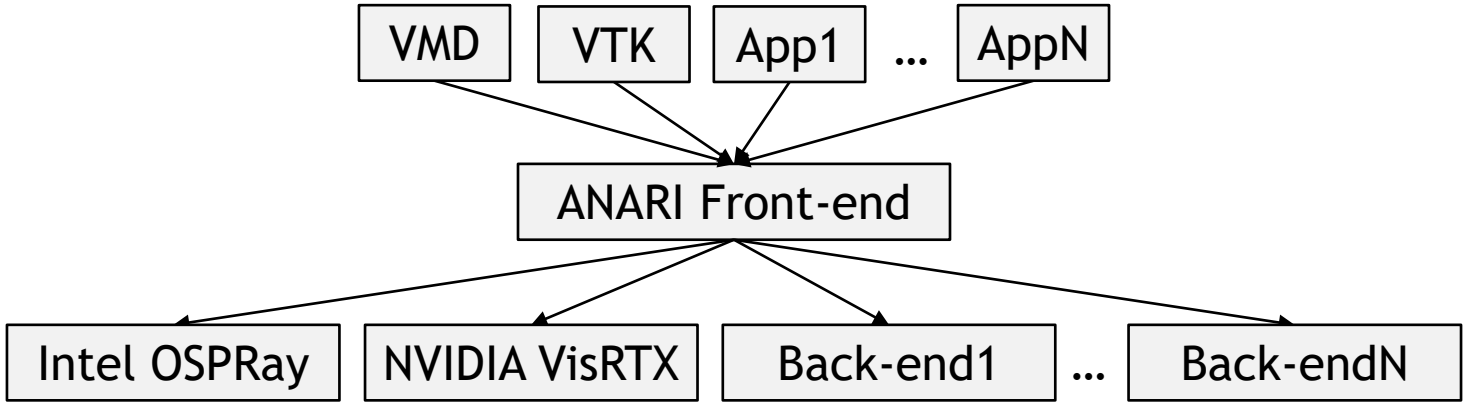
How to Create an ANARI™ Implementation

Johannes Günther, Intel
March 2022

Outline

- **SDK** https://github.com/KhronosGroup/**ANARI-SDK**
  - Front-end library
  - Utilities
  - Example back-end
  - Sample viewer & regression tests
- **Translation layer: ANARI to OSPRay**

Front-end library libs/anari (Sec 2.2)



- **API.cpp: entry anariLoadLibrary**

```
extern "C" const char **anariGetDeviceSubtypes(ANARILibrary l)
{
  return libraryRef(l).getDeviceSubtypes();
}
...
extern "C" ANARILight anariNewLight(ANARIDevice d, const char *type)
{
  return deviceRef(d).newLight(type);
}
```

Library (Sec 3.9)

- include/anari/detail/Library.h
- Use macros to export function symbols for
- Init (optional): void anari_library_##libname##_init()
- Introspection (Sec 3.6):
  - const char **anari_library_##libname##_get_device_subtypes(void *libdata)
  - …
- Modules, …
- Main object:
  - ANARIDevice anari_library_##libname##_new_device(const char *type)

Device (Sec 3.10)

- include/anari/detail/Device.h
- Derive from Device and implement pure virtual functions

```
struct ANARI_INTERFACE Device
{
  virtual int deviceImplements(const char *extension) = 0;
  virtual void deviceSetParameter(const char *id, ANARIDataType, const void *mem) = 0;
  virtual void deviceCommit() = 0;
...
  virtual ANARIArray1D newArray1D(void *appMemory,
      ANARIMemoryDeleter,
      void *userdata,
      ANARIDataType,
      uint64_t numItems1,
      uint64_t byteStride1) = 0;
...
  virtual ANARILight newLight(const char *type) = 0;
...
}
```

Utilities libs/anari_utilities

- **RefCounted** (Sec 3.4)
- **size_t sizeOfDataType(ANARIDataType)**
- **constexpr bool isObject(ANARIDataType)**
- **ParameterizedObject** (Sec 3.5)

```
struct ANARI_INTERFACE ParameterizedObject
{
...
  template <typename T>
  void setParam(const std::string &name, const T &t);

  template <typename T>
  T getParam(const std::string &name, T valIfNotFound);

  bool hasParam(const std::string &name);
...
}
```

Example Back-end examples/example_device

- Exporting Library function symbols

```
extern "C" EXAMPLE_DEVICE_INTERFACE ANARI_DEFINE_LIBRARY_INIT(example)
{
  printf("...loaded example library!\n");
}

extern "C" EXAMPLE_DEVICE_INTERFACE ANARI_DEFINE_LIBRARY_NEW_DEVICE(example, subtype)
{
    return (ANARIDevice) new anari::example_device::ExampleDevice();
}
...
```

Example Back-end: Device

```
struct EXAMPLE_DEVICE_INTERFACE ExampleDevice : public Device,
                                                public RefCounted,
                                                public ParameterizedObject
{
  int deviceImplements(const char *extension) override;
  void deviceSetParameter(const char *id, ANARIDataType, const void *mem) override;
  void deviceCommit() override;
...
  ANARIArray1D newArray1D(void *appMemory,
      ANARIMemoryDeleter deleter,
      void *userdata,
      ANARIDataType,
      uint64_t numItems1,
      uint64_t byteStride1) override;
...
  ANARILight newLight(const char *type) override;
...
}
```

Example Back-end: Setting Parameter

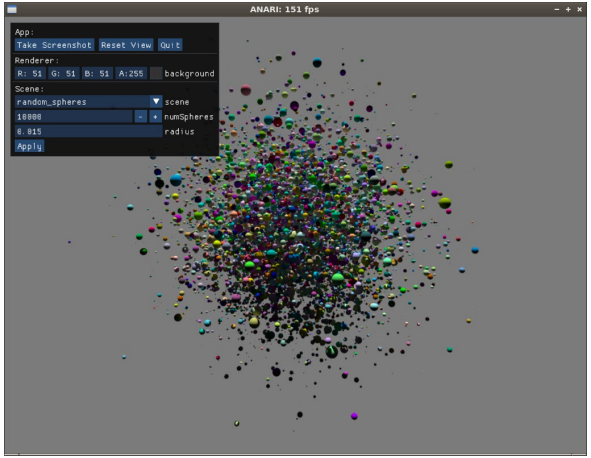
```
void ExampleDevice::setParameter(
ANARIObject object, const char *name, ANARIDataType type, const void *mem)
{
...
  auto *fcn = setParamFcns[type];
  if (fcn)
    fcn(object, name, mem);
}

static std::map<int, SetParamFcn *> setParamFcns = {
    declare_param_setter(int),
    declare_param_setter(float),
...
}

#define declare_param_setter(TYPE)                                    \
  {                                                                   \
    ANARITypeFor<TYPE>::value,                                        \
        [](ANARIObject o, const char *p, const void *v) {             \
          referenceFromHandle(o).setParam(p, v);                      \
        }                                                             \
  }
```

Example Back-end: Getting Parameter

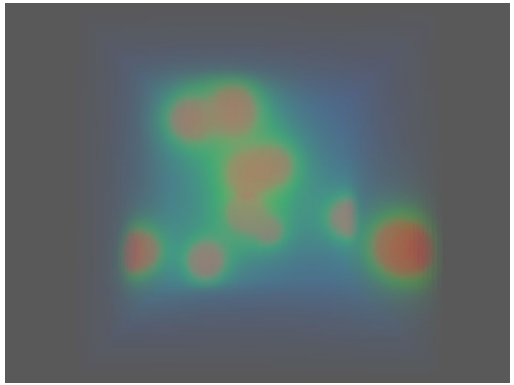
```
void Perspective::commit()
{
...
  float fovy = getParam<float>("fovy", glm::radians(60.f));
  float aspect = getParam<float>("aspect", 1.f);
}
```

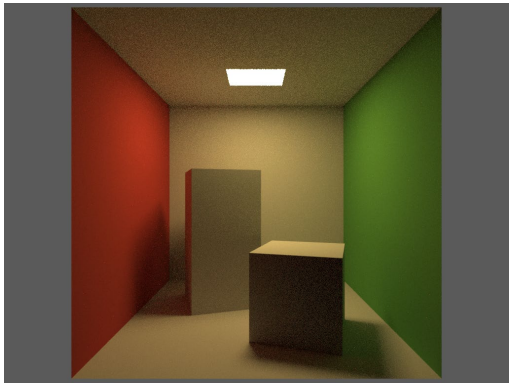
Sample Viewer & Regression Tests

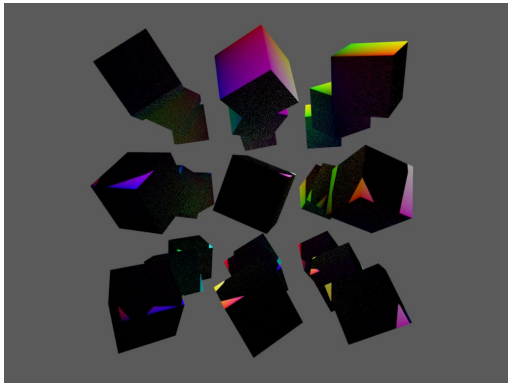
- viewer



- anari_regression_tests





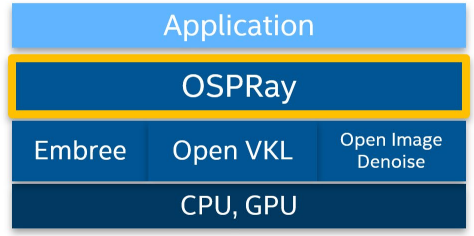


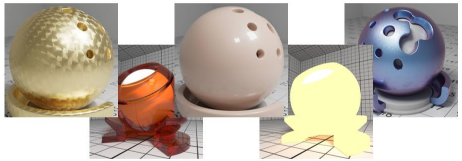
ANARI-OSPRay

https://github.com/ospray/**anari-ospray**

Intel® OSPRay, High-level Features

Combined geometry and volume rendering

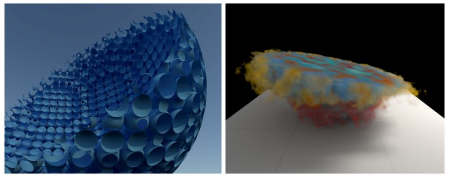




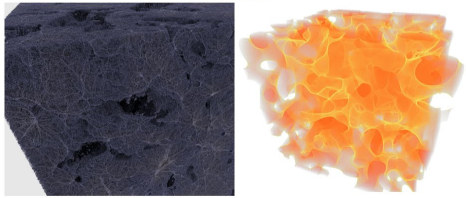
Advanced materials

SciVis renderer and path tracer

Runs at all scales



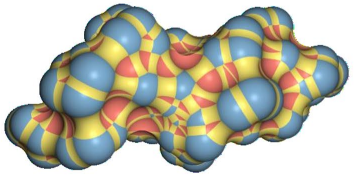
Clipping Geometries



12 billion particles

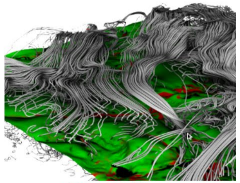
10TB

Huge datasets

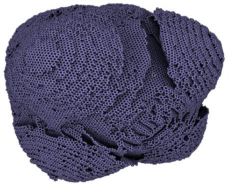


Extensible

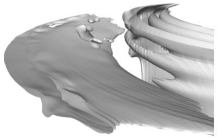
e.g. SES Geometry, UStuttgart



Streamlines



Particles / spheres



Implicit isosurfaces

Non-polygonal Geometry

Base Class, Setting Parameter

- Wraps OSPRay Object

```
void Object::setParam(const char *id, ANARIDataType type, const void *mem)
{
  if (!m_object)
    return;

  if (anari::isObject(type)) {
    setObjectParam(id, type, (*(Object **)mem));
  } else {
    ospSetParam(m_object, id, enumCast<OSPDataType>(type), mem);
  }
}

void Object::setObjectParam(
    const char *id, ANARIDataType type, const Object *obj)
{
  if (obj) {
    auto handle = obj->handle();
    ospSetParam(m_object, id, enumCast<OSPDataType>(type), &handle);
  }
}
```

Translate Enums

```
inline OSPDataType enumCast(int value)
{
  switch (value) {
  case ANARI_DEVICE:
    return OSP_DEVICE;
  case ANARI_BOOL:
    return OSP_BOOL;
  case ANARI_OBJECT:
    return OSP_OBJECT;
  case ANARI_ARRAY:
  case ANARI_ARRAY1D:
  case ANARI_ARRAY2D:
  case ANARI_ARRAY3D:
    return OSP_DATA;
  case ANARI_SURFACE:
    return OSP_GEOMETRIC_MODEL;
  case ANARI_GEOMETRY:
    return OSP_GEOMETRY;
...
}
```

Translate Objects, Parameters, Units

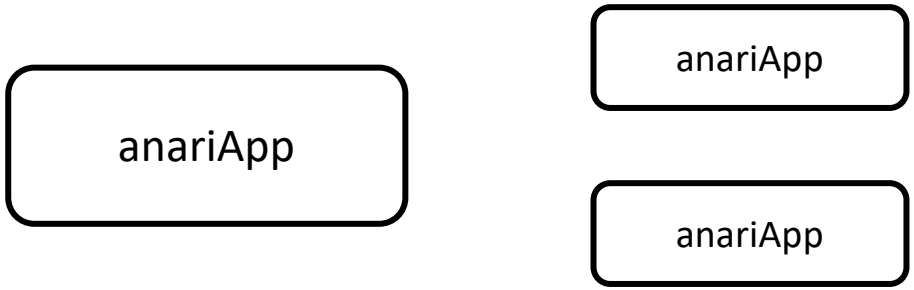
```
Camera::Camera(std::string subtype)
    : Object(ospNewCamera(subtype.c_str()), subtype)
{}

void Camera::setParam(const char *_id, ANARIDataType type, const void *mem)
{
  std::string id(_id);

  if (id == "imageRegion") {
    Object::setParam("imageStart", ANARI_FLOAT32_VEC2, (const float *)mem);
    Object::setParam("imageEnd", ANARI_FLOAT32_VEC2, ((const float *)mem) + 2);
    return;
  }
  if (id == "fovy") {
    float fovy = rad2deg(*(const float *)mem);
    Object::setParam(_id, type, &fovy);
    return;
  }

  Object::setParam(id.c_str(), type, mem);
}
```

Hints

- Lazy and sorted commits
- Where to implement
- Vendor extensions

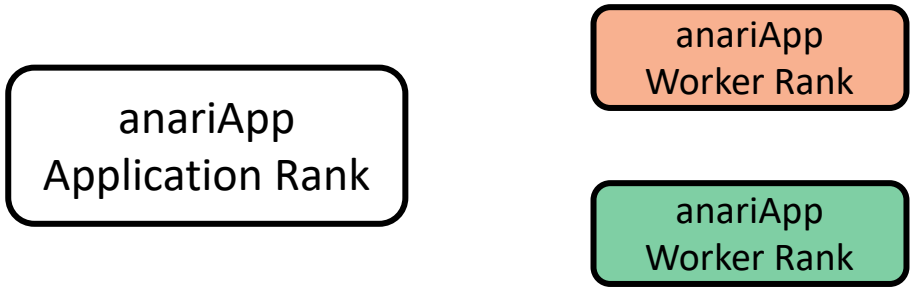
MPI Distributed Rendering, MPI Offload Device

```
export ANARI_LIBRARY=ospray
export OSPRAY_LOAD_MODULES=mpi
export OSPRAY_DEVICE=mpiOffload
mpirun -n 3 ./anariApp
```



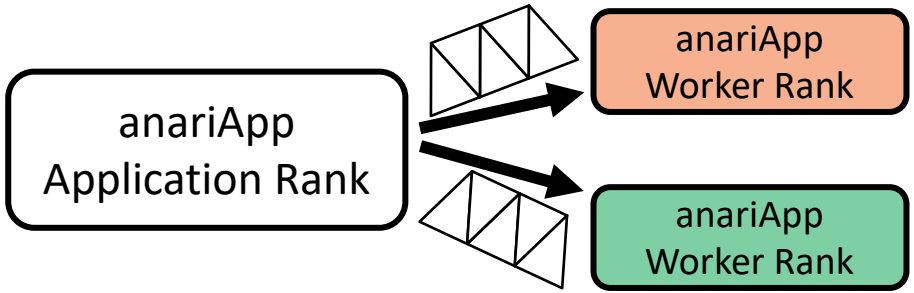
MPI Distributed Rendering, Spawn Ranks

```
export ANARI_LIBRARY=ospray
export OSPRAY_LOAD_MODULES=mpi
export OSPRAY_DEVICE=mpiOffload
mpirun -n 3 ./anariApp
```



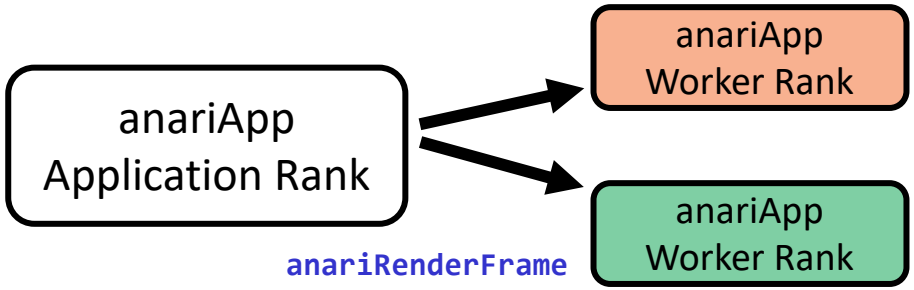
MPI Distributed Rendering, Cmd & Data Queue

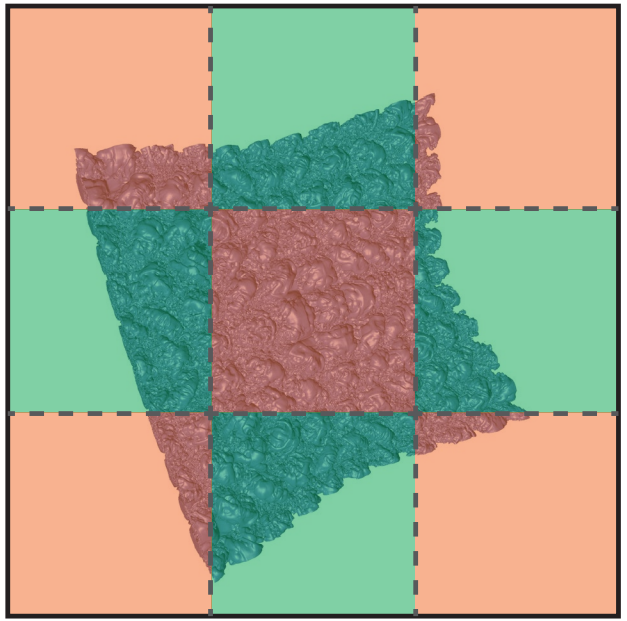
```
export ANARI_LIBRARY=ospray
export OSPRAY_LOAD_MODULES=mpi
export OSPRAY_DEVICE=mpiOffload
mpirun -n 3 ./anariApp
```



MPI Distributed Rendering, Tiled Rendering

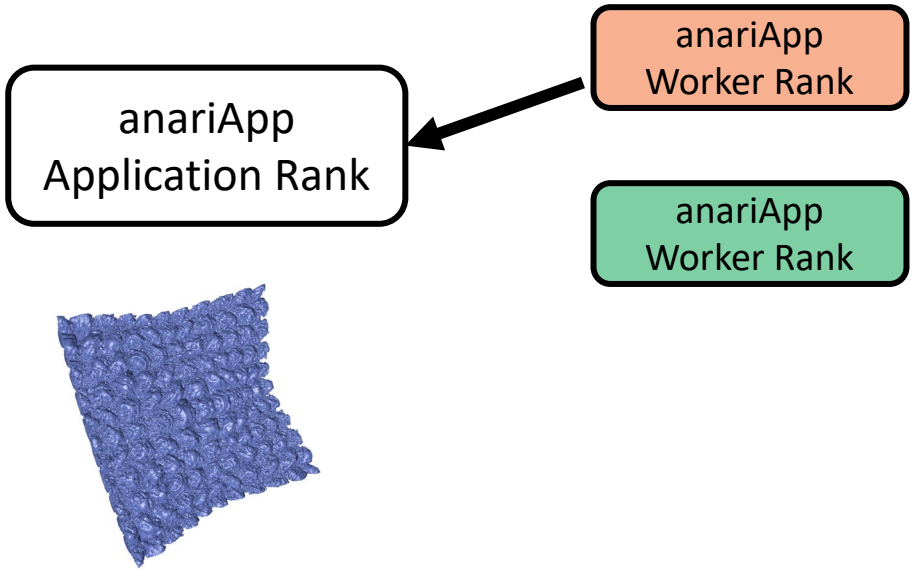
```
export ANARI_LIBRARY=ospray
export OSPRAY_LOAD_MODULES=mpi
export OSPRAY_DEVICE=mpiOffload
mpirun -n 3 ./anariApp
```

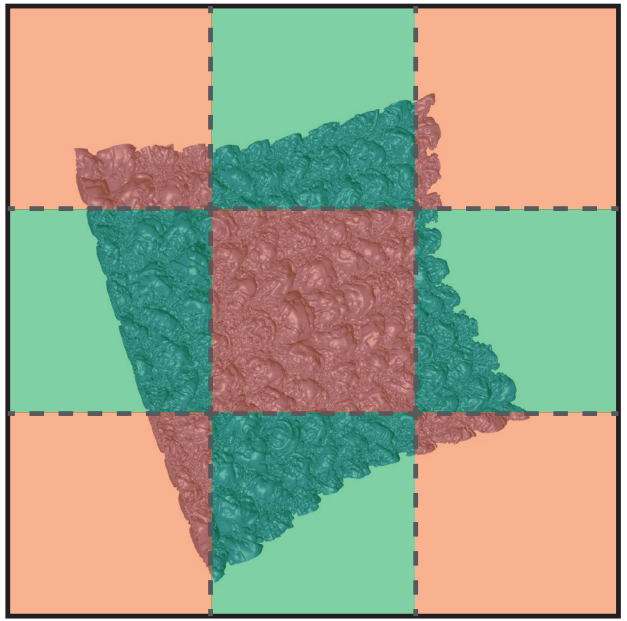




MPI Distributed Rendering, Compositing

```
export ANARI_LIBRARY=ospray
export OSPRAY_LOAD_MODULES=mpi
export OSPRAY_DEVICE=mpiOffload
mpirun -n 3 ./anariApp
```





KHRONOS GROUP