ANARI Working Group Update

SIGGRAPH 2021 Khronos Fast Forward
Outline

• Why was ANARI created?
• What does ANARI look like?
• How has the ANARI progressed?
Why was ANARI created?
Industry Need and Opportunity

• New rendering technology - including ray tracing - is impacting visualization
  - Accurate generation of imagery - global illumination!
  - Sophisticated visual cues provide intuitive understanding of complex data

• But building efficient, portable renderer using low-level APIs - such as Vulkan - is often outside an ISV’s mission
  - Rendering is just a necessary technique to be utilized
  - True for scientific/CAD visualization, 3D animation + professional rendering, and emerging data analytics space

• Define a common higher level API to facilitate advances and broad adoption across visualization communities
  - Leveraging full potential of modern rendering techniques
  - Platform independent
  - Portable code
ANARI (Analytic Rendering Interface) Goals

Analytic Rendering Portability

Common API to describe objects in a scene
API to build the description of a scene rather than specifying the details of the rendering process
Rendering details left to the implementation of the API
Subset of more general scene graph APIs
ANARI Benefits

- Designed for scientific and scalable rendering
- Use state-of-the-art renderers and rendering techniques
  - Exploit hardware-optimized renderers
- Interactivity crucial for exploratory visualization
  - Low overhead APIs
  - Direct coupling with applications
  - Async APIs overlap compute+viz
  - Exploit parallel computing techniques
- In-situ, visualization of massive memory-resident data sets
  - Render live in-memory app data
  - Use data in-place when possible

~1 billion atom protocell membrane w/ ~1400 proteins (VMD)
Strong ANARI Industry Support

All Layers in the Scientific Visualization Stack are Represented
GPU Vendors, Rendering Engines, Visualization Libraries and Tools, Applications
What does ANARI look like?
API Design

- C99 based API
  - optional C++ bindings for added convenience, type safety
- Based around opaque object handles
  - Named types (e.g. “triangle” and “sphere” geometry types)
    \[
    \text{ANARI} \text{Geometry geometry} = \text{anariNewGeometry(“triangle”);}
    \]
  - Named parameters (e.g. “intensity” on lights)
    \[
    \text{anariSetParam(geometry, “vertex.position”, ANARI\_ARRAY1D, arrayHandle);}
    \]
- To make pixels:

  \[
  \text{void anariRenderFrame(ANARI} \text{Device, ANARI} \text{Frame);}
  \]

Ongoing discussions about rendering operations, scene updates, and continuations to better support async and greedy rendering...
Updated features defined in Core

- Optional debug/interface layer tracks API calls
- More focus on asynchronous API
- Additional focus on camera sets
- Array API tuned to minimize copies
- Revamped materials
- Simultaneous devices
- Runtime query enumeration of feature set(s)

NOTE: Specifics about several of these are under active refinement.

Capture API traces with in-lined validation messages that can be compiled standalone.
Runtime Enumeration of ANARI devices, features

- ANARI supports multiple backends (simultaneously), with varying capabilities
- New runtime query of capabilities
  - Enable “matchmaking” of app requirements to devices
  - Feature profiles for high level feature sets plus queries for specific extensions
  - New devices can augment existing "shrink wrapped" application binaries
  - Enable dynamic graphical user and scripting interfaces
How has ANARI progressed?
Progress So Far

- Defined and prioritized uses
- Investigated existing APIs, selected starting-point API
- Wrote first draft specification
- Exploratory implementations
- Exploratory integrations
- Identified friction points, discussing API revisions
- Refining/extending spec
- Receiving advisory group feedback

Approach is largely “implement demonstration first, then write the spec”
ANARI Timeline

Analytic Rendering Exploratory Group
Formed
November 2019

ANARI Working Group
Announced
3rd March 2020

TARGET
ANARI Provisional Specification
Mid 2021 (SC21)

TARGET
First ANARI Specification
2022
Get Involved in Developing ANARI!

Join Khronos and the ANARI Working Group
Have a voice and a vote in the design of the ANARI specification
Gain early access to alpha release APIs, tools, renderers
Fast track ANARI for your own renderer or hardware

Send us your feedback and requirements
What rendering features important to your application domain?
What new application domains and use cases would you use ANARI for?

https://www.khronos.org/anari
anari-chair@lists.khronos.org