Arbitrary Output Variables

Arbitrary Output Variables (AOVs) are the secondary images produced by the renderer. There can be any number of them generated simultaneously and each one may go to a different file, a different display driver, or use different pixel filter settings. In some cases, special pixel filter modes may be used to avoid mixing values from different samples in a non-sensical way; typically these select a single sample to be representative of the whole pixel.

Broadly speaking, AOVs in RIS fall into these main categories: built-in, integrator (global), custom, and light path expressions.

For users interested in what is typically referred to as "render passes" for compositing a beauty image, look at the page on light path expressions (LPE).

Built-in AOVs

The built-in AOVs display mostly geometric information pertaining to points on the visible surfaces seen by the camera. These values are automatically generated by the renderer and are available as AOVs regardless of the active integrator. Here is the current list of available builtin-ins:

Notice that *data* passes (normals, z-depth, world position, etc) are not filtered by default. Filtering together values may result in errors and nonsense values so it is avoided. Additive passes meant for beauty compositing are typically filtered the same to provide clean images and improved anti-aliasing.

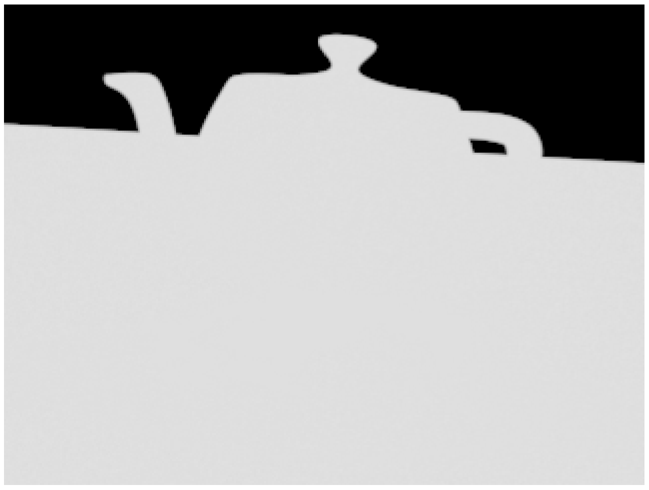
id, rawId, z all use zmin filtering by default.

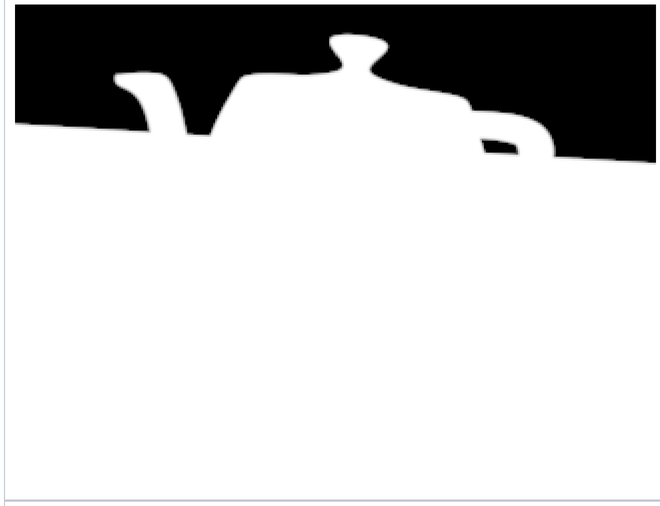
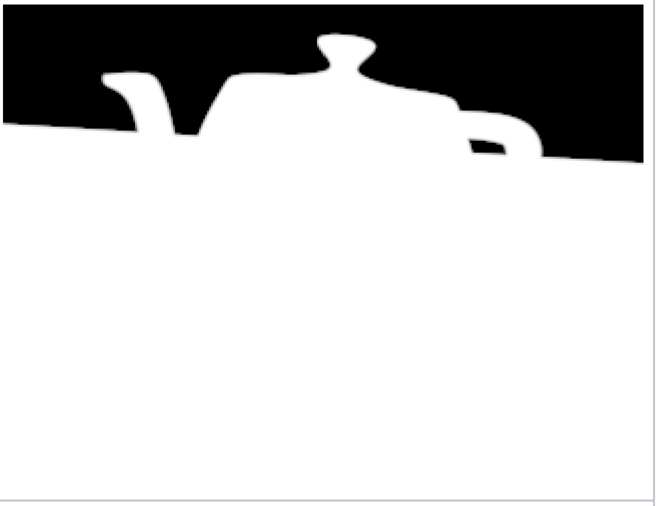
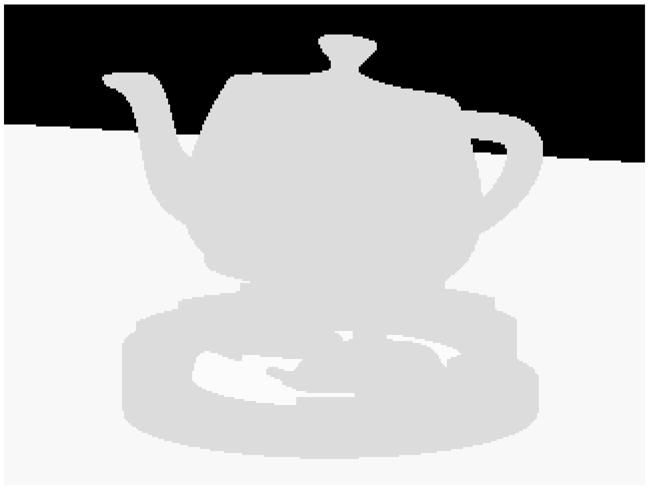
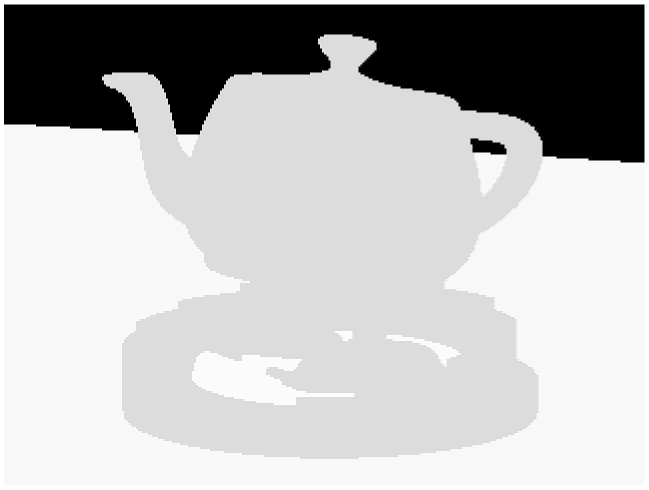
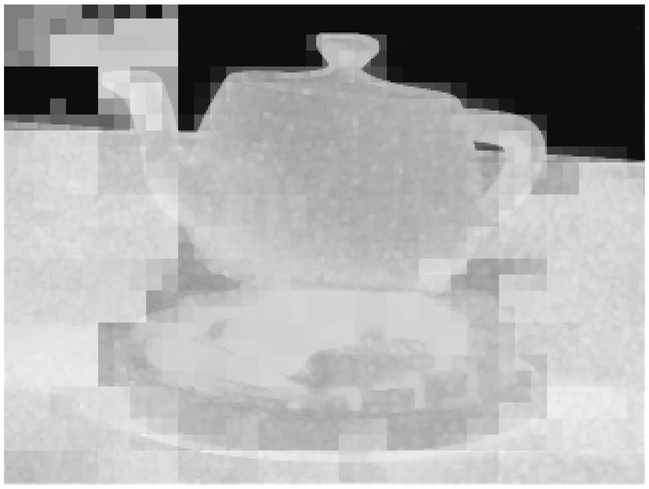
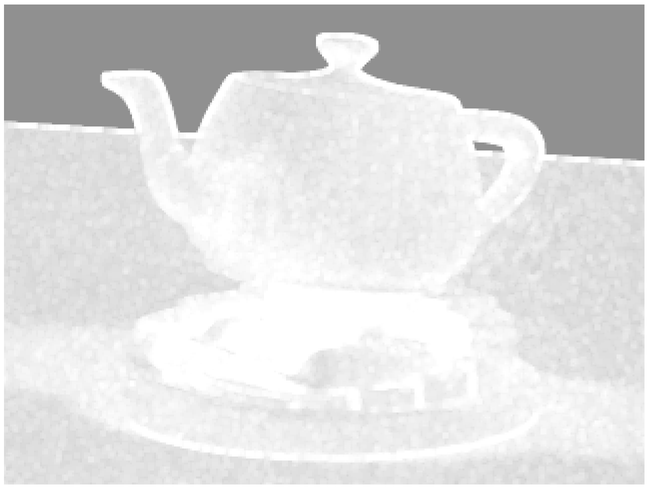
cpuTime and sampleCount use the sum filter by default.

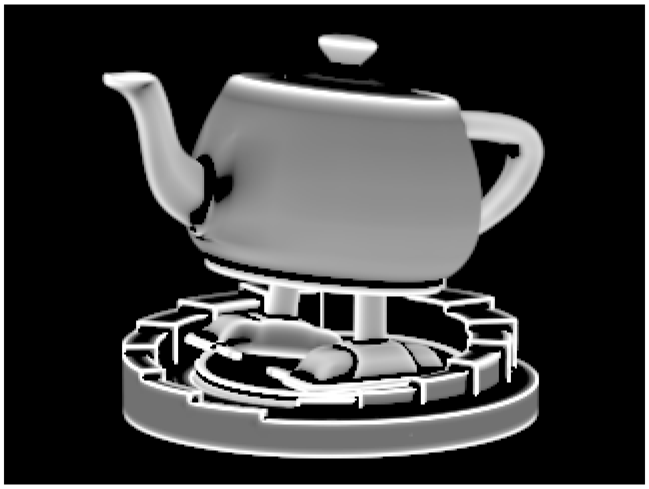
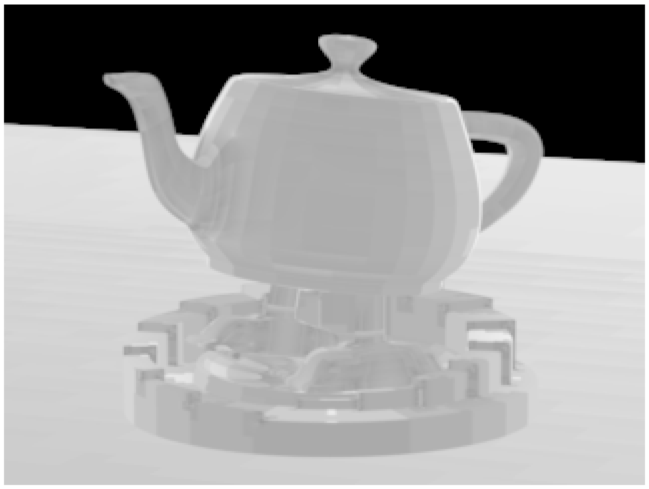
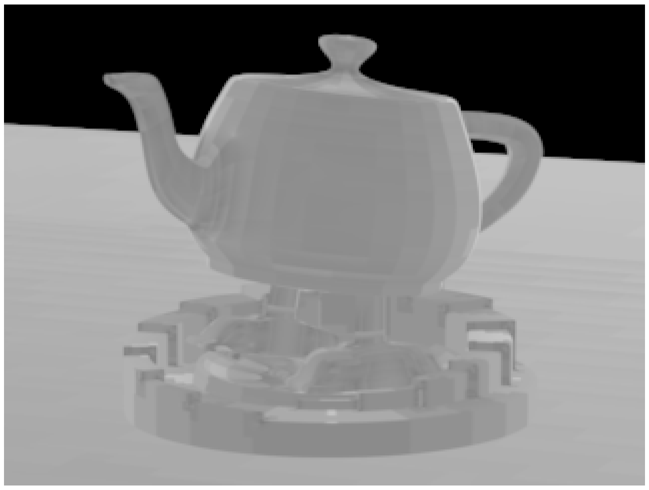
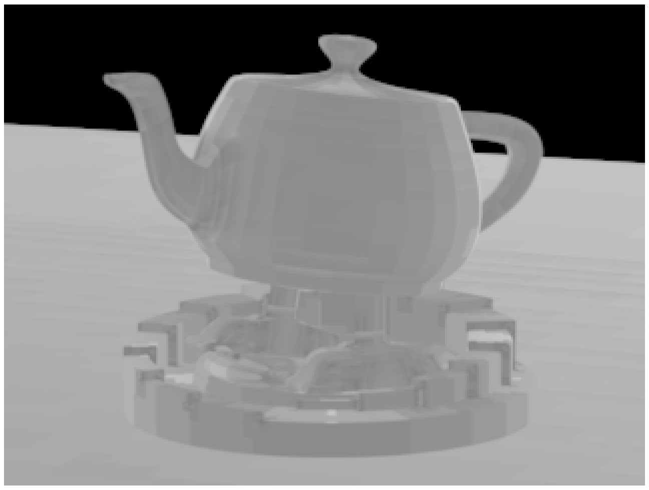
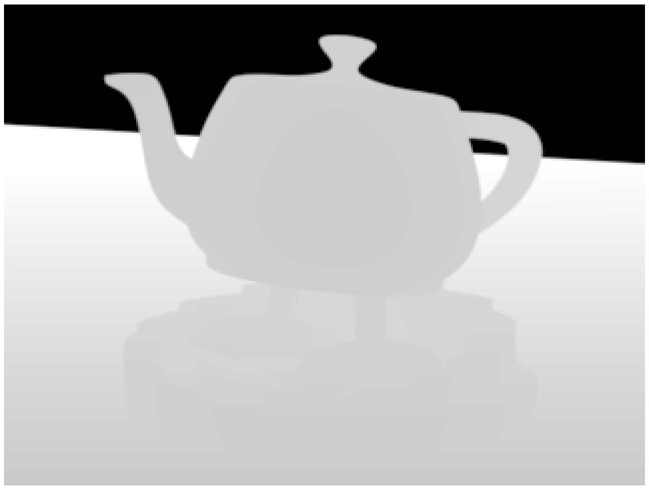
Post motion blur (2D Motion Blur vectors) can be important to a pipeline. While we recommend using motion blur in the render, sometimes a lack of resources forces this to post processing.

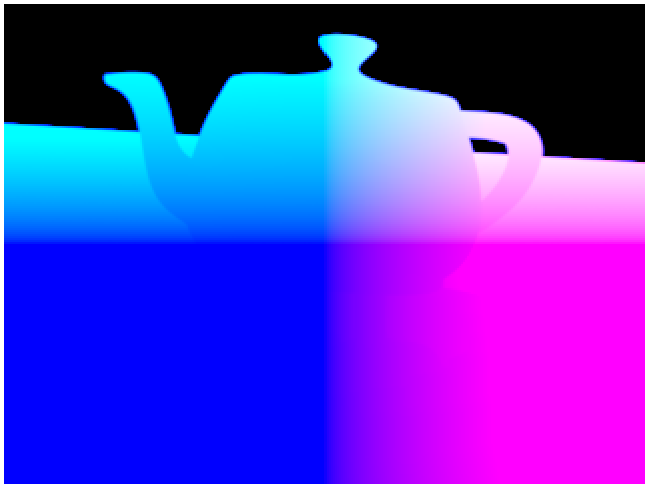
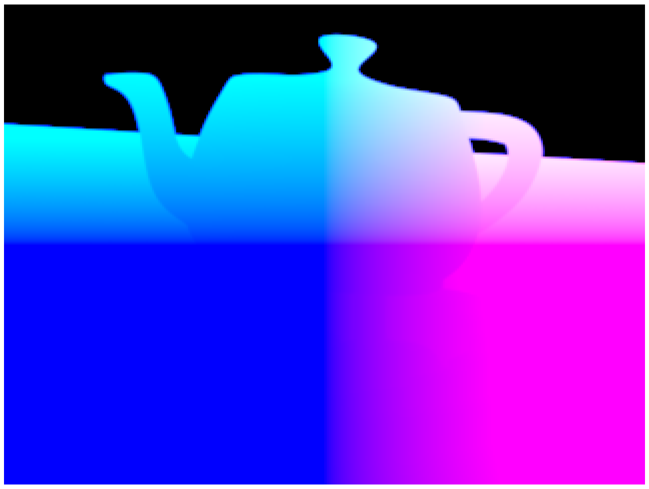
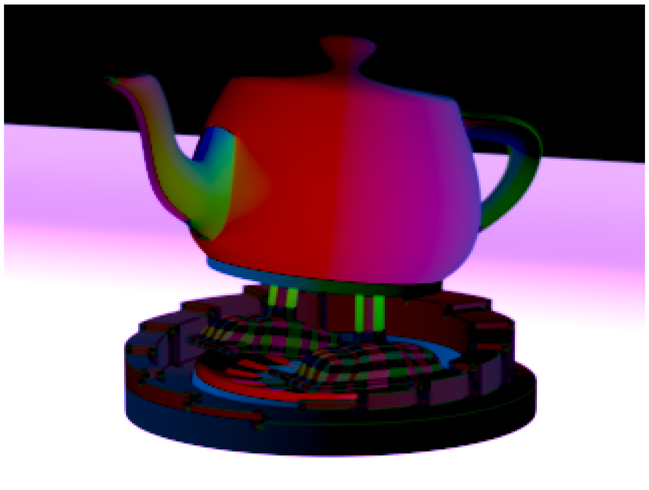
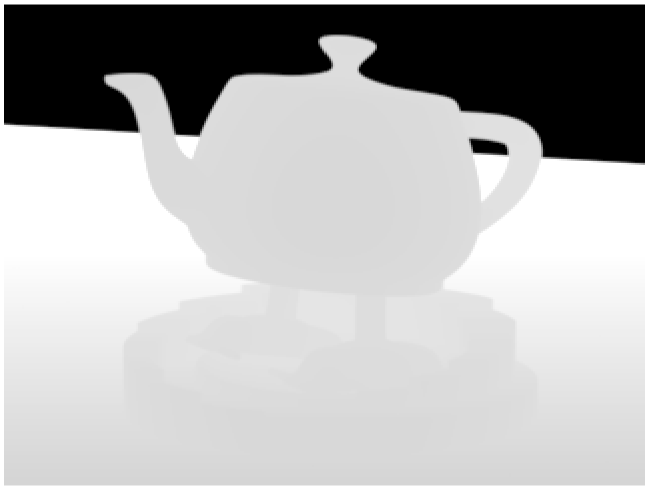
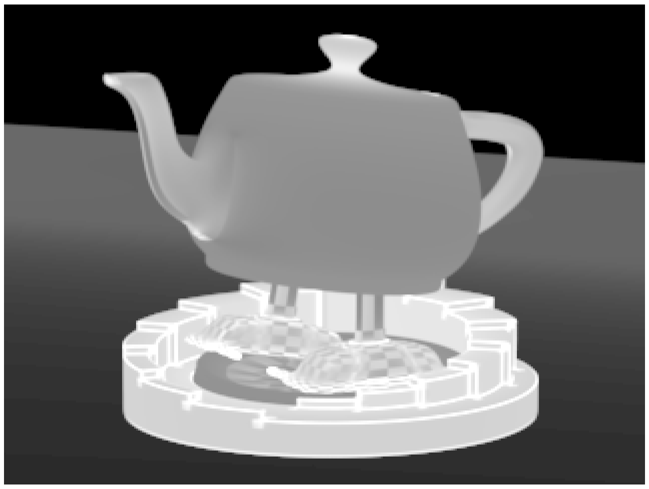
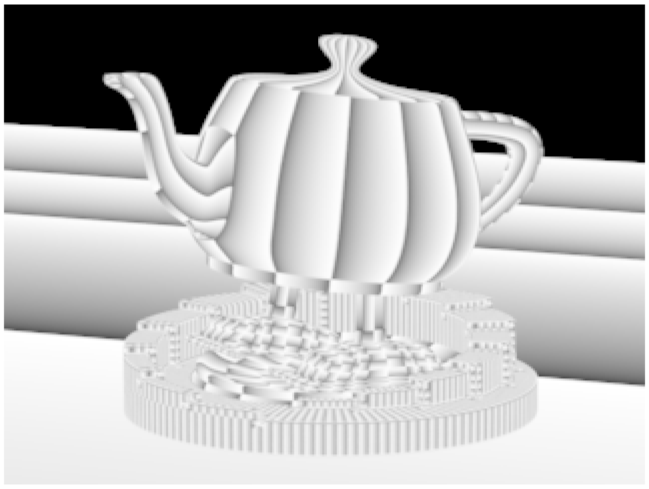
dPdtime AOVs are useful for post blurring in 2D but this does not contain camera motion blur (the camera movement)

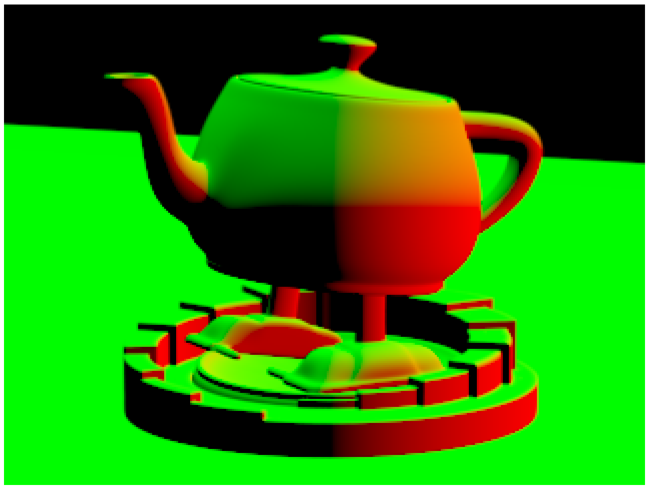
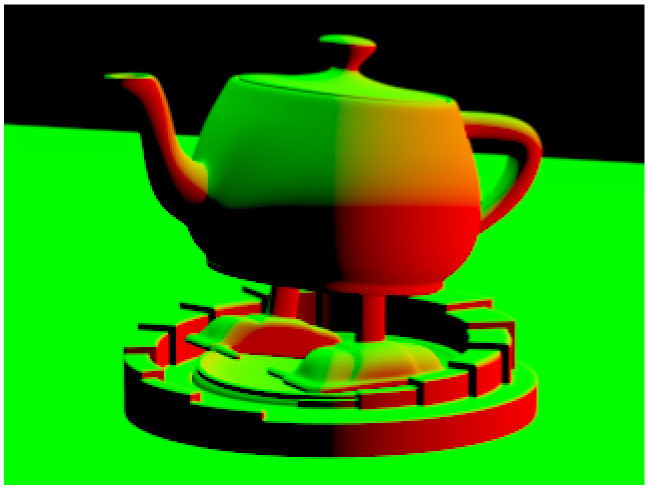
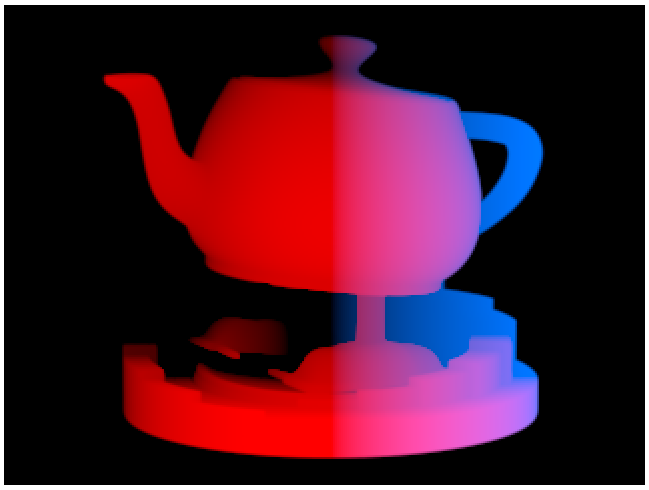
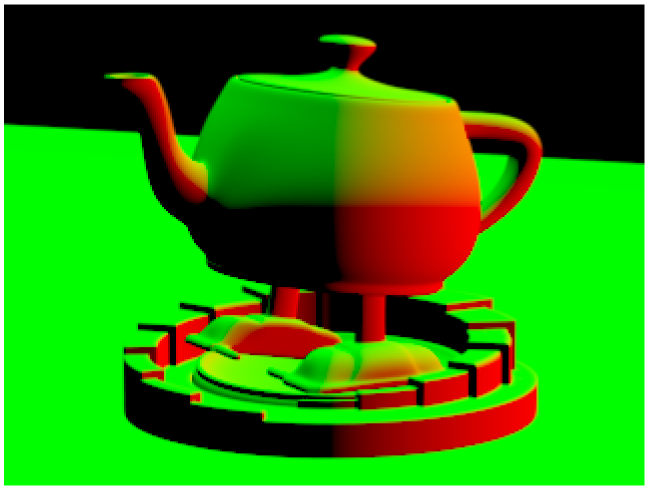
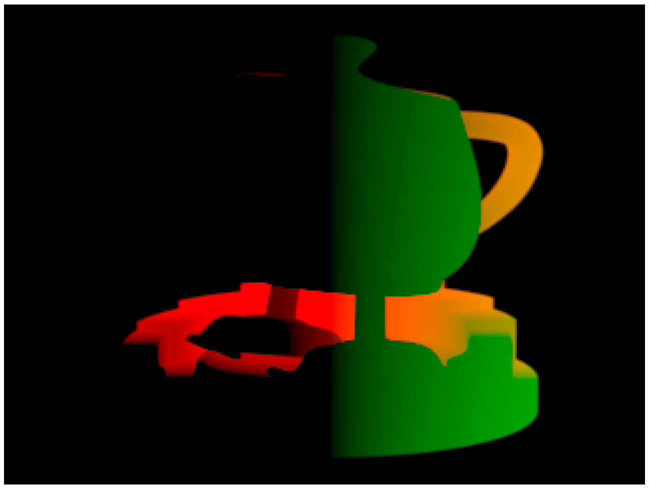
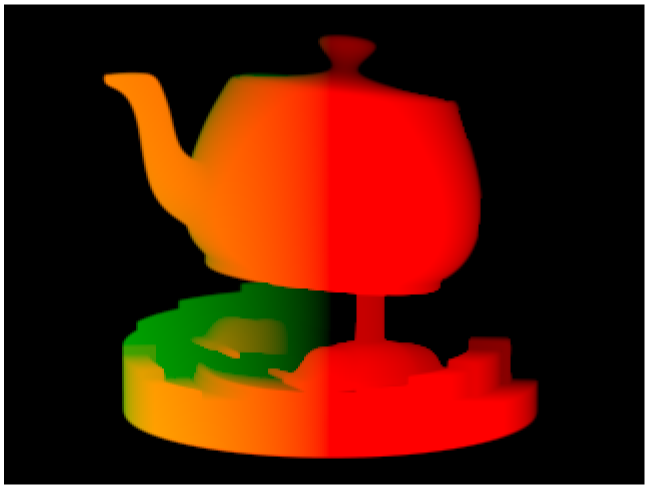
dPcameradtime AOVs are available for camera motion blur. This means motion blur can be separated and controlled with finer detail, especially if the camera movement introduced more blur than you'd like artistically, you can adjust it separately. A static camera will produce no results for this AOV and thus no example below. If you render from a locked camera then it can be omitted without penalty.

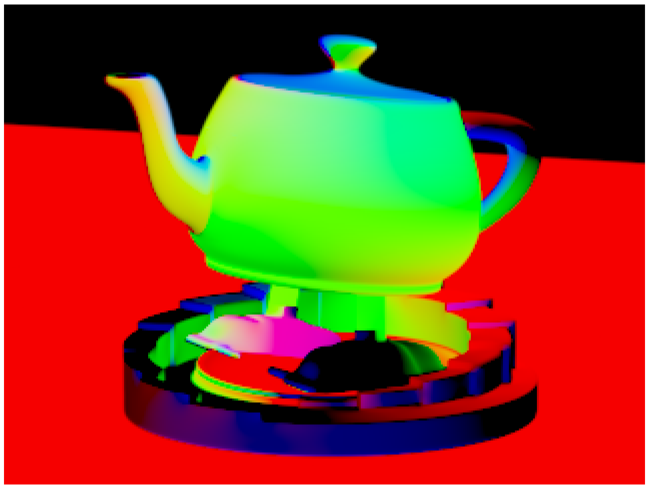
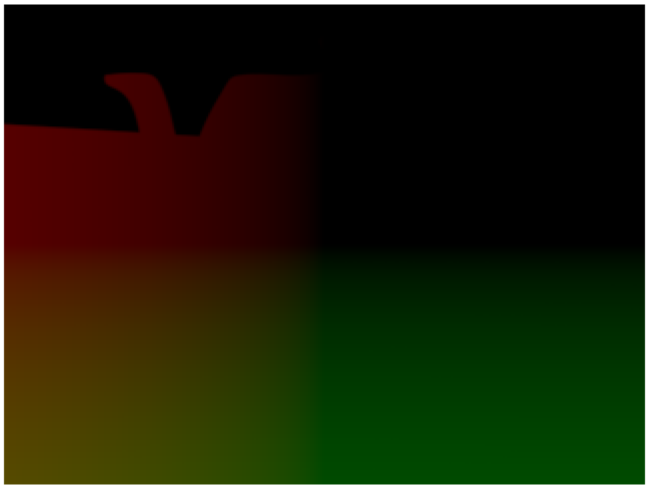
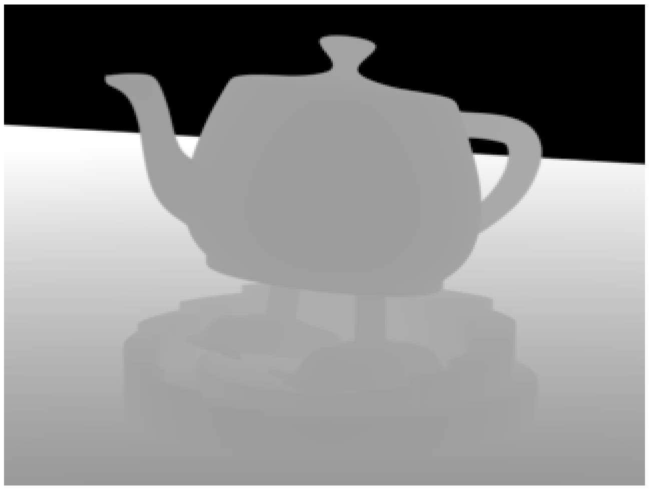
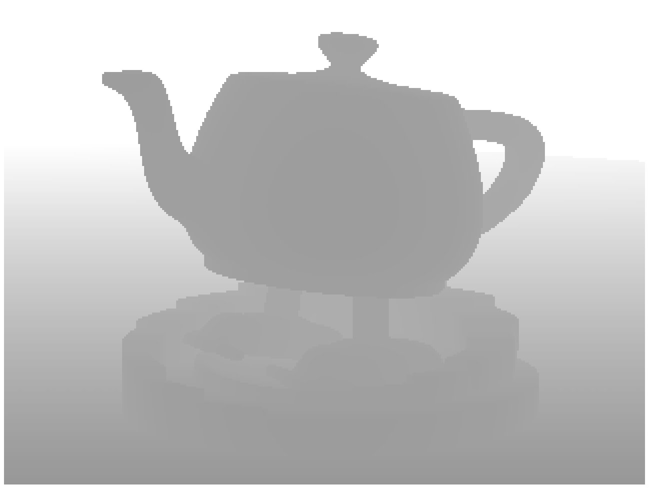
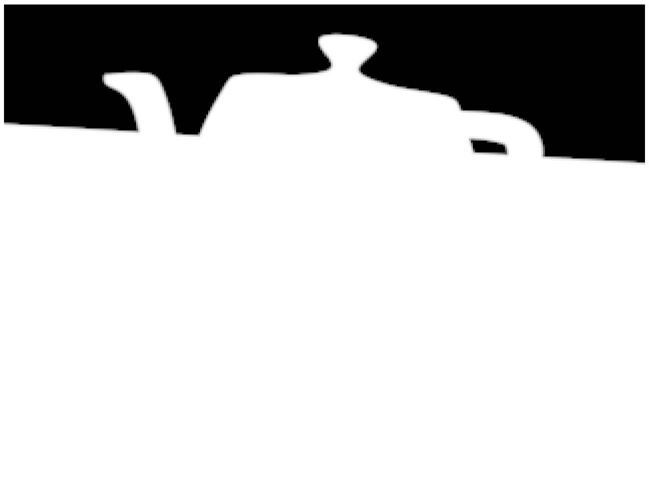
	
color Ci (Final "beauty" color)	float time (Average shading time, normalized between shutter open and close)

	
float a (Alpha)	color Oi (Colored opacity)
	
float id (Number assigned via the identifier attribute as the pixel value)	float rawId (Number assigned via the identifier attribute without conversion to float)
	
float cpuTime (Approximate compute time taken to render a pixel)	float sampleCount (Number of samples taken for the resulting pixel)

	
float curvature (Local surface curvature)	float mpSize (Size of the micropolygon the ray hit)
	
float biasR (Bias applied for reflected rays)	float biasT (Bias applied for transmitted rays)
	
float incidentRayRadius (Radius of incident ray at P)	float incidentRaySpread (Increase in ray radius for each unit distance travelled)

	
point P (Position of the point hit by the incident ray in camera space)	point Po (Position of the hit point before displacement)
	
vector dPdu, dPdv, dPdw (Direction of maximal change in u, v, and w)	float PRadius (Cross-sectional size of the ray at the hit point)
	
float du, dv, dw (Ray footprint or radius in u, v, or w)	float u, v, w (Parametric coordinates on the primitive)

	
normal Ngn (Normalized geometric normal)	normal Nn (Normalized shading normal)
	
vector dPdtime (Instantaneous velocity in 3D, this does not contain camera movement)	normal Non (Normalized shading normal before displacement)
	
vector motionBack (Backward 2D raster-space motion vector)	vector motionFore (Forward 2D raster-space motion vector)

	
vector Tn (Normalized shading tangent)	vector Vn (Normalized view vector)
	
float VLen (Length of view vector)	float z (Depth to ray hit in camera space)
	
float outsideIOR (Incident index of refraction)	

Integrator (Global) AOVs

On top of regular LPE-based AOVs, this integrator outputs a number of standard AOVs typically used by compositors.

Declaration	Content	Channels
`color __Pworld`	P in world-space	__Pworld.r : x component __Pworld.g : y component __Pworld.b : z component
`color __Nworld`	Nn in world-space	__Nworld.r : x component __Nworld.g : y component __Nworld.b : z component
`color __depth`	Multi-purpose AOV	__depth.r : depth from camera in world-space __depth.g : height in world-space __depth.b : geometric facing ratio : abs(Nn.V)
`color __st`	Texture coords	__st.x : s __st.y : t __st.z : 0.0
`color __Pref`	Reference Position primvar (if available)	__Pref.r : x component __Pref.g : y component __Pref.b : z component
`color __Nref`	Reference Normal primvar (if available)	__Nref.r : x component __Nref.g : y component __Nref.b : z component
`color __WPref`	Reference World Position primvar (if available)	__WPref.r : x component __WPref.g : y component __WPref.b : z component
`color __WNref`	Reference World Normal primvar (if available)	__WNref.r : x component __WNref.g : y component __WNref.b : z component

Custom AOVs

In addition to the above, some custom shading plugins may recognize other requests for AOVs and respond to them in their own particular ways. The names of these AOVs and what exactly gets displayed is up to the particular plugin.