenario would require a lot of time-consuming manual work: frame-to-frame adjustment of the roto'ed region and a variety of 2D transformations to match-move the clean plate as the camera's perspective changes. Once we have tracked the scene in 3D, however, we can use a much more efficient workflow. The desired region can simply be cut out once, then projected onto the floor of the 3D scene and rendered through the tracked camera, as shown in Figure 7. Rendering through a 3D camera in this way means that Nuke will automatically take care of matching the patch to the motion and perspective changes in the sequence, as shown in Figures 5 and 6.

Figure 5: Original frame (left) and repaired frame (right), showing a rotoshape around the area extracted from the clean plate.

Figure 6: An earlier frame of the sequence (left) and repaired frame (right). The extracted floor area from the clean plate was projected onto the floor of the 3D scene, through the camera corresponding to the matching frame of the original sequence. Here, the repair was generated by viewing the projected floor area through the tracked camera corresponding to the current frame. Note that even though the camera's perspective has changed, the repair region is still lined up perfectly with the original floor.

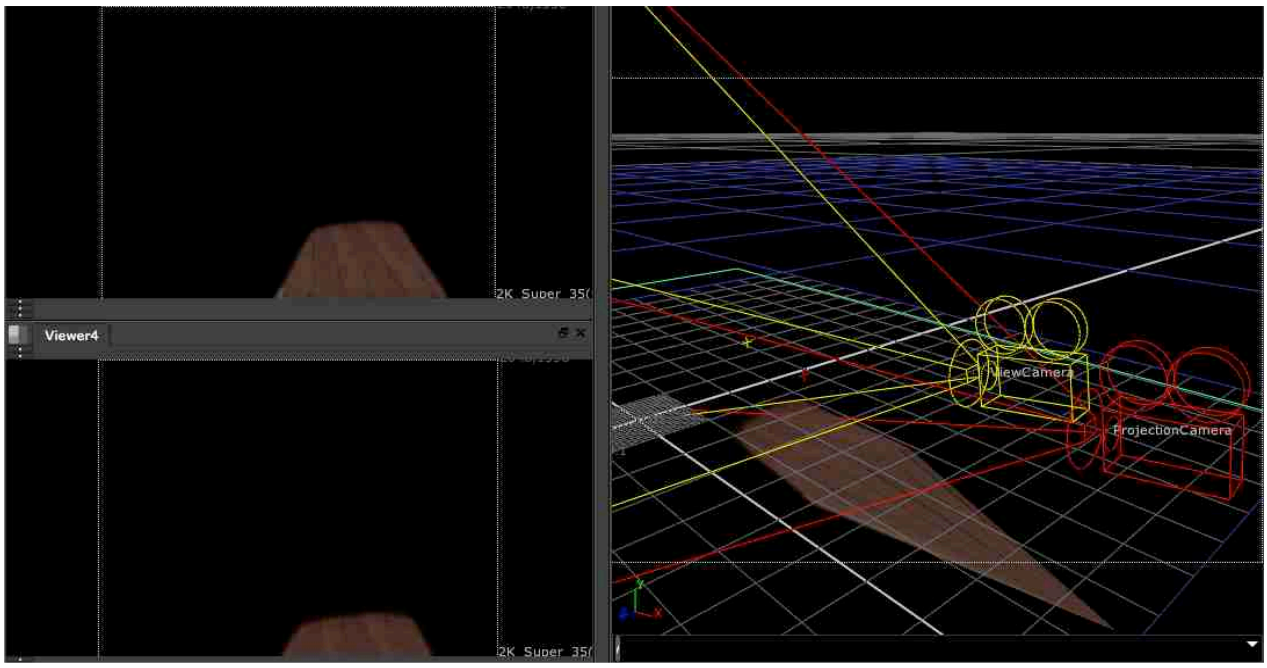

Figure 7: The 3D set-up for frame repair. A section of floor from the clean plate is projected onto a Card positioned on the ground plane using the ProjectionCamera, shown in red, which corresponds to the tracked camera at the frame of the sequence for which the clean plate was created. The ProjectionCamera's view of the floor on the Card is shown in the top left panel. The floor area is then viewed through the ViewCamera, shown in yellow, to create the necessary repair at an earlier frame of the sequence. This view is shown in the bottom left panel.

Figure 8 shows another application of this technique: replacing the green screen seen through the windows in this example with a more realistic outlook. The background plate has simply been texture-mapped onto a Card, which was then scaled and rotated so as to have the desired appearance when seen through the camera corresponding to one frame of the sequence (in fact, the same frame we used for the clean plate earlier). To reproduce the correct appearance of this outdoor scene as the camera's perspective shifts, all we then need to do is view this Card through the tracked camera at each frame, then composite the original

frames back over the top.

Figure 8: Green screen keyed out and replaced with a more realistic background (right). The background plate has been positioned on a Card, so that when the camera's perspective changes (left) the view of the background scene will change accordingly.

The CameraTracker... Why?

The benefits of a fully-integrated 3D camera tracker are manifold. Compositing is increasingly becoming a 3D process, and within Nuke there is already a massively competent 3D compositing environment. Having easy access to the 3D scene information provided by the camera tracker in NukeX allows you to take full advantage of these powerful 3D capabilities.

Traditionally, 3D trackers have either not been available to artists during the composite – the tracking process being handled by a separate match-moving department – or have been available only as stand-alone applications. In either of these scenarios, the need to obtain a 3D track for a particular shot – or just one element within a shot – can introduce a bottleneck into the compositing process. Even if you don't have to send the project off to another team to be tracked, there is still the overhead of needing to do any necessary pre-processing inside another application, then having to export the results after tracking and import them back into the compositor. Clearly, whenever you need to obtain 3D tracks for a shot, having a Nuke node that can provide them is going to make the process significantly more efficient. It also makes it possible to use 3D tracks in situations where it would previously have seemed too time- or labour-intensive to be practical.

Obviously, during the development process every effort has been made to ensure that the CameraTracker will be able to deal with many typical shots with a minimum of user intervention required. However, as with any tracker, there will always be some shots where it needs a little help to achieve the desired result. This help might be in the form of pre-processing - applying a sharpening filter to the input in order to increase the visibility of trackable features, for example. Similarly, it might be necessary to mask out regions where there is significant motion not due to the camera – for instance, either a person walking through the foreground of the shot or a large area of sky in the background with moving cloud could act to obscure the camera motion in the scene. Now, while some 3D trackers provide tools for this type of image-processing work, they are essentially tracking applications and as such will never be able to offer the powerful, complete toolset of a fully-functioning compositor... unlike NukeX. The tools inside NukeX also have the advantage of being Nuke nodes, so anyone familiar with Nuke will have no need to learn new workflows, shortcuts or key presses in order to use them efficiently - they're just the same tools they already use for a variety of other compositing tasks every day.

The same applies to the CameraTracker itself: it's a Nuke node, with Nuke-style controls, and so will be much easier for Nuke users to get to grips with and start using than a separate tracking application might be. Instead of a stand-alone program with its own ways of doing things – and perhaps its own preferred data formats – it's just another Nuke node that behaves as you would expect a Nuke node to behave, thereby reducing the otherwise steep learning curve for an artist who might not have used a 3D tracker before.

Of course, as a Nuke node the CameraTracker also has access to the standard Nuke parameter infrastructure: the expression and parameter linking mechanisms which can be so crucial to ensuring a smooth workflow. Its controls can be accessed via Python scripting, as can both the generated point cloud positions and the camera trajectory in the curve editor. Naturally, any changes to its controls are also saved to the current project file, making collaboration between artists much more straightforward than if a separate application had been used for tracking. Equally, having the CameraTracker (as well as other NukeX nodes) incorporated in the script ensures that the workflow remains smooth and efficient when changes are required: for example, when footage is switched or CG assets are versioned.

In addition, having a 3D tracker embedded in NukeX allows the tracking process to be a much more interactive part of the composite than it would be in a stand-alone application. After running the tracker for the first time, if the results turn out to be not quite what you need, it's very easy to go back up the tree and tweak the pre-processing, or perhaps add an extra roto around an object you've decided you don't want to track. Then you can simply start the tracker again; once it's done, you will immediately be able to see whether the new results work well in your composite, with no exporting or importing of images and data required.

Perhaps more valuably, though, this workflow means that there is actually no need to delay work further down the tree until you have perfect tracks to work with – once it's clear that the tracks are going to be useful, any necessary tweaking can be done at a later date. The tracking could even be refined by someone else while the original artist concentrates on getting the look of the shot right. When the tracks are updated later, any new elements and adjustments further down the tree can be picked up as a natural part of the workflow.

For the future, having an integrated 3D camera tracker in NukeX opens up all sorts of possibilities for us to provide new tools which extend NukeX's 3D capabilities even further. The tracker gives us easy access to the 3D geometry of the scene, and once this geometry is available we can use it to do some clever things. For example, we are currently working on an image-based modeller, which will use the known scene geometry to generate partial geometry for projections. Also, it's no secret that some improvements to Nuke's lighting capabilities are in the works, as we start to integrate technology from Katana. Known geometry will allow a re-lit scene to have realistic shadows.

The DepthGenerator node in NukeX is the first example of a new tool we have only been able to provide since having easy access to the tracking data. (We can promise you it won't be the last.)

DepthGenerator... What?

The DepthGenerator takes the output from the CameraTracker, together with the tracked image sequence, and uses them to create a depth map for each frame of the sequence. Like Furnace's F_Depth plug-in, it does this by looking at how the scene changes between the current frame and those before and after. However, knowing the camera position in relation to the scene at each frame allows it to obtain a much more accurate picture of the depth than was previously possible. In addition, if the camera has a known focal length and film back – or if good estimates for these have been obtained by the CameraTracker – the depth can even be presented in meaningful units – we can say that a particular object is a certain number of metres in front of the camera, for example. Without the camera information it would only ever be possible to get a rough estimate of the relative depths of things in the scene – so we might be able to tell that a person was standing in front of a tree, but not how far in front of the tree they were.

Figure 9: Example depth map of a static scene from the new DepthGenerator.

DepthGenerator... Why?

Having an accurate picture of the depth of a scene – together with the camera geometry – in NukeX's 3D environment opens up new and exciting workflows which were not previously possible inside Nuke. For example, by making use of some of Nuke's existing 3D tools, we can use the depth information to create a partial 3D model of a static scene.

In order to do this, we turn again to the output from the CameraTracker. With a known camera path and camera geometry, a tracked sequence can be displayed on a Card node facing the camera, and positioned so as to replicate the camera's view of the scene. The depth map can then be used, via a DisplaceGeo node, to distort the image on the card so as to create real depth in the scene.

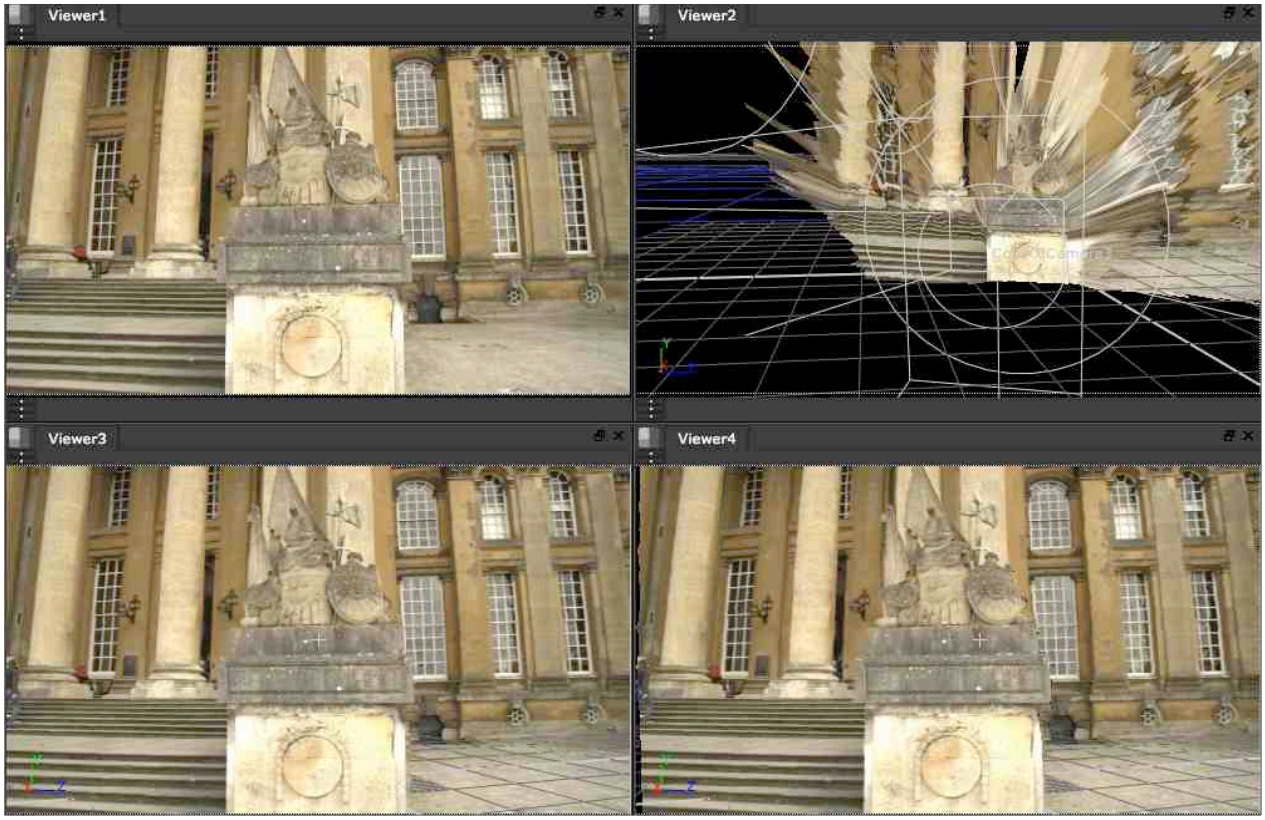

Figure 10: NukeX screen shot demonstrating the use of the DepthGenerator to create a novel view of a scene. Viewer 1 is displaying a view of Blenheim Palace from a pre-tracked image sequence. To create the 3D view in Viewer 2, this image has been projected onto a Card and then displaced according to the depth map output by the DepthGenerator. Viewer 3 shows the original view, recreated in Nuke by viewing this displaced geometry from the original camera position. In Viewer 4, the viewing camera has been displaced slightly from the original position, to approximately recreate the scene as it would have appeared from this new camera position.

Obviously, this distorted image is not a real 3D model: there is information missing everywhere except in the parts of the scene that were visible to the original camera. Move the camera in NukeX too far to one side, and the missing areas will become obvious. However, change the camera position only slightly and the missing data is not significant, or even noticeable. Instead, the depth-displaced image means that as the camera's view shifts, the slight parallax differences that would result if the camera were looking at a real 3D scene are reproduced. In other words, the depth information – in conjunction with Nuke's existing 3D toolset - allows you to create novel views of the scene. Despite this only holding true for relatively small shifts from the original camera position, it can be surprisingly useful. For example, in recent trade show demos we have shown how this workflow can be used to stabilise a wobbling camera or even – more excitingly – to create stereo views from a single, live-action image stream.

Figure 11: Anaglyph view of the Blenheim Palace shot. Novel left and right views of the scene were created using the method shown in Figure 5.

Of course, there are many other potential uses for the DepthGenerator besides creating new views. For example, it could be used to create depth of field effects, or to apply a depth-based colour correction. We could also imagine it being used for creative relighting, by shining a light on a distorted mesh, like the one used in the example shown above. In truth, however, we don't yet know the full extent of where this depth information might be useful... we just know that it will be. In fact, we're keen to find out just what creative uses it will be put to once it's out there.

Lens Distortion tools... What?

For a while now, many of our customers have been asking for better lens distortion tools in Nuke, so they no longer have to rely on stand-alone tools for the removal or application of lens distortion. As with the camera tracker, not having to continually export and re-import sequences from the current composite in order to use these tools can have considerable benefits for artists' efficiency and productivity. As a result, in Nuke 6.0 we are introducing a new LensDistortion node with a much-improved lens model. This new model allows the distortion centre to vary and can also represent the asymmetric distortion patterns common to anamorphic lenses. NukeX takes this a step further by offering three automatic distortion analysis methods: Grid Analysis, Image Analysis and Line Analysis.

The Grid Analysis method will detect the lines or corners of a calibration grid and use their deviation from the expected regular pattern to determine the lens distortion present. If you have a calibration grid that corresponds to your footage – in other words, if it was shot with the same camera, lens and focal length – this method will provide the most accurate measure of the distortion.

If you don't have a grid available, the other two methods can also be useful. The first, Image Analysis, will estimate lens distortion "blind" in a sequence obtained with a moving camera. Like the camera tracker, it tracks recognisable features through the sequence. Lens distortion affects different parts of the image differently – its effect usually becomes more marked as the distance from the centre of the image increases – so will have an effect on the trajectory of a feature as it moves across different parts of the frame. The algorithm for calculating the distortion works by finding the set of lens distortion parameters that, when applied to the input sequence, correct the feature trajectories so that they match as closely as possible the paths the features would follow if there were no distortion present – that is, if they were affected only by the motion of the camera.

Finally, the Line Analysis method will estimate the distortion present from a set of hand-drawn lines. It requires the user to mark points along lines in the image that should be straight, then attempts to undistort the image so that all these points lie on straight lines. Of course, this is the most labour-intensive of the three methods and its performance will be highly dependent on the data provided by the user. However, unlike the others it can be used to calculate the distortion from a single, still frame and can provide a useful fall-back solution in the unlikely event that the other two methods fail.

Lens Distortion tools... Why?

There are two main reasons for making it easier to apply and remove lens distortion in NukeX. Firstly, decent lens distortion tools are needed to support the CameraTracker. Although this can cope with a certain amount of lens distortion, there will be some sequences with severe distortion where it is necessary to correct for this before tracking in order to obtain good tracks. The automatic analysis methods have been designed to make the workflow as smooth and effortless as possible in this situation. With a calibration grid or image sequence available, in many cases a single button press should suffice for calculating the distortion present, though the distortion parameters can of course be tweaked by hand if necessary. The input sequence can then be undistorted prior to tracking using the LensDistortion node, or alternatively the distortion parameters can simply be fed into the CameraTracker, which will then compensate for the distortion internally. The CameraTracker also calculates lens distortion and will refine these values if it is able to obtain a more accurate estimate of the distortion during tracking.

Secondly, the ability to apply and remove lens distortion accurately is essential when combining CG elements with live-action footage. Usually, any distortion will need to be removed from the footage before the new elements are tracked into place, then the combined scene is typically re-distorted in order to match the original camera's view.

Again, there is an advantage to being able to do all this inside a single application. Every time you resample an image in order to distort (or undistort) it, a small amount of blurring is unavoidably introduced. If you have to move to another application in order to apply or remove distortion, rendering the images out before and after, this blurring will occur every time. Inside NukeX, however, there's no need to render the images in between. Both LensDistortion and the CameraTracker calculate values for the distortion model used by Nuke's Card node, which can then be used to apply the same distortion directly to an image as it is texture-mapped onto a Card. For example, if you were setting up a panorama in NukeX, you could use this workflow to ensure that images displayed on adjacent Cards were free from distortion and so would line up properly in 3D space. In this case, the distortion model would be applied during the texture look-up, ensuring that the process of rendering the undistorted panorama would subject the original images to only a single filter hit.

FurnaceCore... What?

FurnaceCore includes twelve plug-ins from our award-winning Furnace tool set. In common with the rest of Furnace, these plug-ins are designed to automate common compositing tasks, or at least make them considerably faster and more efficient. In addition, the FurnaceCore plug-ins have had their user interfaces updated to be more intuitive and "Nuke-like".

The twelve plug-ins selected to be part of FurnaceCore – and therefore NukeX – are:

- ***F_Align*** **-** This plug-in will automatically align two sequences that were acquired at different times, without requiring the user to set tracking points. It's a powerful tool and can be used for both stabilisation and match-moving.
- ***F_Steadiness*** – Another tool that can be used for stabilisation, F_Steadiness automatically tracks and removes camera shake, again without requiring the user to select points for tracking. Its algorithm is capable of removing high-frequency camera wobble without affecting the underlying motion of the camera, and can also be used to correct minor shifts in perspective or scale.
- ***F_DeFlicker2*** - This tool will remove in-scene flicker, such as might be caused by stray light or poorly-synchronised light rigs. It can cope with motion in the scene and can even remove multiple overlapping flickers with different phases.
- ***F_DeGrain*** – Uses a wavelet-based algorithm to perform spatial and temporal grain reduction, based on an initial grain selection by the user. Its temporal component uses the inter-frame motion in a sequence to reduce softening of the images after the grain has been removed.
- ***F_ReGrain*** – Allows you to reapply grain to an image, for instance after adding a CG element to some previously de-grained live action footage. F_ReGrain contains some stock grain samples; alternatively it can sample the grain from one clip and generate similar grain to apply to another.
- ***F_DeNoise*** – This uses the Foundry's advanced motion estimation technology to reduce noise in a sequence without softening the images.
- ***F_Kronos*** – Another plug-in built on motion estimation technology, F_Kronos is a retimer that can be used to slow down a clip. By estimating the pixel motion it can generate new, sharp frames that sit in-between the original images and prolong the clip's duration. Alternatively, it can also be used to speed up a clip and will introduce realistic motion blur automatically. The algorithm behind F_Kronos has been carefully tuned to minimise the edge-dragging artefacts that typically occur between foreground and background objects when sequences are retimed.
- ***F_MotionBlur*** – An alternative way to apply the realistic motion blur generated by F_Kronos, without having to wade through the complex retiming interface in order to do so. F_MotionBlur has simple, user-friendly controls that provide a quick and efficient route to great results.

Figure 12: Image of a car before (left) and after (right) the application of F_MotionBlur.

- ***F_VectorGenerator*** – F_VectorGenerator performs motion estimation and outputs a vector field that describes the motion in a sequence. It uses the same motion estimation algorithm that F_DeGrain, F_Kronos and F_MotionBlur use to produce their results. However, having this technology in a separate plug-in allows the same motion field to be re-used in different places or rendered out to be used later. The vectors produced by F_VectorGenerator can be fed to any of the plug-ins that use them, which will then skip their internal motion estimation steps. Motion estimation is a computationally expensive process, so it can often be more efficient to use F_VectorGenerator to estimate motion in advance if you intend to use more than one tool that requires it.
- ***F_MatchGrade*** – Correct colour and luminance differences between clips automatically, for example to match two sequences acquired at different times of day. F_MatchGrade can automate the often time-consuming colour-grading process by modifying the colour histogram of an image to match a reference image. Its algorithm can compensate for even large colour differences between clips.

Figure 13: Two landscape shots (left) and the result of using F_MatchGrade to apply the colour distribution of one to the other (right).

- ***F_RigRemoval*** – This plug-in also uses motion estimation, this time to remove foreground objects from a clip and reconstruct the clean background. The user simply highlights the object for removal and the plug-in then searches forward and backward in the clip for the information it needs to fill in the missing background region. Provided there is sufficient motion of the foreground relative to the background, this can result in a seamless repair, as shown in Figure 14.

Figure 14: Result of using F_RigRemoval to remove a cat walking along the branch of a tree and produce a clean background plate.

- ***F_WireRemoval*** – In a similar vein, F_WireRemoval removes wires from an image. It is particularly good at dealing with complex shots where background replacement is not possible, for example, when wires cross the actors or where wires cross complex moving backgrounds such as trees, clouds or smoke. Rather than relying on traditional edge-stitching or cloning techniques, its patented algorithm performs complex signal processing to subtract the wire from the original image while preserving grain and background detail.

FurnaceCore... Why?

The twelve plug-ins that make up FurnaceCore encapsulate the key technologies of the Furnace toolset, which won our developers a Scientific and Technical Engineering Award from the Academy of Motion Picture Arts and Sciences in 2006. These tools all use state-of-the-art technology, much of it hinging on motion estimation, to reduce the hard grind of day-to-day compositing.

By including FurnaceCore in NukeX, we are furthering our goal of giving compositing artists greater creative freedom and a smoother, more efficient workflow. Reducing the time that needs to be spent on repetitive, uninspiring yet essential tasks such as wire removal and image alignment will free up more time for the kind of original, imaginative work that only artists can do.

Nuke and NukeX

It's worth a quick digression here to explain how the addition of NukeX can complement, rather than replace, existing Nuke workstations. All NukeX scripts can also be rendered inside Nuke, though the NukeX-specific nodes can only be adjusted or used to analyse inside NukeX. This supports easy and efficient script exchange between artists working in Nuke and NukeX, as well as allowing a more flexible licensing model. Nuke and NukeX share the same render licenses, so a Nuke render farm will be able to render all scripts regardless of whether they were created in Nuke or NukeX.

Summary

Together, the new technologies inside NukeX – FurnaceCore, the 3D CameraTracker, DepthGenerator and LensDistortion tools – add up to deliver a compositing solution which is much more than just the sum of its parts. Tools such as FurnaceCore and the automatic LensDistortion analysis increase productivity by allowing computers to take over some of the relatively pedestrian compositing tasks at which they excel, leaving artists free to concentrate on more creative work. The same can be said of the CameraTracker: as we saw with the green screen example earlier, being able to pick up a 3D track for a shot can deliver immense efficiency gains over the standard 2D workflows.

Increasingly, and as many of our customers are proving every day, compositing is becoming a 3D process – traditionally an area in which Nuke excels. For the future, the easy access to 3D data provided by the CameraTracker will be critical to the development of NukeX. This will allow us to provide a wide range of new tools built around 3D, of which the DepthGenerator is just the start. Essentially, we believe that 3D tracking should be just as natural and integral part of compositing as 2D tracking. With NukeX, for the first time, we present to you a compositing solution in which it can be. We look forward to seeing what you do with it.